This document describes all of the commands that are available to use with the WebLogic Scripting Tool (WLST). This document includes WLST commands for WebLogic Server, as well as custom WLST commands that can be used to manage installed Oracle Fusion Middleware components.
# Contents

**Preface** ............................................................................................................................................................ xxxi  
Documentation Accessibility ........................................................................................................ xxxi  
Conventions ............................................................................................................................................. xxxi  

## 1 Introduction and Roadmap  
1.1 Document Scope and Audience ................................................................................................ 1-1  
1.2 Guide to This Document ........................................................................................................... 1-1  
1.3 Related Documentation .......................................................................................................... 1-3  
1.4 New and Changed WLST Features in This Release ................................................................ 1-3  

## 2 WebLogic Server WLST Online and Offline Command Reference  
2.1 WebLogic Server WLST Command Summary, Alphabetically By Command ...................... 2-1  
2.2 WebLogic Server WLST Online Command Summary .......................................................... 2-6  
2.3 WebLogic Server WLST Offline Command Summary .......................................................... 2-10  

## 3 WLST Command and Variable Reference  
3.1 Overview of WLST Command Categories .............................................................................. 3-1  
3.2 Browse Commands .................................................................................................................... 3-2  
3.2.1 cd ........................................................................................................................................... 3-2  
3.2.2 currentTree............................................................................................................................ 3-3  
3.2.3 prompt................................................................................................................................. 3-4  
3.2.4 pwd ........................................................................................................................................ 3-5  
3.3 Control Commands .................................................................................................................... 3-5  
3.3.1 addTemplate ........................................................................................................................ 3-6  
3.3.2 closeDomain ........................................................................................................................ 3-7  
3.3.3 closeTemplate ...................................................................................................................... 3-7  
3.3.4 connect ............................................................................................................................... 3-7  
3.3.5 createDomain ..................................................................................................................... 3-11  
3.3.6 disconnect ........................................................................................................................... 3-12  
3.3.7 exit ....................................................................................................................................... 3-13  
3.3.8 readDomain ....................................................................................................................... 3-13  
3.3.9 readTemplate ..................................................................................................................... 3-14  
3.3.10 updateDomain .................................................................................................................. 3-15  
3.3.11 writeDomain .................................................................................................................... 3-15  
3.3.12 writeTemplate .................................................................................................................. 3-16
<table>
<thead>
<tr>
<th>Section</th>
<th>Commands</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.4 Customization Commands</td>
<td>addListener</td>
<td>3-17</td>
</tr>
<tr>
<td></td>
<td>addHelpCommandGroup</td>
<td>3-17</td>
</tr>
<tr>
<td></td>
<td>addHelpCommand</td>
<td>3-18</td>
</tr>
<tr>
<td>3.5 Deployment Commands</td>
<td>deploy</td>
<td>3-19</td>
</tr>
<tr>
<td></td>
<td>distributeApplication</td>
<td>3-20</td>
</tr>
<tr>
<td></td>
<td>getWLDM</td>
<td>3-21</td>
</tr>
<tr>
<td></td>
<td>listApplications</td>
<td>3-22</td>
</tr>
<tr>
<td></td>
<td>loadApplication</td>
<td>3-23</td>
</tr>
<tr>
<td></td>
<td>redeploy</td>
<td>3-24</td>
</tr>
<tr>
<td></td>
<td>startApplication</td>
<td>3-25</td>
</tr>
<tr>
<td></td>
<td>stopApplication</td>
<td>3-26</td>
</tr>
<tr>
<td></td>
<td>stopEdit</td>
<td>3-27</td>
</tr>
<tr>
<td></td>
<td>updateApplication</td>
<td>3-28</td>
</tr>
<tr>
<td></td>
<td>encrypt</td>
<td>3-29</td>
</tr>
<tr>
<td></td>
<td>getAvailableCapturedImages</td>
<td>3-30</td>
</tr>
<tr>
<td></td>
<td>isRestartRequired</td>
<td>3-31</td>
</tr>
<tr>
<td></td>
<td>validate</td>
<td>3-32</td>
</tr>
<tr>
<td></td>
<td>invoke</td>
<td>3-33</td>
</tr>
<tr>
<td></td>
<td>delete</td>
<td>3-34</td>
</tr>
<tr>
<td></td>
<td>encryptImageCapture</td>
<td>3-35</td>
</tr>
<tr>
<td>3.6 Diagnostics Commands</td>
<td>deploy</td>
<td>3-36</td>
</tr>
<tr>
<td></td>
<td>distributeApplication</td>
<td>3-37</td>
</tr>
<tr>
<td></td>
<td>getWLDM</td>
<td>3-38</td>
</tr>
<tr>
<td></td>
<td>listApplications</td>
<td>3-39</td>
</tr>
<tr>
<td></td>
<td>loadApplication</td>
<td>3-40</td>
</tr>
<tr>
<td></td>
<td>redeploy</td>
<td>3-41</td>
</tr>
<tr>
<td></td>
<td>startApplication</td>
<td>3-42</td>
</tr>
<tr>
<td></td>
<td>stopApplication</td>
<td>3-43</td>
</tr>
<tr>
<td></td>
<td>updateApplication</td>
<td>3-44</td>
</tr>
<tr>
<td></td>
<td>encryptImageCapture</td>
<td>3-45</td>
</tr>
<tr>
<td>3.7 Editing Commands</td>
<td>activate</td>
<td>3-46</td>
</tr>
<tr>
<td></td>
<td>assign</td>
<td>3-47</td>
</tr>
<tr>
<td></td>
<td>cancelEdit</td>
<td>3-48</td>
</tr>
<tr>
<td></td>
<td>create</td>
<td>3-49</td>
</tr>
<tr>
<td></td>
<td>delete</td>
<td>3-50</td>
</tr>
<tr>
<td></td>
<td>encrypt</td>
<td>3-51</td>
</tr>
<tr>
<td></td>
<td>get</td>
<td>3-52</td>
</tr>
<tr>
<td></td>
<td>getActivationTask</td>
<td>3-53</td>
</tr>
<tr>
<td></td>
<td>invoke</td>
<td>3-54</td>
</tr>
<tr>
<td></td>
<td>isRestartRequired</td>
<td>3-55</td>
</tr>
<tr>
<td></td>
<td>loadDB</td>
<td>3-56</td>
</tr>
<tr>
<td></td>
<td>loadProperties</td>
<td>3-57</td>
</tr>
<tr>
<td></td>
<td>saveProperties</td>
<td>3-58</td>
</tr>
<tr>
<td></td>
<td>set</td>
<td>3-59</td>
</tr>
<tr>
<td></td>
<td>setOption</td>
<td>3-60</td>
</tr>
<tr>
<td></td>
<td>showChanges</td>
<td>3-61</td>
</tr>
<tr>
<td></td>
<td>startEdit</td>
<td>3-62</td>
</tr>
<tr>
<td></td>
<td>stopEdit</td>
<td>3-63</td>
</tr>
<tr>
<td></td>
<td>unassign</td>
<td>3-64</td>
</tr>
<tr>
<td></td>
<td>undo</td>
<td>3-65</td>
</tr>
<tr>
<td></td>
<td>validate</td>
<td>3-66</td>
</tr>
<tr>
<td>3.8 Information Commands</td>
<td>addListener</td>
<td>3-67</td>
</tr>
<tr>
<td></td>
<td>configToScript</td>
<td>3-68</td>
</tr>
<tr>
<td></td>
<td>dumpStack</td>
<td>3-69</td>
</tr>
<tr>
<td></td>
<td>dumpVariables</td>
<td>3-70</td>
</tr>
</tbody>
</table>
3.8.5 find ............................................................................................................................. 3-60
3.8.6getConfigManager ........................................................................................................... 3-61
3.8.7 getMBean ....................................................................................................................... 3-62
3.8.8 getMBI .......................................................................................................................... 3-62
3.8.9 getPath .......................................................................................................................... 3-63
3.8.10 listChildTypes ............................................................................................................. 3-63
3.8.11 lookup ......................................................................................................................... 3-64
3.8.12 ls ................................................................................................................................... 3-64
3.8.13 man ............................................................................................................................. 3-68
3.8.14 redirect ....................................................................................................................... 3-68
3.8.15 removeListener ........................................................................................................... 3-69
3.8.16 showListeners ............................................................................................................. 3-69
3.8.17 startRecording ............................................................................................................ 3-70
3.8.18 state ........................................................................................................................... 3-70
3.8.19 stopRecording ............................................................................................................ 3-71
3.8.20 stopRedirect ................................................................................................................ 3-72
3.8.21 storeUserConfig .......................................................................................................... 3-72
3.8.22 threadDump ................................................................................................................ 3-74
3.8.23 viewMBean ................................................................................................................ 3-74
3.8.24 writeIniFile ................................................................................................................ 3-75
3.9 Life Cycle Commands ...................................................................................................... 3-76
3.9.1 migrate ......................................................................................................................... 3-76
3.9.2 resume ......................................................................................................................... 3-77
3.9.3 shutdown ..................................................................................................................... 3-78
3.9.4 start ............................................................................................................................. 3-80
3.9.5 startServer .................................................................................................................. 3-80
3.9.6 suspend ....................................................................................................................... 3-82
3.10 Node Manager Commands ............................................................................................ 3-83
3.10.1 nm ............................................................................................................................. 3-84
3.10.2 nmConnect ................................................................................................................ 3-84
3.10.3 nmDisconnect .......................................................................................................... 3-86
3.10.4 nmEnroll .................................................................................................................. 3-87
3.10.5 nmGenBootStartupProps ......................................................................................... 3-88
3.10.6 nmKill ....................................................................................................................... 3-88
3.10.7 nmLog ......................................................................................................................... 3-89
3.10.8 nmServerLog ............................................................................................................. 3-90
3.10.9 nmServerStatus ....................................................................................................... 3-90
3.10.10 nmStart .................................................................................................................. 3-91
3.10.11 nmVersion .............................................................................................................. 3-92
3.10.12 startNodeManager ................................................................................................. 3-92
3.10.13 stopNodeManager ................................................................................................. 3-93
3.11 Tree Commands .......................................................................................................... 3-94
3.11.1 custom ........................................................................................................................ 3-94
3.11.2 domainConfig .......................................................................................................... 3-95
3.11.3 domainCustom ......................................................................................................... 3-96
3.11.4 domainRuntime ....................................................................................................... 3-97
3.11.5 edit ............................................................................................................................. 3-98
4 Infrastructure Security Custom WLST Commands

4.1 Overview of WLST Security Commands .................................................. 4-2
4.2 Audit Configuration Commands ................................................................. 4-2
  4.2.1 getNonJavaEAAuditMBeanName ................................................... 4-3
  4.2.2 getAuditPolicy ............................................................................. 4-3
  4.2.3 setAuditPolicy ............................................................................. 4-4
  4.2.4 getAuditRepository ..................................................................... 4-6
  4.2.5 setAuditRepository ..................................................................... 4-6
  4.2.6 listAuditEvents ......................................................................... 4-7
  4.2.7 exportAuditConfig ................................................................. 4-8
  4.2.8 importAuditConfig ................................................................. 4-9
  4.2.9 createAuditDBView ................................................................. 4-10
  4.2.10 listAuditComponents ............................................................. 4-10
  4.2.11 registerAudit .......................................................................... 4-11
  4.2.12 deregisterAudit ....................................................................... 4-11
4.3 SSL Configuration Commands ................................................................. 4-12
  4.3.1 addCertificateRequest ............................................................. 4-13
  4.3.2 addSelfSignedCertificate ....................................................... 4-13
  4.3.3 changeKeyStorePassword ...................................................... 4-14
  4.3.4 changeWalletPassword .......................................................... 4-15
  4.3.5 configureSSL .......................................................................... 4-15
  4.3.6 createKeyStore ......................................................................... 4-16
  4.3.7 createWallet ............................................................................. 4-17
  4.3.8 deleteKeyStore .......................................................................... 4-17
  4.3.9 deleteWallet ............................................................................. 4-18
  4.3.10 exportKeyStore ....................................................................... 4-18
  4.3.11 exportKeyStoreObject ......................................................... 4-19
  4.3.12 exportWallet ......................................................................... 4-20
  4.3.13 exportWalletObject .............................................................. 4-20
  4.3.14 generateKey ........................................................................... 4-22
  4.3.15 getKeyStoreObject ............................................................... 4-22
  4.3.16 getSSL .................................................................................. 4-23
  4.3.17 getWalletObject ..................................................................... 4-24
  4.3.18 importKeyStore ................................................................. 4-24
  4.3.19 importKeyStoreObject ........................................................ 4-25
  4.3.20 importWallet ........................................................................ 4-26
  4.3.21 importWalletObject ............................................................. 4-27
  4.3.22 listKeyStoreObjects ............................................................. 4-28
  4.3.23 listKeyStores ......................................................................... 4-28
  4.3.24 listWalletObjects ................................................................. 4-29
  4.3.25 listWallets ............................................................................. 4-29
  4.3.26 removeKeyStoreObject .......................................................... 4-30
<table>
<thead>
<tr>
<th>Section</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.3.27</td>
<td>removeWalletObject ........................................... 4-31</td>
</tr>
<tr>
<td>4.4</td>
<td>Oracle Identity Federation Commands .................................. 4-32</td>
</tr>
<tr>
<td>4.4.1</td>
<td>addConfigListEntryInMap ........................................ 4-34</td>
</tr>
<tr>
<td>4.4.2</td>
<td>addConfigMapEntryInMap ........................................ 4-35</td>
</tr>
<tr>
<td>4.4.3</td>
<td>addConfigPropertyListEntry .................................... 4-35</td>
</tr>
<tr>
<td>4.4.4</td>
<td>addConfigPropertyMapEntry .................................... 4-36</td>
</tr>
<tr>
<td>4.4.5</td>
<td>addCustomAuthnEngine ........................................... 4-36</td>
</tr>
<tr>
<td>4.4.6</td>
<td>addCustomSPEngine .............................................. 4-37</td>
</tr>
<tr>
<td>4.4.7</td>
<td>addFederationListEntryInMap .................................. 4-37</td>
</tr>
<tr>
<td>4.4.8</td>
<td>addFederationMapEntryInMap .................................. 4-38</td>
</tr>
<tr>
<td>4.4.9</td>
<td>addFederationPropertyListEntry ............................. 4-38</td>
</tr>
<tr>
<td>4.4.10</td>
<td>addFederationPropertyMapEntry .............................. 4-39</td>
</tr>
<tr>
<td>4.4.11</td>
<td>deleteCustomAuthnEngine ....................................... 4-39</td>
</tr>
<tr>
<td>4.4.12</td>
<td>deleteCustomSPEngine ........................................... 4-40</td>
</tr>
<tr>
<td>4.4.13</td>
<td>deleteProviderFederation ....................................... 4-40</td>
</tr>
<tr>
<td>4.4.14</td>
<td>deleteUserFederation ........................................... 4-40</td>
</tr>
<tr>
<td>4.4.15</td>
<td>changeMessageFederation ....................................... 4-41</td>
</tr>
<tr>
<td>4.4.16</td>
<td>changePeerProviderDescription ................................ 4-41</td>
</tr>
<tr>
<td>4.4.17</td>
<td>changeSessionStore ............................................. 4-42</td>
</tr>
<tr>
<td>4.4.18</td>
<td>createConfigPropertyList ...................................... 4-42</td>
</tr>
<tr>
<td>4.4.19</td>
<td>createConfigPropertyListInMap .............................. 4-42</td>
</tr>
<tr>
<td>4.4.20</td>
<td>createConfigPropertyMap ....................................... 4-43</td>
</tr>
<tr>
<td>4.4.21</td>
<td>createConfigPropertyMapInMap .............................. 4-43</td>
</tr>
<tr>
<td>4.4.22</td>
<td>createFederationPropertyList ................................ 4-44</td>
</tr>
<tr>
<td>4.4.23</td>
<td>createFederationPropertyListInMap ........................ 4-44</td>
</tr>
<tr>
<td>4.4.24</td>
<td>createFederationPropertyMap ................................ 4-44</td>
</tr>
<tr>
<td>4.4.25</td>
<td>createFederationPropertyMapInMap ........................ 4-45</td>
</tr>
<tr>
<td>4.4.26</td>
<td>createPeerProviderEntry ...................................... 4-45</td>
</tr>
<tr>
<td>4.4.27</td>
<td>getConfigListValueInMap ...................................... 4-46</td>
</tr>
<tr>
<td>4.4.28</td>
<td>getConfigMapEntryInMap ...................................... 4-46</td>
</tr>
<tr>
<td>4.4.29</td>
<td>getConfigProperty ............................................... 4-47</td>
</tr>
<tr>
<td>4.4.30</td>
<td>getConfigPropertyList .......................................... 4-47</td>
</tr>
<tr>
<td>4.4.31</td>
<td>getConfigPropertyMapEntry ................................... 4-48</td>
</tr>
<tr>
<td>4.4.32</td>
<td>getFederationListValueInMap ................................ 4-48</td>
</tr>
<tr>
<td>4.4.33</td>
<td>getFederationMapEntryInMap ................................ 4-48</td>
</tr>
<tr>
<td>4.4.34</td>
<td>getFederationProperty ........................................... 4-49</td>
</tr>
<tr>
<td>4.4.35</td>
<td>getFederationPropertyList .................................... 4-49</td>
</tr>
<tr>
<td>4.4.36</td>
<td>extractproviderprops .......................................... 4-50</td>
</tr>
<tr>
<td>4.4.37</td>
<td>setproviderprops ................................................ 4-50</td>
</tr>
<tr>
<td>4.4.38</td>
<td>getFederationPropertyMapEntry .............................. 4-51</td>
</tr>
<tr>
<td>4.4.39</td>
<td>listCustomAuthnEngines ...................................... 4-51</td>
</tr>
<tr>
<td>4.4.40</td>
<td>listCustomSPEngines .......................................... 4-51</td>
</tr>
<tr>
<td>4.4.41</td>
<td>loadMetadata ...................................................... 4-52</td>
</tr>
<tr>
<td>4.4.42</td>
<td>oifStatus ............................................................ 4-52</td>
</tr>
<tr>
<td>4.4.43</td>
<td>removeConfigListInMap ........................................ 4-53</td>
</tr>
<tr>
<td>4.4.44</td>
<td>removeConfigMapEntryInMap .................................. 4-53</td>
</tr>
<tr>
<td>4.4.45</td>
<td>removeConfigMapInMap ......................................... 4-54</td>
</tr>
</tbody>
</table>
4.4.46  removeConfigProperty ........................................................... 4-54
4.4.47  removeConfigPropertyList .................................................. 4-55
4.4.48  removeConfigPropertyMap .................................................. 4-55
4.4.49  removeConfigPropertyMapEntry ........................................ 4-56
4.4.50  removeFederationListInMap ................................................ 4-56
4.4.51  removeFederationMapInMap ............................................... 4-56
4.4.52  removeFederationMapEntryInMap ..................................... 4-57
4.4.53  removeFederationProperty ................................................. 4-57
4.4.54  removeFederationPropertyList ......................................... 4-58
4.4.55  removeFederationPropertyMap ......................................... 4-58
4.4.56  removeFederationPropertyMapEntry ................................ 4-59
4.4.57  removePeerProviderEntry .................................................. 4-59
4.4.58  setConfigProperty .......................................................... 4-59
4.4.59  setCustomAuthnEngine ..................................................... 4-60
4.4.60  setCustomSPEngine ........................................................ 4-60
4.4.61  setFederationProperty ....................................................... 4-61
4.5  Directory Integration Platform Commands .............................. 4-61
4.6  Security Commands ............................................................... 4-62
4.6.1  createAppRole ................................................................. 4-63
4.6.2  deleteAppRole ................................................................. 4-64
4.6.3  grantAppRole ................................................................. 4-64
4.6.4  revokeAppRole ............................................................... 4-64
4.6.5  listAppRoles ................................................................. 4-65
4.6.6  listAppRolesMembers ...................................................... 4-65
4.6.7  grantPermission .............................................................. 4-66
4.6.8  revokePermission ............................................................ 4-67
4.6.9  listPermissions ............................................................... 4-67
4.6.10 deleteAppPolicies ......................................................... 4-68
4.6.11 migrateSecurityStore ....................................................... 4-68
4.6.12 updateCred ................................................................. 4-72
4.6.13 createCred ................................................................. 4-72
4.6.14 deleteCred ................................................................. 4-73
4.6.15 modifyBootStrapCredential .......................................... 4-73
4.6.16 addBootStrapCredential .............................................. 4-74
4.6.17 exportEncryptionKey .................................................... 4-75
4.6.18 importEncryptionKey .................................................... 4-75
4.6.19 restoreEncryptionKey ................................................... 4-76
4.6.20 rollOverEncryptionKey .................................................. 4-76
4.6.21 reassociateSecurityStore ............................................... 4-77
4.6.22 upgradeSecurityStore .................................................... 4-77
4.6.23 createResourceType ..................................................... 4-77
4.6.24 getResourceType .......................................................... 4-78
4.6.25 deleteResourceType .................................................... 4-78
4.6.26 listSecurityStoreInfo .................................................... 4-79
4.6.27 listAppStripes ............................................................. 4-79
4.6.28 listCodeSourcePermissions ........................................... 4-80
4.6.29 createResource .......................................................... 4-80
4.7.34 configureOAAM ................................................................. 4-111
4.7.35 registerOIFDAPPartner ..................................................... 4-112
4.7.36 enableCoexistMode .......................................................... 4-112
4.7.37 disableCoexistMode ......................................................... 4-113
4.7.38 editGITOVValues ............................................................ 4-113
4.7.39 editWebgate11gAgent ....................................................... 4-114
4.7.40 deleteWebgate11gAgent ................................................... 4-115
4.7.41 displayWebgate11gAgent .................................................. 4-116
4.7.42 displayOAMMetrics ......................................................... 4-116
4.7.43 updateOIMHostPort ......................................................... 4-117
4.7.44 configureOIM ................................................................. 4-117
4.7.45 updateOSSOResponseCookieConfig .................................. 4-118
4.7.46 deleteOSSOResponseCookieConfig .................................. 4-119
4.7.47 displaySimpleModeGlobalPassphrase .............................. 4-119
4.7.48 exportSelectedPartners ................................................... 4-119
4.7.49 migrateArtifacts ............................................................. 4-120
4.7.50 registerThirdPartyTAPPartner ........................................ 4-120

4.8 Oracle Security Token Service ........................................... 4-121
4.8.1 getPartner ........................................................................ 4-123
4.8.2 getAllRequesterPartners .................................................. 4-124
4.8.3 getAllRelyingPartyPartners .............................................. 4-124
4.8.4 getAllIssuingAuthorityPartners ....................................... 4-124
4.8.5 isPartnerPresent ............................................................ 4-125
4.8.6 createPartner ................................................................. 4-125
4.8.7 updatePartner ................................................................. 4-126
4.8.8 deletePartner ................................................................. 4-126
4.8.9 getPartnerUsernameTokenUsername ................................ 4-127
4.8.10 getPartnerUsernameTokenPassword .............................. 4-127
4.8.11 setPartnerUsernameTokenCredential ............................ 4-128
4.8.12 deletePartnerUsernameTokenCredential ....................... 4-128
4.8.13 getPartnerSigningCert .................................................. 4-129
4.8.14 getPartnerEncryptionCert .............................................. 4-129
4.8.15 setPartnerSigningCert .................................................. 4-130
4.8.16 setPartnerEncryptionCert .............................................. 4-130
4.8.17 deletePartnerSigningCert .............................................. 4-131
4.8.18 deletePartnerEncryptionCert ........................................ 4-131
4.8.19 getPartnerAllIdentityAttributes ................................... 4-131
4.8.20 getPartnerIdentityAttribute .......................................... 4-132
4.8.21 setPartnerIdentityAttribute .......................................... 4-132
4.8.22 deletePartnerIdentityAttribute ....................................... 4-133
4.8.23 getAllWSPrefixAndPartnerMappings .............................. 4-134
4.8.24 getWSPrefixAndPartnerMapping ................................... 4-134
4.8.25 createWSPrefixAndPartnerMapping ............................... 4-134
4.8.26 deleteWSPrefixAndPartnerMapping ............................... 4-135
4.8.27 getAllPartnerProfiles .................................................... 4-135
4.8.28 getPartnerProfile .......................................................... 4-136
4.8.29 createRequesterPartnerProfile ....................................... 4-136
createRelyingPartyPartnerProfile................................................................. 4-138
createIssuingAuthorityPartnerProfile .................................................. 4-139
deletePartnerProfile................................................................................. 4-140
getAllIssuanceTemplates ........................................................................... 4-140
getIssuanceTemplate .................................................................................. 4-141
createIssuanceTemplate ............................................................................. 4-141
deleteIssuanceTemplate ............................................................................. 4-143
getAllValidationTemplates ........................................................................ 4-143
getValidationTemplate ................................................................................ 4-143
createWSSValidationTemplate .................................................................. 4-144
createWSTrustValidationTemplate ............................................................. 4-147
deleteValidationTemplate ........................................................................... 4-149
Oracle Keystore Service ............................................................................ 4-150
changeKeyPassword .................................................................................... 4-151
changeKeyStorePassword ........................................................................... 4-151
createKeyStore ............................................................................................ 4-152
deleteKeyStore ........................................................................................... 4-152
deleteKeyStoreEntry ................................................................................... 4-153
exportKeyStore ............................................................................................ 4-153
exportKeyStoreCertificate .......................................................................... 4-154
exportKeyStoreCertificateRequest ............................................................. 4-155
generateKeyPair .......................................................................................... 4-155
generateSecretKey ..................................................................................... 4-156
getKeyStoreCertificates .............................................................................. 4-157
ggetKeyStoreSecretKeyProperties ............................................................... 4-157
importKeyStore ............................................................................................ 4-158
importKeyStoreCertificate ......................................................................... 4-158
listExpiringCertificates ............................................................................. 4-159
listKeyStoreAliases .................................................................................... 4-160
listKeyStores ............................................................................................... 4-160
Library Oracle Virtual Directory (libOVD) Commands ............................... 4-161
addDNAtribute ............................................................................................ 4-163
activateLibOVDConfigChanges .................................................................. 4-163
addAttributeExclusionRule ....................................................................... 4-164
addAttributeRule ......................................................................................... 4-164
addDomainExclusionRule .......................................................................... 4-165
addDomainRule ............................................................................................ 4-165
addJoinRule .................................................................................................. 4-165
addLDAPHost .............................................................................................. 4-166
addMappingContext .................................................................................... 4-167
addPlugin ..................................................................................................... 4-167
addPluginParam .......................................................................................... 4-168
addToRequestControlExcludeList .............................................................. 4-168
addToRequestControlIncludeList .............................................................. 4-169
assignViewToAdapter .................................................................................. 4-169
createJoinAdapter ....................................................................................... 4-170
createLDAPAdapter .................................................................................... 4-170
4.10.17 createLDAPAdapterWithDefaultPlugins .............................................................. 4-171
4.10.18 createView ...................................................................................................... 4-172
4.10.19 deleteAdapter ................................................................................................ 4-173
4.10.20 deleteAttributeExclusionRule ...................................................................... 4-173
4.10.21 deleteAttributeRule ...................................................................................... 4-174
4.10.22 deleteDomainExclusionRule ......................................................................... 4-174
4.10.23 deleteDomainRule ......................................................................................... 4-174
4.10.24 deleteMappingContext ................................................................................. 4-175
4.10.25 deleteView .................................................................................................... 4-175
4.10.26 getAdapterDetails ......................................................................................... 4-176
4.10.27 listAdapters .................................................................................................. 4-176
4.10.28 listAllMappingContextIds ............................................................................ 4-177
4.10.29 listAttributeRules ......................................................................................... 4-177
4.10.30 listDomainRules ........................................................................................... 4-177
4.10.31 listViews ........................................................................................................ 4-178
4.10.32 modifyLDAPAdapter .................................................................................... 4-178
4.10.33 modifySocketOptions .................................................................................. 4-181
4.10.34 removeAllRequestControlExcludeList ....................................................... 4-181
4.10.35 removeAllRequestControlIncludeList ......................................................... 4-182
4.10.36 removeFromRequestControlExcludeList ..................................................... 4-182
4.10.37 removeDNAttribute ...................................................................................... 4-182
4.10.38 removeFromRequestControlIncludeList ..................................................... 4-183
4.10.39 removeJoinRule ........................................................................................... 4-183
4.10.40 removeLDAPHost ......................................................................................... 4-184
4.10.41 removePlugin ............................................................................................... 4-184
4.10.42 removePluginParam .................................................................................... 4-185
4.10.43 replacePluginParam .................................................................................... 4-185
4.10.44 unassignViewFromAdapter ........................................................................ 4-186
4.10.45 listSSLStoreType .......................................................................................... 4-186
4.10.46 enableKSSForSSL ......................................................................................... 4-187
4.10.47 enableJKSForSSL ........................................................................................ 4-187
4.10.48 createKeyStoreAndEnableJKSForSSL .......................................................... 4-187
4.10.49 importTrustedCertificateIntoSSLStore ...................................................... 4-188
4.10.50 migrateAllTrustedCertificatesFromJKSToKSS .......................................... 4-188
4.10.51 migrateTrustedCertificatesFromJKSToKSS ................................................. 4-189
4.10.52 changeLDAPHostPort ................................................................................ 4-189
4.10.53 removeLDAPHostPort ................................................................................. 4-190
4.10.54 setReadOnlyForLDAPHost ......................................................................... 4-190
4.11 Identity Directory Service Commands ............................................................... 4-191
4.11.1 activateDSConfigChanges .......................................................................... 4-192
4.11.2 addAttributeInEntityConfig ........................................................................ 4-193
4.11.3 addAttributePropsInEntityConfig ................................................................. 4-193
4.11.4 addAttributeRefForEntity ............................................................................ 4-194
4.11.5 addAttrrefPropsInEntityConfig .................................................................... 4-195
4.11.6 addCommonPropertyForOperationConfig ............................................... 4-195
4.11.7 addEntity ...................................................................................................... 4-196
4.11.8 addEntityProps ............................................................................................. 4-197
5 User Messaging Service (UMS) Custom WLST Commands

5.1 UMS WLST Command Group .......................................................... 5-1
5.1.1 manageUserMessagingPrefs ......................................................... 5-1
5.1.2 deployUserMessagingDriver .......................................................... 5-3

6 DMS Custom WLST Commands

6.1 DMS Metric Commands ................................................................. 6-1
6.1.1 displayMetricTableNames ............................................................ 6-2
6.1.2 displayMetricTables ................................................................. 6-3
6.1.3 dumpMetrics ............................................................................. 6-6
6.1.4 reloadMetricRules ................................................................. 6-8
6.2 DMS Event Tracing Commands .................................................... 6-9
6.2.1 addDMSEventDestination ......................................................... 6-10
6.2.2 addDMSEventFilter ................................................................. 6-11
6.2.3 addDMSEventRoute ................................................................. 6-14
6.2.4 enableDMSEventTrace ............................................................. 6-15
6.2.5 listDMSEventConfiguration ..................................................... 6-16
6.2.6 listDMSEventDestination ......................................................... 6-16
6.2.7 listDMSEventFilter ................................................................. 6-17
6.2.8 listDMSEventRoutes ............................................................... 6-18
### 7 Logging Custom WLST Commands

<table>
<thead>
<tr>
<th>Section</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1 Log Configuration Commands</td>
<td>configureLogHandler, getLogLevel, listLoggers, listLogHandlers, setLogLevel, startTracing, configureTracingLoggers, listActiveTraces, listTracingLoggers, stopTracing</td>
</tr>
<tr>
<td>7.2 Search and Display Commands</td>
<td>displayLogs, listLogs</td>
</tr>
<tr>
<td>7.3 Selective Tracing Commands</td>
<td>updateDMSEventDestination, updateDMSEventRoute</td>
</tr>
</tbody>
</table>

### 8 Metadata Services (MDS) Custom WLST Commands

<table>
<thead>
<tr>
<th>Section</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1 Repository Management Commands</td>
<td>createMetadataPartition, deleteMetadataPartition, deregisterMetadataDBRepository, registerMetadataDBRepository</td>
</tr>
<tr>
<td>8.2 Application Metadata Management Commands</td>
<td>deleteMetadata, exportMetadata, importMetadata, purgeMetadata</td>
</tr>
<tr>
<td>8.3 Sandbox Metadata Management Commands</td>
<td>exportSandboxMetadata, importSandboxMetadata</td>
</tr>
<tr>
<td>8.4 Application Label Management Commands</td>
<td>createMetadataLabel, deleteMetadataLabel, listMetadataLabels, promoteMetadataLabel, purgeMetadataLabels</td>
</tr>
<tr>
<td>8.5 Application Management Deployment Commands</td>
<td>getMDSArchiveConfig, importMAR</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>8.6</td>
<td>Multitenancy Management Commands</td>
</tr>
<tr>
<td>8.6.1</td>
<td>deprovisionTenant</td>
</tr>
<tr>
<td>8.6.2</td>
<td>listTenants</td>
</tr>
<tr>
<td>9</td>
<td>Oracle SOA Suite Custom WLST Commands</td>
</tr>
<tr>
<td>9.1</td>
<td>Overview of WSLT Command Categories</td>
</tr>
<tr>
<td>9.2</td>
<td>Deployment Commands</td>
</tr>
<tr>
<td>9.2.1</td>
<td>sca_deployComposite</td>
</tr>
<tr>
<td>9.2.2</td>
<td>sca_undeployComposite</td>
</tr>
<tr>
<td>9.3</td>
<td>SOA Composite Application Management Commands</td>
</tr>
<tr>
<td>9.3.1</td>
<td>sca_startComposite</td>
</tr>
<tr>
<td>9.3.2</td>
<td>sca_stopComposite</td>
</tr>
<tr>
<td>9.3.3</td>
<td>sca_activateComposite</td>
</tr>
<tr>
<td>9.3.4</td>
<td>sca_retireComposite</td>
</tr>
<tr>
<td>9.3.5</td>
<td>sca_assignDefaultComposite</td>
</tr>
<tr>
<td>9.3.6</td>
<td>sca_getDefaultCompositeRevision</td>
</tr>
<tr>
<td>9.3.7</td>
<td>sca_listDeployedComposites</td>
</tr>
<tr>
<td>9.3.8</td>
<td>sca_listCompositesFailedInServerStart</td>
</tr>
<tr>
<td>9.4</td>
<td>SOA Composite Application Compilation Commands</td>
</tr>
<tr>
<td>9.4.1</td>
<td>sca_setProp</td>
</tr>
<tr>
<td>9.4.2</td>
<td>sca_compile</td>
</tr>
<tr>
<td>9.5</td>
<td>SOA Composite Application Packaging Commands</td>
</tr>
<tr>
<td>9.5.1</td>
<td>sca_package</td>
</tr>
<tr>
<td>9.6</td>
<td>SOA Composite Application Test Commands</td>
</tr>
<tr>
<td>9.6.1</td>
<td>sca_test</td>
</tr>
<tr>
<td>9.7</td>
<td>SOA Composite Application HTTP Client-Based Export and Import Commands</td>
</tr>
<tr>
<td>9.7.1</td>
<td>sca_exportComposite</td>
</tr>
<tr>
<td>9.7.2</td>
<td>sca_exportUpdates</td>
</tr>
<tr>
<td>9.7.3</td>
<td>sca_importUpdates</td>
</tr>
<tr>
<td>9.7.4</td>
<td>sca_exportSharedData</td>
</tr>
<tr>
<td>9.7.5</td>
<td>sca_removeSharedData</td>
</tr>
<tr>
<td>9.8</td>
<td>SOA Composite Application MBean-Based Export and Import Commands</td>
</tr>
<tr>
<td>9.8.1</td>
<td>sca_exportCompositeMb</td>
</tr>
<tr>
<td>9.8.2</td>
<td>sca_exportUpdatesMb</td>
</tr>
<tr>
<td>9.8.3</td>
<td>sca_importUpdatesMb</td>
</tr>
<tr>
<td>9.8.4</td>
<td>sca_exportSharedDataMb</td>
</tr>
<tr>
<td>9.9</td>
<td>SOA Composite Application Partition Management Commands</td>
</tr>
<tr>
<td>9.9.1</td>
<td>sca_createPartition</td>
</tr>
<tr>
<td>9.9.2</td>
<td>sca_deletePartition</td>
</tr>
<tr>
<td>9.9.3</td>
<td>sca_startCompositesInPartition</td>
</tr>
<tr>
<td>9.9.4</td>
<td>sca_stopCompositesInPartition</td>
</tr>
<tr>
<td>9.9.5</td>
<td>sca_activateCompositesInPartition</td>
</tr>
<tr>
<td>9.9.6</td>
<td>sca_retireCompositesInPartition</td>
</tr>
<tr>
<td>9.9.7</td>
<td>sca_listPartitions</td>
</tr>
<tr>
<td>9.9.8</td>
<td>sca_listCompositesInPartition</td>
</tr>
<tr>
<td>9.10</td>
<td>SOA Composite Application Preconfigured WLDF Watches Command</td>
</tr>
<tr>
<td>9.10.1</td>
<td>sca_createWatches</td>
</tr>
</tbody>
</table>
9.11 Configuration Plan Management Commands ................................................................. 9-28
9.11.1 sca_attachPlan ........................................................................................................... 9-29
9.11.2 sca_extractPlan ....................................................................................................... 9-29
9.11.3 sca_generatePlan .................................................................................................. 9-30
9.11.4 sca_validatePlan ..................................................................................................... 9-31
9.12 Task Validation Commands .......................................................................................... 9-32
9.12.1 sca_validateTask .................................................................................................... 9-32
9.13 SOA Health Check Commands ...................................................................................... 9-32
9.13.1 executeHealthCheckAndGenerateReport ................................................................ 9-34
9.13.2 executeHealthCheckCategoryAndGenerateReport ................................................... 9-34
9.13.3 executeHealthChecksAndGenerateReport .............................................................. 9-35
9.13.4 getUnsuccessfulHCRetultIds .................................................................................. 9-35
9.13.5 getUnsuccessfulHCRetultReport ............................................................................ 9-35
9.13.6 listHealthChecks .................................................................................................... 9-36
9.13.7 listHealthChecksInCategory ................................................................................... 9-36
9.13.8 listHealthCheckCategories .................................................................................... 9-37
9.13.9 deleteAllHCRetultResults ...................................................................................... 9-37
9.13.10 deleteHCRetultResults .......................................................................................... 9-38
9.13.11 getUnsuccessfulHCRetultReport ........................................................................... 9-38
9.13.12 getHCParameter .................................................................................................. 9-39
9.13.13 setHCParameter .................................................................................................. 9-39
9.13.14 resetHCParameter ............................................................................................... 9-40
9.13.15 getHCParameters ................................................................................................. 9-40
9.13.16 enableHCStartupRun ............................................................................................ 9-41
9.13.17 setHCStartupRunCategory .................................................................................... 9-41
9.13.18 getHCStartupRun .................................................................................................. 9-42
9.13.19 createHCPeriodicRun ............................................................................................ 9-42
9.13.20 enableHCPeriodicRun .......................................................................................... 9-43
9.13.21 setHCPeriodicRunInterval .................................................................................... 9-43
9.13.22 setHCPeriodicRunSchedule .................................................................................. 9-43
9.13.23 setHCPeriodicRunCategory .................................................................................. 9-44
9.13.24 getHCPeriodicRun ................................................................................................ 9-44
9.13.25 getHCPeriodicRuns .............................................................................................. 9-45
9.13.26 removeHCPeriodicRun ........................................................................................ 9-46
9.13.27 getDisabledHealthChecks .................................................................................... 9-46
9.13.28 addDisabledHealthCheck ...................................................................................... 9-46
9.13.29 removeDisabledHealthCheck ................................................................................ 9-47

10 WebCenter Portal Custom WLST Commands

10.1 Overview of Oracle WebCenter Portal WLST Command Categories .......................... 10-2
10.2 General .......................................................................................................................... 10-3
10.2.1 deleteConnection .................................................................................................... 10-3
10.2.2 setWebCenterServiceFrameworkConfig .............................................................. 10-4
10.2.3 getWebCenterServiceFrameworkConfig .............................................................. 10-5
10.2.4 webcenterErrorOccured ......................................................................................... 10-6
10.2.5 getWebCenterConnectionTypes .............................................................................. 10-6
10.3 Analytics ....................................................................................................................... 10-7
10.6.9 listJCRFileSystemConnections ................................................................. 10-54
10.6.10 createJCRSharePointConnection .............................................................. 10-55
10.6.11 setJCRSharePointConnection ................................................................. 10-57
10.6.12 listJCRSharePointConnections ............................................................... 10-59
10.6.13 listDocumentsSpacesProperties .............................................................. 10-59
10.6.14 setDocumentsSpacesProperties .............................................................. 10-60
10.6.15 deleteDocumentsSpacesProperties .......................................................... 10-62
10.6.16 exportFoldersGData ................................................................................ 10-62
10.6.17 migrateFoldersGDataToFrameworkFolders .............................................. 10-63
10.7 Discussions and Announcements .................................................................. 10-65
10.7.1 createDiscussionForumConnection .......................................................... 10-66
10.7.2 setDiscussionForumConnection ............................................................... 10-69
10.7.3 getDiscussionForumConnectionProperty ................................................ 10-71
10.7.4 deleteDiscussionForumConnectionProperty .......................................... 10-72
10.7.5 listDiscussionForumConnections ............................................................. 10-73
10.7.6 listDefaultDiscussionForumConnection ................................................. 10-74
10.7.7 setDefaultDiscussionForumConnection ................................................. 10-75
10.7.8 setDiscussionForumServiceProperty ..................................................... 10-76
10.7.9 removeDiscussionForumServiceProperty .............................................. 10-77
10.7.10 listDiscussionForumServiceProperties .............................................. 10-78
10.7.11 setAnnouncementServiceProperty ........................................................ 10-79
10.7.12 removeAnnouncementServiceProperty ................................................. 10-79
10.7.13 listAnnouncementServiceProperties ..................................................... 10-80
10.7.14 addDiscussionsServerAdmin ................................................................. 10-81
10.7.15 addDiscussionsCategoryAdmin .............................................................. 10-82
10.7.16 syncDiscussionServerPermissions ....................................................... 10-82
10.7.17 setDiscussionsServerProperty .............................................................. 10-83
10.7.18 getDiscussionsServerProperty .............................................................. 10-84
10.7.19 removeDiscussionsServerProperty ...................................................... 10-85
10.8 External Applications .................................................................................. 10-86
10.8.1 createExtAppConnection ......................................................................... 10-86
10.8.2 setExtAppConnection ............................................................................... 10-87
10.8.3 listExtAppConnections ............................................................................ 10-89
10.8.4 addExtAppField ...................................................................................... 10-90
10.8.5 setExtAppField ....................................................................................... 10-91
10.8.6 removeExtAppField ................................................................................ 10-92
10.8.7 addExtAppCredential ............................................................................ 10-93
10.8.8 setExtAppCredential ............................................................................... 10-94
10.8.9 removeExtAppCredential ...................................................................... 10-95
10.9 Instant Messaging and Presence .................................................................. 10-96
10.9.1 createIMPConnection ............................................................................. 10-97
10.9.2 setIMPConnection .................................................................................. 10-99
10.9.3 setIMPConnectionProperty .................................................................... 10-101
10.9.4 deleteIMPConnectionProperty .............................................................. 10-102
10.9.5 listIMPAdapters ..................................................................................... 10-103
10.9.6 listIMPCollections .................................................................................. 10-104
10.9.7 listDefaultIMPCollection ....................................................................... 10-105
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.9.8</td>
<td>setDefaultIMPConnection .................................................. 10-105</td>
</tr>
<tr>
<td>10.9.9</td>
<td>setIMPServiceProperty ....................................................... 10-106</td>
</tr>
<tr>
<td>10.9.10</td>
<td>removeIMPServiceProperty .................................................. 10-108</td>
</tr>
<tr>
<td>10.9.11</td>
<td>listIMPServiceProperties ................................................... 10-108</td>
</tr>
<tr>
<td>10.9.12</td>
<td>createIMPExtAppConnection ................................................ 10-109</td>
</tr>
<tr>
<td>10.10</td>
<td>Identity Store ........................................................................ 10-110</td>
</tr>
<tr>
<td>10.10.1</td>
<td>setWebCenterIdStoreSearchConfig ........................................ 10-110</td>
</tr>
<tr>
<td>10.10.2</td>
<td>listWebCenterIdStoreSearchConfig ........................................ 10-111</td>
</tr>
<tr>
<td>10.11</td>
<td>Lifecycle .............................................................................. 10-112</td>
</tr>
<tr>
<td>10.11.1</td>
<td>deployWebCenterPortal ......................................................... 10-114</td>
</tr>
<tr>
<td>10.11.2</td>
<td>propagateWebCenterPortal ..................................................... 10-117</td>
</tr>
<tr>
<td>10.11.3</td>
<td>exportWebCenterPortals ....................................................... 10-119</td>
</tr>
<tr>
<td>10.11.4</td>
<td>exportWebCenterPortalTemplates ........................................... 10-121</td>
</tr>
<tr>
<td>10.11.5</td>
<td>importWebCenterPortals ....................................................... 10-122</td>
</tr>
<tr>
<td>10.11.6</td>
<td>listWebCenterPortalArchive ................................................ 10-126</td>
</tr>
<tr>
<td>10.11.7</td>
<td>exportWebCenterPortalChanges ............................................. 10-127</td>
</tr>
<tr>
<td>10.11.8</td>
<td>importWebCenterPortalChanges ............................................. 10-128</td>
</tr>
<tr>
<td>10.11.9</td>
<td>exportWebCenterPortalConnections ....................................... 10-129</td>
</tr>
<tr>
<td>10.11.10</td>
<td>importWebCenterPortalConnections ....................................... 10-132</td>
</tr>
<tr>
<td>10.11.11</td>
<td>setSpaceState ................................................................. 10-134</td>
</tr>
<tr>
<td>10.11.12</td>
<td>exportWebCenterResource .................................................... 10-135</td>
</tr>
<tr>
<td>10.11.13</td>
<td>importWebCenterResource .................................................... 10-137</td>
</tr>
<tr>
<td>10.11.14</td>
<td>importWebCenterTranslations ............................................... 10-139</td>
</tr>
<tr>
<td>10.11.15</td>
<td>exportWebCenterApplication ............................................... 10-140</td>
</tr>
<tr>
<td>10.11.16</td>
<td>importWebCenterApplication ............................................... 10-141</td>
</tr>
<tr>
<td>10.11.17</td>
<td>exportPortletClientMetadata ............................................... 10-142</td>
</tr>
<tr>
<td>10.11.18</td>
<td>importPortletClientMetadata ............................................... 10-143</td>
</tr>
<tr>
<td>10.11.19</td>
<td>showProducerImportFailures ............................................... 10-143</td>
</tr>
<tr>
<td>10.11.20</td>
<td>retryAllFailedProducerImports ............................................ 10-144</td>
</tr>
<tr>
<td>10.11.21</td>
<td>cloneWebCenterManagedServer ............................................. 10-145</td>
</tr>
<tr>
<td>10.11.22</td>
<td>exportGroupSpaces (Deprecated) .......................................... 10-146</td>
</tr>
<tr>
<td>10.11.23</td>
<td>exportGroupSpaceTemplates (Deprecated) ............................. 10-147</td>
</tr>
<tr>
<td>10.12</td>
<td>Mail ...................................................................................... 10-148</td>
</tr>
<tr>
<td>10.12.1</td>
<td>createMailConnection ......................................................... 10-151</td>
</tr>
<tr>
<td>10.12.2</td>
<td>setMailConnection ............................................................. 10-153</td>
</tr>
<tr>
<td>10.12.3</td>
<td>setMailConnectionProperty ............................................... 10-156</td>
</tr>
<tr>
<td>10.12.4</td>
<td>deleteMailConnectionProperty ............................................. 10-157</td>
</tr>
<tr>
<td>10.12.5</td>
<td>listMailConnections .......................................................... 10-158</td>
</tr>
<tr>
<td>10.12.6</td>
<td>listDefaultMailConnection .................................................. 10-159</td>
</tr>
<tr>
<td>10.12.7</td>
<td>setDefaultMailConnection ................................................... 10-160</td>
</tr>
<tr>
<td>10.12.8</td>
<td>setMailServiceProperty ....................................................... 10-160</td>
</tr>
<tr>
<td>10.12.9</td>
<td>removeMailServiceProperty ................................................ 10-162</td>
</tr>
<tr>
<td>10.12.10</td>
<td>listMailServiceProperties ................................................... 10-163</td>
</tr>
<tr>
<td>10.12.11</td>
<td>createMailExtApp ............................................................... 10-163</td>
</tr>
<tr>
<td>10.13</td>
<td>Migration ............................................................................. 10-164</td>
</tr>
<tr>
<td>10.13.1</td>
<td>migrateFrameworkApplicationConnections .............................. 10-164</td>
</tr>
</tbody>
</table>
10.13.2 migrateFrameworkApplicationToWebCenter ................................................. 10-166
10.13.3 migrateFrameworkApplicationDocumentsToWebCenter ............................ 10-168
10.13.4 migrateFrameworkApplicationDiscussionsToWebCenter ............................ 10-169
10.14 Notifications.................................................................................................... 10-171
10.14.1 setNotificationsConfig ................................................................................. 10-171
10.14.2 getNotificationsConfig ............................................................................... 10-172
10.15 People Connections ...................................................................................... 10-173
10.15.1 startSyncProfiles .......................................................................................... 10-174
10.15.2 stopSyncProfiles .......................................................................................... 10-174
10.15.3 isSyncProfilesRunning .................................................................................. 10-175
10.15.4 syncProfile .................................................................................................. 10-176
10.15.5 setProfileConfig ........................................................................................... 10-176
10.15.6 getProfileConfig ........................................................................................... 10-177
10.15.7 listProfileConfig ........................................................................................... 10-178
10.15.8 setProfileCacheNumberOfObjects ................................................................. 10-179
10.15.9 setProfileSyncLDAPReadBatchSize ............................................................... 10-180
10.15.10 setProfileCacheTimeToLive ....................................................................... 10-180
10.15.11 setProfilePhotoSync .................................................................................... 10-181
10.15.12 printProfileConfig ...................................................................................... 10-182
10.16 Personal Events .............................................................................................. 10-182
10.16.1 createPersonalEventConnection ................................................................. 10-183
10.16.2 setPersonalEventConnection ...................................................................... 10-184
10.16.3 listPersonalEventConnections .................................................................... 10-185
10.17 Personalization .............................................................................................. 10-186
10.17.1 createWCPSCMISConnection ..................................................................... 10-187
10.17.2 createWCPSActivityGraphConnection ......................................................... 10-189
10.17.3 createWCPSPeopleConnection .................................................................... 10-190
10.17.4 createWCPSCustomConnection ................................................................... 10-191
10.17.5 listWCPSCMISConnection .......................................................................... 10-191
10.17.6 listWCPSActivityGraphConnection ............................................................. 10-192
10.17.7 listWCPSPeopleConnection ....................................................................... 10-193
10.17.8 listWCPSCustomConnection ...................................................................... 10-194
10.17.9 deleteWCPSCMISConnection ..................................................................... 10-195
10.17.10 deleteWCPSActivityGraphConnection ....................................................... 10-196
10.17.11 deleteWCPSPeopleConnection .................................................................. 10-196
10.17.12 deleteWCPSCustomConnection ................................................................ 10-197
10.17.13 setWCPSConnectionProperty .................................................................... 10-198
10.18 Portlet Producers .......................................................................................... 10-199
10.18.1 registerWSRPProducer .............................................................................. 10-200
10.18.2 setWSRPProducer ...................................................................................... 10-204
10.18.3 listWSRPProducers .................................................................................... 10-208
10.18.4 deregisterWSRPProducer .......................................................................... 10-209
10.18.5 listWSRPProducerRegistrationProperties .................................................. 10-210
10.18.6 listWSRPProducerUserCategories ............................................................. 10-211
10.18.7 mapWSRPProducerUserCategory ............................................................... 10-212
10.18.8 registerPDKJavaProducer ......................................................................... 10-213
10.18.9 setPDKJavaProducer .................................................................................. 10-215
11 Application Development Framework (ADF) Custom WLST Commands

11.1 Overview of WLST Command Categories ................................................................. 11-1
11.2 ADF-Specific WLST Commands ................................................................................ 11-1
11.2.1 adf_createFileUrlConnection .............................................................................. 11-2
11.2.2 adf_createHttpUrlConnection ............................................................................ 11-2
11.2.3 adf_setUrlConnectionAttributes ....................................................................... 11-3
11.2.4 adf_listUrlConnection ....................................................................................... 11-3
11.2.5 getADFMArchiveConfig ..................................................................................... 11-4
11.2.6 exportJarVersions ............................................................................................... 11-6
11.2.7 exportApplicationJarVersions ........................................................................... 11-6
11.2.8 exportApplicationSelectedJar Versions .............................................................. 11-7
11.2.9 createWebServiceConnection ............................................................................. 11-8
11.2.10 listWebServiceConnection ................................................................................ 11-8
11.2.11 deleteWebServiceConnection ........................................................................... 11-9
11.2.12 listUpgradeHandlers ....................................................................................... 11-9
11.2.13 upgradeADFMetadataApp ............................................................................... 11-10
11.2.14 upgradeADFMetadataAppHandlers ................................................................... 11-10
11.2.15 upgradeADFMetadata ..................................................................................... 11-11
11.2.16 upgradeADFMetadataHandlers ........................................................................ 11-11

12 Portal Custom WLST Commands

12.1 Database Access Descriptor Commands ................................................................ 12-1
12.1.1 listDads ............................................................................................................. 12-2
12.1.2 createPortalDad .............................................................................................. 12-2
12.1.3 updatePortalDad .............................................................................................. 12-3
14.3.16 setWebServicePolicyOverride .......................................................... 14-42
14.4 Policy Set Management Commands ..................................................... 14-43
14.4.1 abortRepositorySession ................................................................. 14-45
14.4.2 attachPolicySet ........................................................ ...................... 14-45
14.4.3 attachPolicySetPolicy ................................................................. 14-46
14.4.4 beginRepositorySession .............................................................. 14-47
14.4.5 clonePolicySet .............................................................................. 14-47
14.4.6 commitRepositorySession .............................................................. 14-48
14.4.7 createPolicySet ............................................................................. 14-49
14.4.8 deleteAllPolicySets ...................................................................... 14-50
14.4.9 deletePolicySet ............................................................................. 14-51
14.4.10 describeRepositorySession ......................................................... 14-51
14.4.11 detachPolicySetPolicy ............................................................... 14-52
14.4.12 displayPolicySet ........................................................................ 14-52
14.4.13 enablePolicySet ........................................................................ 14-53
14.4.14 enablePolicySetPolicy .............................................................. 14-54
14.4.15 listPolicySets ............................................................................ 14-54
14.4.16 migrateAttachments ................................................................. 14-55
14.4.17 modifyPolicySet ....................................................................... 14-56
14.4.18 setPolicySetConstraint ............................................................. 14-57
14.4.19 setPolicySetDescription ............................................................ 14-58
14.4.20 setPolicySetPolicyOverride ...................................................... 14-58
14.4.21 validatePolicySet ..................................................................... 14-59
14.5 Oracle WSM Repository Management Commands .............................. 14-60
14.5.1 upgradeWSMPolicyRepository .................................................... 14-60
14.5.2 resetWSMPolicyRepository ......................................................... 14-61
14.5.3 exportRepository ........................................................................ 14-62
14.5.4 importRepository ...................................................................... 14-63
14.6 Deployment Descriptor Migration Commands ...................................... 14-64
14.6.1 exportJRFWSApplicationPDD .................................................... 14-65
14.6.2 importJRFWSApplicationPDD .................................................... 14-65
14.6.3 savePddToAllAppInstancesInDomain ....................................... 14-66
14.7 Token Issuer Trust Configuration Commands ..................................... 14-67
14.7.1 deleteWSMTokenIssuerTrust ..................................................... 14-68
14.7.2 deleteWSMTokenIssuerTrustAttributeRule ............................... 14-69
14.7.3 displayWSMTokenIssuerTrust .................................................... 14-69
14.7.4 setWSMTokenIssuerTrust .......................................................... 14-70
14.7.5 setWSMTokenIssuerTrustAttributeFilter .................................. 14-71
14.7.6 setWSMTokenIssuerTrustAttributeMapping .............................. 14-72
14.7.7 revokeWSMTokenIssuerTrust ..................................................... 14-74
14.7.8 exportWSMTokenIssuerTrustMetadata ..................................... 14-74
14.7.9 importWSMTokenIssuerTrustMetadata ..................................... 14-75
14.8 Diagnostic Commands .................................................................... 14-76
14.8.1 checkWSMStatus .................................................................... 14-76
14.9 JKS Keystore Configuration Commands ........................................... 14-79
14.9.1 deleteWSMKeyStoreEntry ......................................................... 14-80
14.9.2 deleteWSMKeyStoreEntries ....................................................... 14-81
14.9.3 exportWSMCertificate ...................................................... 14-81
14.9.4 importWSMCertificate .................................................. 14-82
14.9.5 listWSMKeystoreAliases ................................................ 14-83
14.9.6 displayWSMCertificate .................................................. 14-84
14.10 OWSM Configuration Commands ........................................ 14-85
14.10.1 displayConfiguration .................................................... 14-85
14.10.2 setConfiguration ......................................................... 14-86
14.11 Upgrade OWSM Configuration Command ......................... 14-87
14.11.1 upgradeWSM ............................................................... 14-87

15 Diagnostic Framework Custom WLST Commands

15.1 Incident Commands .......................................................... 15-1
15.1.1 createIncident .............................................................. 15-2
15.1.2 getIncidentFile ........................................................... 15-3
15.1.3 listADRHomes ............................................................ 15-3
15.1.4 listIncidents ............................................................... 15-4
15.1.5 listProblems .............................................................. 15-4
15.1.6 reloadCustomRules .................................................... 15-5
15.1.7 showIncident .............................................................. 15-5
15.2 Diagnostic Dump Commands .............................................. 15-6
15.2.1 describeDump ............................................................ 15-6
15.2.2 executeDump ............................................................. 15-7
15.2.3 listDumps ................................................................. 15-8
15.3 Dump Sampling Commands .............................................. 15-9
15.3.1 addDumpSample ....................................................... 15-9
15.3.2 enableDumpSampling ................................................ 15-10
15.3.3 getSamplingArchives ................................................ 15-11
15.3.4 isDumpSamplingEnabled ........................................... 15-11
15.3.5 listDumpSamples ...................................................... 15-12
15.3.6 removeDumpSample .................................................. 15-13
15.3.7 updateDumpSample .................................................. 15-13

16 Information Rights Management Custom WLST Commands

16.1 Overview of WLST IRM Commands .................................... 16-1
16.2 General Server Commands ............................................... 16-2
16.2.1 addIRMRefreshPeriod ............................................... 16-2
16.2.2 getIRMRefreshPeriod ............................................... 16-3
16.2.3 removeIRMRefreshPeriod ......................................... 16-3
16.2.4 updateIRMRefreshPeriod ......................................... 16-4
16.2.5 addIRMSyncWindow .................................................. 16-4
16.2.6 getIRMSyncWindow ................................................... 16-5
16.2.7 getIRMSyncWindows .................................................. 16-5
16.2.8 removeIRMSyncWindow ............................................ 16-5
16.2.9 updateIRMSyncWindow ............................................. 16-6
16.2.10 getIRMCryptoSchema ............................................... 16-7
16.2.12 setIRMCryptoSchema ...................................................................................................... 16-7
16.2.13 getIRMDeviceCount ...................................................................................................... 16-7
16.2.14 setIRMDeviceCount ...................................................................................................... 16-8
16.2.15 getIRMJournalCleanUp ............................................................................................... 16-8
16.2.16 setIRMJournalCleanUp ............................................................................................... 16-8
16.2.17 getIRMLicenseStateCleanUp .......................................................................................... 16-9
16.2.18 setIRMLicenseStateCleanUp .......................................................................................... 16-9
16.2.19 getIRMPrivacyURL ...................................................................................................... 16-10
16.2.20 setIRMPrivacyURL ...................................................................................................... 16-10
16.2.21 getIRMKeyStore ......................................................................................................... 16-11
16.2.22 setIRMKeyStore ......................................................................................................... 16-11
16.3 Migration Commands ....................................................................................................... 16-11
16.3.1 setIRMExportFolder ..................................................................................................... 16-12
16.3.2 getIRMExportFolder ..................................................................................................... 16-12
16.3.3 setIRMImportFolder ..................................................................................................... 16-12
16.3.4 getIRMImportFolder ..................................................................................................... 16-13
16.4 Test Content Commands ............................................................................................... 16-13
16.4.1 addIRMTestContent ................................................................................................. 16-13
16.4.2 getIRMTestContent ................................................................................................. 16-14
16.4.3 getIRMTestContents ................................................................................................. 16-15
16.4.4 removeIRMTestContent ............................................................................................ 16-15
16.4.5 updateIRMTestContent ............................................................................................ 16-16
16.5 Languages Support Commands ..................................................................................... 16-16
16.5.1 addIRMTranslation ................................................................................................. 16-17
16.5.2 getIRMDefaultTranslation ....................................................................................... 16-17
16.5.3 getIRMTranslations ................................................................................................. 16-18
16.5.4 removeIRMTranslation ............................................................................................ 16-18
16.5.5 setIRMTranslations ................................................................................................. 16-18
16.6 Oracle IRM Desktop Installers Commands ..................................................................... 16-19
16.6.1 addIRMDownload ..................................................................................................... 16-19
16.6.2 getIRMDownload ..................................................................................................... 16-20
16.6.3 getIRMDownloads .................................................................................................... 16-20
16.6.4 removeIRMDownload ............................................................................................... 16-21
16.6.5 updateIRMDownload ............................................................................................... 16-21

17 Oracle WebCenter: Imaging Custom WLST Commands

17.1 Overview of Imaging WLST Command Categories .......................................................... 17-1
17.2 Diagnostic Commands ...................................................................................................... 17-1
17.2.1 clearIPMWorkflowFaults ............................................................................................ 17-2
17.2.2 listIPMWorkflowFaults ............................................................................................... 17-2
17.2.3 repairIPMWorkflowFaults ......................................................................................... 17-3
17.2.4 sumIPMWorkflowFaults ............................................................................................. 17-3
17.2.5 resetIpmDMSMetrics ................................................................................................. 17-4
17.3 Imaging Configuration Commands .................................................................................. 17-4
17.3.1 createIPMConnection ............................................................................................... 17-5
17.3.2 getIPMConfig ........................................................................................................... 17-6
17.3.3 grantIPMCredAccess ............................................................................................... 17-6
18 Oracle Business Process Management Custom WLST Commands
18.1 BPMLifecycleAdmin Command Group ................................................................. 18-1
18.1.1 create_public_share .......................................................................................... 18-2
18.1.2 unlock_public_share ........................................................................................ 18-2
18.1.3 export_public_share ....................................................................................... 18-3
18.1.4 delete_public_share ...................................................................................... 18-4
18.1.5 publish_template ............................................................................................ 18-5
18.1.6 export_template .............................................................................................. 18-5
18.1.7 delete_template ............................................................................................... 18-6
18.1.8 connectProjectRepository ............................................................................. 18-7
18.1.9 disconnectProjectRepository ......................................................................... 18-7
18.1.10 importProject ............................................................................................... 18-8
18.1.11 createProject ............................................................................................... 18-8
18.1.12 deleteProject ............................................................................................... 18-9
18.1.13 exportProject ............................................................................................... 18-10
18.1.14 unlockProject .............................................................................................. 18-10

19 Oracle WebCenter Content Custom WLST Commands
19.1 Overview of WLST WebCenter Content Command Categories ...................... 19-1
19.2 WLST WebCenter Content Help ....................................................................... 19-2
19.3 Getter and Setter Methods Implementation ...................................................... 19-2
19.4 Server Configuration Commands ..................................................................... 19-3
19.4.1 getUCMHttpServerAddress .......................................................................... 19-3
19.4.2 getUCMServerPort ...................................................................................... 19-4
19.4.3 setUCMServerPort ...................................................................................... 19-4
19.4.4 getUCMIpAddressFilter ............................................................................. 19-4
19.4.5 setUCMIpAddressFilter ............................................................................. 19-5
19.4.6 getUCMUseSSL ......................................................................................... 19-5
19.5 Email Configuration Commands ..................................................................... 19-6
19.5.1 getUCMMailServer ..................................................................................... 19-6
19.5.2 setUCMMailServer ..................................................................................... 19-6
19.5.3 getUCMSmtpPort ....................................................................................... 19-7
19.5.4 getUCMSysAdminAddress .......................................................................... 19-7
19.5.5 setUCMSysAdminAddress ........................................................................ 19-8
19.6 System Status Commands ............................................................................... 19-8
19.6.1 getUCMCSVersion ..................................................................................... 19-8
19.6.2 getUCMServerUptime .............................................................................. 19-9
19.7 General Configuration Commands ................................................................. 19-9
  19.7.1 getUCMOverRideFormat ............................................................................ 19-10
  19.7.2 setUCMOverRideFormat .......................................................................... 19-10
  19.7.3 getUCMDownloadApplet ........................................................................... 19-11
  19.7.4 setUCMDownloadApplet .......................................................................... 19-11
  19.7.5 getUCMMultiUpload ............................................................................... 19-12
  19.7.6 setUCMMultiUpload ............................................................................... 19-12
  19.7.7 getUCMUseAccounts ............................................................................. 19-12
  19.7.8 setUCMUseAccounts ............................................................................. 19-13
  19.7.9 getUCMIsAutoNumber ........................................................................... 19-13
  19.7.10 setUCMIsAutoNumber .......................................................................... 19-14
  19.7.11 getUCMAutoNumberPrefix ................................................................... 19-14
  19.7.12 setUCMAutoNumberPrefix ................................................................... 19-15
  19.7.13 getUCMMajorRevLabelSeq .................................................................. 19-15
  19.7.14 setUCMMajorRevLabelSeq .................................................................. 19-15
  19.7.15 getUCMMinorRevLabelSeq .................................................................. 19-16
  19.7.16 setUCMMinorRevLabelSeq .................................................................. 19-16
  19.7.17 getUCMjspServerEnabled ..................................................................... 19-17
  19.7.18 setUCMjspServerEnabled ..................................................................... 19-17
  19.7.19 getUCMjspEnabledGroups ................................................................... 19-18
  19.7.20 setUCMjspEnabledGroups ................................................................... 19-18
19.8 Content Security Configuration Commands ................................................... 19-18
  19.8.1 getUCMCopyAccess ............................................................................... 19-19
  19.8.2 setUCMCopyAccess ............................................................................... 19-19
  19.8.3 getUCMExclusiveCheckout ..................................................................... 19-20
  19.8.4 setUCMExclusiveCheckout ..................................................................... 19-20
  19.8.5 getUCMAuthorDelete ............................................................................. 19-21
  19.8.6 setUCMAuthorDelete ............................................................................. 19-21
  19.8.7 getUCMShowOnlyKnownAccounts ............................................................ 19-21
  19.8.8 setUCMShowOnlyKnownAccounts ............................................................ 19-22
19.9 Component Manager Configuration Commands ............................................. 19-22
  19.9.1 getUCMComponentStatus ....................................................................... 19-23
  19.9.2 setUCMComponentStatus ....................................................................... 19-23
  19.9.3 installUCMComponent .......................................................................... 19-24
  19.9.4 uninstallUCMComponent ...................................................................... 19-24
  19.9.5 downloadUCMComponent ...................................................................... 19-24
  19.9.6 getUCMComponentConfig ...................................................................... 19-25
  19.9.7 updateUCMComponentConfig ................................................................. 19-25
19.10 Records Management Configuration Commands .......................................... 19-26
  19.10.1 getRMLevel .......................................................................................... 19-26
  19.10.2 getRMConfigurationLevel ...................................................................... 19-27
  19.10.3 getRMFeatures ...................................................................................... 19-27
  19.10.4 getRMDispositionActions ..................................................................... 19-28
  19.10.5 rmUpdate ............................................................................................... 19-29
  19.10.6 addOutgoingProvider ........................................................................... 19-29
  19.10.7 registerSource ....................................................................................... 19-30
19.11 User Interface Commands ........................................................................... 19-30
20 Oracle WebCenter Enterprise Capture Custom WLST Commands

20.1 Overview of WLST Oracle WebCenter Enterprise Capture Command Categories .......................... 20-1
20.2 Configuration Commands ........................................................................................................ 20-2
  20.2.1 listWorkspaces .................................................................................................................. 20-3
  20.2.2 listBatches ......................................................................................................................... 20-3
  20.2.3 exportBatch ....................................................................................................................... 20-3
  20.2.4 exportWorkspace ............................................................................................................... 20-4
  20.2.5 importWorkspace ............................................................................................................. 20-4
  20.2.6 unlockBatch ...................................................................................................................... 20-5
  20.2.7 listLockedBatches ............................................................................................................ 20-5
  20.2.8 listCaptureConfig ............................................................................................................ 20-6
  20.2.9 getCaptureConfig .......................................................................................................... 20-6
  20.2.10 setCaptureConfig ......................................................................................................... 20-7
  20.2.11 scanForClientBundles ................................................................................................... 20-7
  20.2.12 deleteBundle ................................................................................................................ 20-7
  20.2.13 setObjectProperty ....................................................................................................... 20-8
  20.2.14 getObjectProperty ....................................................................................................... 20-9
  20.2.15 setObjectCredentials ................................................................................................. 20-10
  20.2.16 updateScript ................................................................................................................. 20-10

21 Enterprise Scheduler Custom WLST Commands

21.1 Enterprise Scheduler Custom WLST Commands ........................................................................ 21-1
  21.1.1 essGetRequestContent ................................................................................................. 21-2
  21.1.2 essManageRequest .................................................................................................... 21-3
  21.1.3 essManageRuntimeConfig ......................................................................................... 21-3
  21.1.4 essManageServer .................................................................................................... 21-4
  21.1.5 essQueryRequests ..................................................................................................... 21-5

A Capture Object Properties

A.1 Client Profile Object Properties .......................................................................................... A-1
A.2 Import Processor Job Object Properties ............................................................................... A-3
  A.2.1 General Import Job Properties ....................................................................................... A-3
  A.2.2 Folder Import Job Specific Properties ............................................................................. A-5
  A.2.3 List File Import Job Specific Properties ........................................................................ A-6
  A.2.4 Email Job Specific Properties ....................................................................................... A-6
A.3 Recognition Processor Job Object Properties ......................................................................... A-8
  A.3.1 General Recognition Job Properties ............................................................................... A-8
A.3.2 BarcodeDefinition Class Properties ................................................................. A-11
A.3.3 DocumentDefinition Class Properties ............................................................. A-11
A.3.4 RecognitionJobField Class Properties ............................................................ A-12
A.3.5 SeparatorDefinition Class Properties ............................................................... A-12
A.4 Commit Profile Object Properties ........................................................................ A-12
A.4.1 General Commit Profile Properties .................................................................... A-13
A.4.2 Text File Commit Profile Specific Properties ..................................................... A-14
A.4.3 Webcenter Content Commit Profile Specific Properties ..................................... A-15
A.4.4 ContentAttributeMappingInfo Class Properties .............................................. A-16
A.4.5 CaptureToContentFieldMapping Class Properties .......................................... A-16
A.4.6 ContentFieldNameInfo Class Properties ......................................................... A-16
A.4.7 Webcenter Imaging Commit Profile Specific Properties .................................. A-16
A.4.8 PDF Searchable Document Output Specific Properties .................................. A-17
A.4.9 DOCS Commit Driver Properties ..................................................................... A-18
A.5 Document Conversion Job Object Properties ...................................................... A-18
This preface describes the document accessibility features and conversions used in this guide—WebLogic Scripting Tool Command Reference.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Conventions

The following text conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>boldface</td>
<td>Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.</td>
</tr>
<tr>
<td>italic</td>
<td>Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.</td>
</tr>
<tr>
<td>monospace</td>
<td>Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.</td>
</tr>
</tbody>
</table>
This section describes the contents and organization of this guide—*WebLogic Scripting Tool Command Reference*.

- **Section 1.1, "Document Scope and Audience"**
- **Section 1.2, "Guide to This Document"**
- **Section 1.3, "Related Documentation"**
- **Section 1.4, "New and Changed WLST Features in This Release"**

### 1.1 Document Scope and Audience

This document describes all of the commands that are available to use with the WebLogic Scripting Tool (WLST). This document includes WLST commands for WebLogic Server, as well as custom WLST commands that can be used to manage installed Oracle Fusion Middleware components.

**Note:** Custom WLST commands for a given Oracle Fusion Middleware component are available for use only if the component is installed in the `ORACLE_HOME` directory.

This document is written for WebLogic Server administrators and operators who deploy Java EE applications using the Java Platform, Enterprise Edition (Java EE) from Oracle. It is assumed that readers are familiar with Web technologies and the operating system and platform where WebLogic Server is installed.

### 1.2 Guide to This Document

This document is organized as follows:

- This chapter, "Introduction and Roadmap," introduces the organization of this guide and lists related documentation.
- **Chapter 2, "WebLogic Server WLST Online and Offline Command Reference,"** summarizes WebLogic Server WLST commands alphabetically and by online/offline usage.
- **Chapter 3, "WLST Command and Variable Reference,"** provides detailed descriptions for each of the WebLogic Server WLST commands and variables.
- **Chapter 4, "Infrastructure Security Custom WLST Commands,"** provides detailed descriptions for each of the custom WLST commands that can be used to manage the Oracle Fusion Middleware Infrastructure Security components.
Chapter 5, "User Messaging Service (UMS) Custom WLST Commands," provides detailed descriptions of the custom WLST commands that can be used to manage the Oracle Fusion Middleware User Messaging Service (UMS) component.

Chapter 6, "DMS Custom WLST Commands," provides detailed descriptions of the custom WLST commands that can be used to manage the Oracle Fusion Middleware Dynamic Monitoring Service (DMS) component.

Chapter 7, "Logging Custom WLST Commands," provides detailed descriptions of the custom WLST commands that can be used to manage the Oracle Fusion Middleware Logging component.

Chapter 8, "Metadata Services (MDS) Custom WLST Commands," provides detailed descriptions of the custom WLST commands that can be used to manage the Oracle Fusion Middleware Metadata Services (MDS) component.

Chapter 9, "Oracle SOA Suite Custom WLST Commands," provides detailed descriptions of the custom WLST commands that can be used to manage the Oracle Fusion Middleware SOA component.

Chapter 10, "WebCenter Portal Custom WLST Commands," provides detailed descriptions for each of the custom WLST commands that can be used to manage the Oracle Fusion Middleware WebCenter Portal component.

Chapter 11, "Application Development Framework (ADF) Custom WLST Commands," provides detailed descriptions of the custom WLST commands that can be used to manage the Oracle Fusion Middleware ADF component.

Chapter 12, "Portal Custom WLST Commands," provides detailed descriptions of the custom WLST commands that can be used to manage the Oracle Fusion Middleware Portals component.

Chapter 13, "Java Required Files Custom WLST Commands," provides detailed descriptions of the custom WLST commands that can be used to manage the Oracle Fusion Middleware JRF component.

Chapter 14, "Web Services Custom WLST Commands," provides detailed descriptions of the custom WLST commands that can be used to manage the Oracle Fusion Middleware Web Services component.

Chapter 15, "Diagnostic Framework Custom WLST Commands," provides detailed descriptions of the custom WLST commands that can be used to manage the Oracle Fusion Middleware Diagnostic Framework component.

Chapter 16, "Information Rights Management Custom WLST Commands," provides detailed descriptions of the custom WLST commands that can be used to manage the Oracle Fusion Middleware Information Rights Management component.

Chapter 17, "Oracle WebCenter: Imaging Custom WLST Commands," provides detailed descriptions of the custom WLST commands that can be used to manage the Oracle Imaging and Process Management component.


Chapter 19, "Oracle WebCenter Content Custom WLST Commands," provides detailed descriptions of the custom WLST commands for Oracle WebCenter Content.
Chapter 20, "Oracle WebCenter Enterprise Capture Custom WLST Commands," provides detailed descriptions of the custom WLST commands for Oracle WebCenter Capture.

Chapter 21, "Enterprise Scheduler Custom WLST Commands," provides detailed descriptions of the custom WLST commands for Oracle Enterprise Scheduling Service (ESS).

Chapter A, "Capture Object Properties," lists all the properties that can be set for each Capture object using the `setObjectProperty` command.

1.3 Related Documentation

For information about how to use the WebLogic Scripting Tool, refer to *Oracle Fusion Middleware Oracle WebLogic Scripting Tool*.

WLST is one of several interfaces for managing and monitoring WebLogic Server. For information about the other management interfaces, see:

- "Deployment Tools" in *Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server* describes several tools that WebLogic Server provides for deploying applications and stand-alone modules.
- *Administration Console Online Help* describes a Web-based graphical user interface for managing and monitoring WebLogic domains.
- *Creating WebLogic Domains Using the Configuration Wizard* describes using a graphical user interface to create a WebLogic domain or extend an existing one.
- *Creating Templates and Domains Using the Pack and Unpack Commands* describes commands that recreate existing WebLogic domains quickly and easily.
- *Oracle Fusion Middleware Developing Custom Management Utilities With JMX for Oracle WebLogic Server* describes using Java Management Extensions (JMX) APIs to monitor and modify WebLogic Server resources.
- *Oracle Fusion Middleware Administrator’s Guide* describes how to manage Oracle Fusion Middleware, including how to start and stop Oracle Fusion Middleware, how to configure and reconfigure components, and how to back up and recover.

1.4 New and Changed WLST Features in This Release

For a comprehensive listing of the new WebLogic Server features introduced in this release, see *Oracle Fusion Middleware What’s New in Oracle WebLogic Server*. 
This chapter provides an alphabetized lists of all WLST online and offline commands. The following sections summarize the WebLogic Server WLST commands, as follows:

- Section 2.1, "WebLogic Server WLST Command Summary, Alphabetically By Command"
- Section 2.2, "WebLogic Server WLST Online Command Summary"
- Section 2.3, "WebLogic Server WLST Offline Command Summary"

**Note:** You can list a summary of all online and offline commands from the command-line using the following commands, respectively:

```python
help("online")
help("offline")
```

For information about custom WLST commands for Fusion Middleware (FMW) components, refer to the appropriate chapter in this document. For information on how to run FMW custom commands, see "Using Custom WLST Commands" in the *Oracle Fusion Middleware Administrator’s Guide*.

### 2.1 WebLogic Server WLST Command Summary, Alphabetically By Command

The following tables summarizes each of the WebLogic Server WLST commands, alphabetically by command. This table does not include custom WLST commands for FMW components. For a list of custom commands for a given FMW component, refer to the appropriate chapter in this document.

**Table 2–1  WebLogic Server WLST Command Summary**

<table>
<thead>
<tr>
<th>This command...</th>
<th>Enables you to...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>activate</td>
<td>Activate changes saved during the current editing session but not yet deployed.</td>
<td>Online</td>
</tr>
<tr>
<td>addHelpCommand</td>
<td>Adds new command help for a command to an existing command group. Once added to the group, the command (along with a brief description) is displayed in the command list for the group when you enter the help('commandGroup') command.</td>
<td>Online or Offline</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>This command...</th>
<th>Enables you to...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>addHelpCommandGroup</code></td>
<td>Adds a new help command group to those shown by the WLST help() command.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td><code>addListener</code></td>
<td>Add a JMX listener to the specified MBean.</td>
<td>Online</td>
</tr>
<tr>
<td><code>addTemplate</code></td>
<td>Extend the current WebLogic domain using an application or service extension template.</td>
<td>Offline</td>
</tr>
<tr>
<td><code>assign</code></td>
<td>Assign resources to one or more destinations.</td>
<td>Offline</td>
</tr>
<tr>
<td><code>cancelEdit</code></td>
<td>Cancel an edit session, release the edit lock, and discard all unsaved changes. This operation can be called by any user with administrator privileges, even if the user did not start the edit session.</td>
<td>Online</td>
</tr>
<tr>
<td><code>cd</code></td>
<td>Navigate the hierarchy of configuration or runtime beans.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td><code>closeDomain</code></td>
<td>Close the current WebLogic domain.</td>
<td>Offline</td>
</tr>
<tr>
<td><code>closeTemplate</code></td>
<td>Close the current domain template.</td>
<td>Offline</td>
</tr>
<tr>
<td><code>configToScript</code></td>
<td>Convert an existing server configuration (config directory) to an executable WLST script.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td><code>connect</code></td>
<td>Connect WLST to a WebLogic Server instance.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td><code>create</code></td>
<td>Create a configuration bean of the specified type for the current bean.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td><code>currentTree</code></td>
<td>Return the current location in the hierarchy.</td>
<td>Online</td>
</tr>
<tr>
<td><code>custom</code></td>
<td>Navigate to the root of custom MBeans that are registered in the Runtime MBean Server.</td>
<td>Online</td>
</tr>
<tr>
<td><code>delete</code></td>
<td>Delete an instance of a configuration bean of the specified type for the current configuration bean.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td><code>deploy</code></td>
<td>Deploy an application to a WebLogic Server instance.</td>
<td>Online</td>
</tr>
<tr>
<td><code>disconnect</code></td>
<td>Disconnect WLST from a WebLogic Server instance.</td>
<td>Online</td>
</tr>
<tr>
<td><code>distributeApplication</code></td>
<td>Copy the deployment bundle to the specified targets.</td>
<td>Online</td>
</tr>
<tr>
<td><code>domainConfig</code></td>
<td>Navigate to the last MBean to which you navigated in the domain configuration hierarchy or to the root of the hierarchy, DomainMBean.</td>
<td>Online</td>
</tr>
<tr>
<td><code>domainCustom</code></td>
<td>Navigate to the tree of custom MBeans that are registered in the Domain Runtime MBean Server.</td>
<td>Online</td>
</tr>
<tr>
<td><code>domainRuntime</code></td>
<td>Navigate to the last MBean to which you navigated in the domain runtime hierarchy or to the root of the hierarchy, DomainRuntimeMBean.</td>
<td>Online</td>
</tr>
<tr>
<td><code>dumpStack</code></td>
<td>Display stack trace from the last exception that occurred while performing a WLST action, and reset the stack trace.</td>
<td>Online or Offline</td>
</tr>
</tbody>
</table>
### Table 2–1 (Cont.) WebLogic Server WLST Command Summary

<table>
<thead>
<tr>
<th>This command...</th>
<th>Enables you to...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>dumpVariables</td>
<td>Display all variables used by WLST, including their name and value.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>edit</td>
<td>Navigate to the last MBean to which you navigated in the configuration edit MBean hierarchy or to the root of the hierarchy, DomainMBean.</td>
<td>Online</td>
</tr>
<tr>
<td>encrypt</td>
<td>Encrypt the specified string.</td>
<td>Online</td>
</tr>
<tr>
<td>exit</td>
<td>Exit WLST from the user session and close the scripting shell.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>exportDiagnosticData</td>
<td>Execute a query against the specified log file.</td>
<td>Offline</td>
</tr>
<tr>
<td>exportDiagnosticDataFromServer</td>
<td>Executes a query on the server side and retrieves the exported WebLogic Diagnostic Framework (WLDF) data.</td>
<td>Online</td>
</tr>
<tr>
<td>find</td>
<td>Find MBeans and attributes in the current hierarchy.</td>
<td>Online</td>
</tr>
<tr>
<td>get</td>
<td>Return the value of the specified attribute.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>getActivationTask</td>
<td>Return the latest ActivationTask MBean on which a user can get status.</td>
<td>Online</td>
</tr>
<tr>
<td>getAvailableCapturedImages</td>
<td>Returns a list of the previously captured diagnostic images.</td>
<td>Online</td>
</tr>
<tr>
<td>getConfigManager</td>
<td>Return the latest ConfigurationManagerBean MBean which manages the change process.</td>
<td>Online</td>
</tr>
<tr>
<td>getMBean</td>
<td>Return the MBean by browsing to the specified path.</td>
<td>Online</td>
</tr>
<tr>
<td>getMBI</td>
<td>Return the MBeanInfo for the specified MBeanType or the cmo variable.</td>
<td>Online</td>
</tr>
<tr>
<td>getPath</td>
<td>Return the MBean path for the specified MBean instance.</td>
<td>Online</td>
</tr>
<tr>
<td>getWLDM</td>
<td>Return the WebLogic DeploymentManager object.</td>
<td>Online</td>
</tr>
<tr>
<td>invoke</td>
<td>Invoke a management operation on the current configuration bean.</td>
<td>Online</td>
</tr>
<tr>
<td>isRestartRequired</td>
<td>Determine whether a server restart is required.</td>
<td>Online</td>
</tr>
<tr>
<td>jndi</td>
<td>Navigates to the JNDI tree for the server to which WLST is currently connected.</td>
<td>Online</td>
</tr>
<tr>
<td>listApplications</td>
<td>List all applications that are currently deployed in the domain.</td>
<td>Online</td>
</tr>
<tr>
<td>listChildTypes</td>
<td>List all the children MBeans that can be created or deleted for the cmo.</td>
<td>Online</td>
</tr>
<tr>
<td>loadApplication</td>
<td>Load an application and deployment plan into memory.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>loadDB</td>
<td>Load SQL files into a database.</td>
<td>Offline</td>
</tr>
<tr>
<td>loadProperties</td>
<td>Load property values from a file.</td>
<td>Online and Offline</td>
</tr>
<tr>
<td>lookup</td>
<td>Look up the specified MBean.</td>
<td>Online</td>
</tr>
</tbody>
</table>
Table 2–1  (Cont.) WebLogic Server WLST Command Summary

<table>
<thead>
<tr>
<th>This command...</th>
<th>Enables you to...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>ls</td>
<td>List all child beans and/or attributes for the current configuration or runtime bean.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>man</td>
<td>Display help from MBeanInfo for the current MBean or its specified attribute.</td>
<td>Online</td>
</tr>
<tr>
<td>migrate</td>
<td>Migrate services to a target server within a cluster.</td>
<td>Online</td>
</tr>
<tr>
<td>nm</td>
<td>Determine whether WLST is connected to Node Manager.</td>
<td>Online</td>
</tr>
<tr>
<td>nmConnect</td>
<td>Connect WLST to Node Manager to establish a session.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>nmDisconnect</td>
<td>Disconnect WLST from a Node Manager session.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>nmEnroll</td>
<td>Enroll the machine on which WLST is currently running.</td>
<td>Online</td>
</tr>
<tr>
<td>nmGenBootStartupProps</td>
<td>Generates the Node Manager property files, \textit{boot.properties} and \textit{startup.properties}, for the specified server.</td>
<td>Online</td>
</tr>
<tr>
<td>nmKill</td>
<td>Kill the specified server instance that was started with Node Manager.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>nmLog</td>
<td>Return the Node Manager log.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>nmServerLog</td>
<td>Return the server output log of the server that was started with Node Manager.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>nmServerStatus</td>
<td>Return the status of the server that was started with Node Manager.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>nmStart</td>
<td>Start a server in the current domain using Node Manager.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>nmVersion</td>
<td>Return the Node Manager server version.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>prompt</td>
<td>Toggle the display of path information at the prompt.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>pwd</td>
<td>Display the current location in the configuration or runtime bean hierarchy.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>readDomain</td>
<td>Open an existing WebLogic domain for updating.</td>
<td>Offline</td>
</tr>
<tr>
<td>readTemplate</td>
<td>Open an existing domain template for WebLogic domain creation.</td>
<td>Offline</td>
</tr>
<tr>
<td>redeploy</td>
<td>Reload classes and redeploy a previously deployed application.</td>
<td>Online</td>
</tr>
<tr>
<td>This command...</td>
<td>Enables you to...</td>
<td>Use with WLST...</td>
</tr>
<tr>
<td>-----------------</td>
<td>-------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>redirect</td>
<td>Redirect WLST output to the specified filename.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>removeListener</td>
<td>Remove a listener that was previously defined.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>resume</td>
<td>Resume a server instance that is suspended or in <strong>ADMIN</strong> state.</td>
<td>Online</td>
</tr>
<tr>
<td>save</td>
<td>Save the edits that have been made but have not yet been saved.</td>
<td>Online</td>
</tr>
<tr>
<td>saveDiagnosticImageCaptureFile</td>
<td>Downloads the specified diagnostic image capture.</td>
<td>Online</td>
</tr>
<tr>
<td>saveDiagnosticImageCaptureEntryFile</td>
<td>Downloads a specific entry from the diagnostic image capture.</td>
<td>Online</td>
</tr>
<tr>
<td>serverRuntime</td>
<td>Navigate to the last MBean to which you navigated in the runtime MBean hierarchy or to the root of the hierarchy, <strong>ServerRuntimeMBean</strong>.</td>
<td>Online</td>
</tr>
<tr>
<td>set</td>
<td>Set the specified attribute value for the current configuration bean.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>setOption</td>
<td>Set options related to a WebLogic domain creation or update</td>
<td>Offline</td>
</tr>
<tr>
<td>showChanges</td>
<td>Show the changes made by the current user during the current edit session.</td>
<td>Online</td>
</tr>
<tr>
<td>showListeners</td>
<td>Show all listeners that are currently defined.</td>
<td>Online</td>
</tr>
<tr>
<td>shutdown</td>
<td>Gracefully shut down a running server instance or cluster.</td>
<td>Online</td>
</tr>
<tr>
<td>start</td>
<td>Start a Managed Server instance or a cluster using Node Manager.</td>
<td>Online</td>
</tr>
<tr>
<td>startApplication</td>
<td>Start an application, making it available to users.</td>
<td>Online</td>
</tr>
<tr>
<td>startEdit</td>
<td>Start a configuration edit session on behalf of the currently connected user.</td>
<td>Online</td>
</tr>
<tr>
<td>startNodeManager</td>
<td>Start Node Manager at default port (5556).</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>startRecording</td>
<td>Record all user interactions with WLST; useful for capturing commands to replay.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>startServer</td>
<td>Start the Administration Server.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>state</td>
<td>Returns a map of servers or clusters and their state using Node Manager.</td>
<td>Online</td>
</tr>
<tr>
<td>stopApplication</td>
<td>Stop an application, making it un available to users.</td>
<td>Online</td>
</tr>
<tr>
<td>stopEdit</td>
<td>Stop the current edit session, release the edit lock, and discard unsaved changes.</td>
<td>Online</td>
</tr>
<tr>
<td>stopNodeManager</td>
<td>Stop Node Manager.</td>
<td>Online or Offline</td>
</tr>
</tbody>
</table>
### 2.2 WebLogic Server WLST Online Command Summary

The following table summarizes the WebLogic Server WLST online commands, alphabetically by command. This table does not include custom WLST commands for FMW components. For a list of custom commands for a given FMW component, refer to the appropriate chapter in this document.

#### Table 2–2 WebLogic Server WLST Online Command Summary

<table>
<thead>
<tr>
<th>This command...</th>
<th>Enables you to...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>activate</td>
<td>Activate changes saved during the current editing session but not yet deployed.</td>
<td>Online</td>
</tr>
<tr>
<td>addHelpCommand</td>
<td>Adds new command help for a command to an existing command group. Once added to the group, the command (along with a brief description) is displayed in the command list for the group when you enter the <code>help('commandGroup')</code> command.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>This command...</td>
<td>Enables you to...</td>
<td></td>
</tr>
<tr>
<td>---------------------------------</td>
<td>----------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>addHelpCommandGroup</td>
<td>Adds a new help command group to those shown by the WLST help() command, and specifies the resource bundle in which the help information is defined for the group.</td>
<td></td>
</tr>
<tr>
<td>addListener</td>
<td>Add a JMX listener to the specified MBean.</td>
<td></td>
</tr>
<tr>
<td>cancelEdit</td>
<td>Cancel an edit session, release the edit lock, and discard all unsaved changes. This operation can be called by any user with administrator privileges, even if the user did not start the edit session.</td>
<td></td>
</tr>
<tr>
<td>cd</td>
<td>Navigate the hierarchy of configuration or runtime beans.</td>
<td></td>
</tr>
<tr>
<td>configToScript</td>
<td>Convert an existing server configuration (config directory) to an executable WLST script.</td>
<td></td>
</tr>
<tr>
<td>connect</td>
<td>Connect WLST to a WebLogic Server instance.</td>
<td></td>
</tr>
<tr>
<td>create</td>
<td>Create a configuration bean of the specified type for the current bean.</td>
<td></td>
</tr>
<tr>
<td>currentTree</td>
<td>Return the current tree location.</td>
<td></td>
</tr>
<tr>
<td>custom</td>
<td>Navigate to the root of custom MBeans that are registered in the Runtime MBean Server.</td>
<td></td>
</tr>
<tr>
<td>delete</td>
<td>Delete an instance of a configuration bean of the specified type for the current configuration bean.</td>
<td></td>
</tr>
<tr>
<td>deploy</td>
<td>Deploy an application to a WebLogic Server instance.</td>
<td></td>
</tr>
<tr>
<td>disconnect</td>
<td>Disconnect WLST from a WebLogic Server instance.</td>
<td></td>
</tr>
<tr>
<td>distributeApplication</td>
<td>Copy the deployment bundle to the specified targets.</td>
<td></td>
</tr>
<tr>
<td>domainConfig</td>
<td>Navigate to the last MBean to which you navigated in the domain configuration hierarchy or to the root of the hierarchy, DomainMBean.</td>
<td></td>
</tr>
<tr>
<td>domainCustom</td>
<td>Navigate to the tree of custom MBeans that are registered in the Domain Runtime MBean Server.</td>
<td></td>
</tr>
<tr>
<td>domainRuntime</td>
<td>Navigate to the last MBean to which you navigated in the domain runtime hierarchy or to the root of the hierarchy, DomainRuntimeMBean.</td>
<td></td>
</tr>
<tr>
<td>dumpStack</td>
<td>Display stack trace from the last exception that occurred, and reset the trace.</td>
<td></td>
</tr>
<tr>
<td>dumpVariables</td>
<td>Display all variables used by WLST, including their name and value.</td>
<td></td>
</tr>
<tr>
<td>edit</td>
<td>Navigate to the last MBean to which you navigated in the configuration edit MBean hierarchy or to the root of the hierarchy, DomainMBean.</td>
<td></td>
</tr>
<tr>
<td>encrypt</td>
<td>Encrypt the specified string.</td>
<td></td>
</tr>
<tr>
<td>exit</td>
<td>Exit WLST from the interactive session and close the scripting shell.</td>
<td></td>
</tr>
<tr>
<td>exportDiagnosticDataFromServer</td>
<td>Execute a query on the server side and retrieves the exported WebLogic Diagnostic Framework (WLDF) data.</td>
<td></td>
</tr>
<tr>
<td>find</td>
<td>Find MBeans and attributes in the current hierarchy.</td>
<td></td>
</tr>
<tr>
<td>get</td>
<td>Return the value of the specified attribute.</td>
<td></td>
</tr>
<tr>
<td>getActivationTask</td>
<td>Return the latest ActivationTask MBean on which a user can get status.</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><code>getAvailableCapturedImages</code></td>
<td>Returns a list of the previously captured diagnostic images.</td>
<td></td>
</tr>
<tr>
<td><code>getConfigManager</code></td>
<td>Return the latest ConfigurationManagerBean MBean which manages the change process.</td>
<td></td>
</tr>
<tr>
<td><code>getMBean</code></td>
<td>Return the MBean by browsing to the specified path.</td>
<td></td>
</tr>
<tr>
<td><code>getMBI</code></td>
<td>Return the MBeanInfo for the specified MBeanType or the <code>cmo</code> variable.</td>
<td></td>
</tr>
<tr>
<td><code>getPath</code></td>
<td>Return the MBean path for the specified MBean instance.</td>
<td></td>
</tr>
<tr>
<td><code>getWLDM</code></td>
<td>Return the WebLogic DeploymentManager object.</td>
<td></td>
</tr>
<tr>
<td><code>invoke</code></td>
<td>Invoke a management operation on the current configuration bean.</td>
<td></td>
</tr>
<tr>
<td><code>isRestartRequired</code></td>
<td>Determine whether a server restart is required.</td>
<td></td>
</tr>
<tr>
<td><code>jndi</code></td>
<td>Navigates to the JNDI tree for the server to which WLST is currently connected.</td>
<td></td>
</tr>
<tr>
<td><code>listApplications</code></td>
<td>List all applications that are currently deployed in the domain.</td>
<td></td>
</tr>
<tr>
<td><code>listChildTypes</code></td>
<td>List all the children MBeans that can be created or deleted for the <code>cmo</code>.</td>
<td></td>
</tr>
<tr>
<td><code>loadApplication</code></td>
<td>Load an application and deployment plan into memory.</td>
<td></td>
</tr>
<tr>
<td><code>loadProperties</code></td>
<td>Load property values from a file.</td>
<td></td>
</tr>
<tr>
<td><code>lookup</code></td>
<td>Look up the specified MBean.</td>
<td></td>
</tr>
<tr>
<td><code>ls</code></td>
<td>List all child beans and/or attributes for the current configuration or runtime bean.</td>
<td></td>
</tr>
<tr>
<td><code>man</code></td>
<td>Display help from MBeanInfo for the current MBean or its specified attribute.</td>
<td></td>
</tr>
<tr>
<td><code>migrate</code></td>
<td>Migrate services to a target server within a cluster.</td>
<td></td>
</tr>
<tr>
<td><code>nm</code></td>
<td>Determine whether WLST is connected to Node Manager.</td>
<td></td>
</tr>
<tr>
<td><code>nmConnect</code></td>
<td>Connect WLST to Node Manager to establish a session.</td>
<td></td>
</tr>
<tr>
<td><code>nmDisconnect</code></td>
<td>Disconnect WLST from a Node Manager session.</td>
<td></td>
</tr>
<tr>
<td><code>nmEnroll</code></td>
<td>Enroll the machine on which WLST is currently running.</td>
<td></td>
</tr>
<tr>
<td><code>nmGenBootStartupProps</code></td>
<td>Generates the Node Manager property files, <code>boot.properties</code> and <code>startup.properties</code>, for the specified server.</td>
<td></td>
</tr>
<tr>
<td><code>nmKill</code></td>
<td>Kill the specified server instance that was started with Node Manager.</td>
<td></td>
</tr>
<tr>
<td><code>nmLog</code></td>
<td>Return the Node Manager log.</td>
<td></td>
</tr>
<tr>
<td><code>nmServerLog</code></td>
<td>Return the server output log of the server that was started with Node Manager.</td>
<td></td>
</tr>
<tr>
<td><code>nmServerStatus</code></td>
<td>Return the status of the server that was started with Node Manager.</td>
<td></td>
</tr>
<tr>
<td><code>nmStart</code></td>
<td>Start a server in the current domain using Node Manager.</td>
<td></td>
</tr>
<tr>
<td><code>nmVersion</code></td>
<td>Return the Node Manager server version.</td>
<td></td>
</tr>
<tr>
<td><code>prompt</code></td>
<td>Toggle the display of path information at the prompt.</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>pwd</td>
<td>Display the current location in the configuration or runtime bean hierarchy.</td>
<td></td>
</tr>
<tr>
<td>redeploy</td>
<td>Reload classes and redeploy a previously deployed application.</td>
<td></td>
</tr>
<tr>
<td>redirect</td>
<td>Redirect WLST output to the specified filename.</td>
<td></td>
</tr>
<tr>
<td>removeListener</td>
<td>Remove a listener that was previously defined.</td>
<td></td>
</tr>
<tr>
<td>resume</td>
<td>Resume a server instance that is suspended or in ADMIN state.</td>
<td></td>
</tr>
<tr>
<td>save</td>
<td>Save the edits that have been made but have not yet been saved.</td>
<td></td>
</tr>
<tr>
<td>saveDiagnosticImageCaptureFile</td>
<td>Downloads the specified diagnostic image capture.</td>
<td></td>
</tr>
<tr>
<td>saveDiagnosticImageCaptureEntryFile</td>
<td>Downloads a specific entry from the diagnostic image capture.</td>
<td></td>
</tr>
<tr>
<td>serverConfig</td>
<td>Navigate to the last MBean to which you navigated in the configuration MBean hierarchy or to the root of the hierarchy, DomainMBean.</td>
<td></td>
</tr>
<tr>
<td>serverRuntime</td>
<td>Navigate to the last MBean to which you navigated in the runtime MBean hierarchy or to the root of the hierarchy, ServerRuntimeMBean.</td>
<td></td>
</tr>
<tr>
<td>set</td>
<td>Set the specified attribute value for the current configuration bean.</td>
<td></td>
</tr>
<tr>
<td>showChanges</td>
<td>Show the changes made by the current user during the current edit session.</td>
<td></td>
</tr>
<tr>
<td>showListeners</td>
<td>Show all listeners that are currently defined.</td>
<td></td>
</tr>
<tr>
<td>shutdown</td>
<td>Gracefully shut down a running server instance or cluster.</td>
<td></td>
</tr>
<tr>
<td>start</td>
<td>Start a Managed Server instance or a cluster using Node Manager.</td>
<td></td>
</tr>
<tr>
<td>startApplication</td>
<td>Start an application, making it available to users.</td>
<td></td>
</tr>
<tr>
<td>startEdit</td>
<td>Start a configuration edit session on behalf of the currently connected user.</td>
<td></td>
</tr>
<tr>
<td>startNodeManager</td>
<td>Start Node Manager at default port (5556).</td>
<td></td>
</tr>
<tr>
<td>startRecording</td>
<td>Record all user interactions with WLST; useful for capturing commands to replay.</td>
<td></td>
</tr>
<tr>
<td>startServer</td>
<td>Start the Administration Server.</td>
<td></td>
</tr>
<tr>
<td>state</td>
<td>Returns a map of servers or clusters and their state using Node Manager.</td>
<td></td>
</tr>
<tr>
<td>stopApplication</td>
<td>Stop an application, making it un available to users.</td>
<td></td>
</tr>
<tr>
<td>stopEdit</td>
<td>Stop the current edit session, release the edit lock, and discard unsaved changes.</td>
<td></td>
</tr>
<tr>
<td>stopNodeManager</td>
<td>Stop Node Manager.</td>
<td></td>
</tr>
<tr>
<td>stopRedirect</td>
<td>Stop the redirection of WLST output to a file.</td>
<td></td>
</tr>
<tr>
<td>storeUserConfig</td>
<td>Create a user configuration file and an associated key file.</td>
<td></td>
</tr>
<tr>
<td>suspend</td>
<td>Suspend a running server.</td>
<td></td>
</tr>
<tr>
<td>threadDump</td>
<td>Display a thread dump for the specified server.</td>
<td></td>
</tr>
<tr>
<td>undeploy</td>
<td>Undeploy an application from the specified servers.</td>
<td></td>
</tr>
</tbody>
</table>
The following table summarizes the WebLogic Server WLST offline commands, alphabetically by command.

<table>
<thead>
<tr>
<th>This command...</th>
<th>Enables you to...</th>
</tr>
</thead>
<tbody>
<tr>
<td>undo</td>
<td>Revert all unsaved or unactivated edits.</td>
</tr>
<tr>
<td>updateApplication</td>
<td>Update an application configuration using a new deployment plan.</td>
</tr>
<tr>
<td>validate</td>
<td>Validate the changes that have been made but have not yet been saved.</td>
</tr>
<tr>
<td>viewMBean</td>
<td>Display information about an MBean, such as the attribute names and values, and operations.</td>
</tr>
<tr>
<td>writeIniFile</td>
<td>Convert WLST definitions and method declarations to a Python (.py) file.</td>
</tr>
</tbody>
</table>

### 2.3 WebLogic Server WLST Offline Command Summary

The following table summarizes the WebLogic Server WLST offline commands, alphabetically by command.

#### Table 2–3 WebLogic Server WLST Offline Command Summary

<table>
<thead>
<tr>
<th>This command...</th>
<th>Enables you to...</th>
</tr>
</thead>
<tbody>
<tr>
<td>addHelpCommand</td>
<td>Adds new command help for a command to an existing command group. Once added to the group, the command (along with a brief description) is displayed in the command list for the group when you enter the <code>help('commandGroup')</code> command.</td>
</tr>
<tr>
<td>addHelpCommandGroup</td>
<td>Adds a new help command group to those shown by the WLST <code>help()</code> command, and specifies the resource bundle in which the help information is defined for the group.</td>
</tr>
<tr>
<td>addTemplate</td>
<td>Extend the current domain using an application or service extension template.</td>
</tr>
<tr>
<td>assign</td>
<td>Assign resources to one or more destinations.</td>
</tr>
<tr>
<td>cd</td>
<td>Navigate the hierarchy of configuration or runtime beans.</td>
</tr>
<tr>
<td>closeDomain</td>
<td>Close the current domain.</td>
</tr>
<tr>
<td>closeTemplate</td>
<td>Close the current domain template.</td>
</tr>
<tr>
<td>configToScript</td>
<td>Convert an existing server configuration (<code>config</code> directory) to an executable WLST script.</td>
</tr>
<tr>
<td>connect</td>
<td>Connect WLST to a WebLogic Server instance.</td>
</tr>
<tr>
<td>create</td>
<td>Create a configuration bean of the specified type for the current bean.</td>
</tr>
<tr>
<td>delete</td>
<td>Delete an instance of a configuration bean of the specified type for the current configuration bean.</td>
</tr>
<tr>
<td>dumpStack</td>
<td>Display stack trace from the last exception that occurred while performing a WLST action, and reset the stack trace.</td>
</tr>
<tr>
<td>dumpVariables</td>
<td>Display all variables used by WLST, including their name and value.</td>
</tr>
<tr>
<td>exit</td>
<td>Exit WLST from the interactive session and close the scripting shell.</td>
</tr>
<tr>
<td>exportDiagnosticData</td>
<td>Execute a query against the specified log file.</td>
</tr>
<tr>
<td>get</td>
<td>Return the value of the specified attribute.</td>
</tr>
</tbody>
</table>
# Table 2–3 (Cont.) WebLogic Server WLST Offline Command Summary

<table>
<thead>
<tr>
<th>This command...</th>
<th>Enables you to...</th>
</tr>
</thead>
<tbody>
<tr>
<td>loadApplication</td>
<td>Load an application and deployment plan into memory.</td>
</tr>
<tr>
<td>loadDB</td>
<td>Load SQL files into a database.</td>
</tr>
<tr>
<td>loadProperties</td>
<td>Load property values from a file.</td>
</tr>
<tr>
<td>ls</td>
<td>List all child beans and/or attributes for the current configuration or runtime bean.</td>
</tr>
<tr>
<td>nmConnect</td>
<td>Connect WLST to Node Manager to establish a session.</td>
</tr>
<tr>
<td>prompt</td>
<td>Toggle the display of path information at the prompt.</td>
</tr>
<tr>
<td>pwd</td>
<td>Display the current location in the configuration or runtime bean hierarchy.</td>
</tr>
<tr>
<td>readDomain</td>
<td>Open an existing WebLogic domain for updating.</td>
</tr>
<tr>
<td>readTemplate</td>
<td>Open an existing domain template for domain creation.</td>
</tr>
<tr>
<td>redirect</td>
<td>Redirect WLST output to the specified filename.</td>
</tr>
<tr>
<td>set</td>
<td>Set the specified attribute value for the current configuration bean.</td>
</tr>
<tr>
<td>setOption</td>
<td>Set options related to a WebLogic domain creation or update.</td>
</tr>
<tr>
<td>startNodeManager</td>
<td>Start Node Manager at default port (5556).</td>
</tr>
<tr>
<td>startRecording</td>
<td>Record all user interactions with WLST; useful for capturing commands to replay.</td>
</tr>
<tr>
<td>startServer</td>
<td>Start the Administration Server.</td>
</tr>
<tr>
<td>stopNodeManager</td>
<td>Stop Node Manager.</td>
</tr>
<tr>
<td>stopRedirect</td>
<td>Stop the redirection of WLST output to a file.</td>
</tr>
<tr>
<td>threadDump</td>
<td>Display a thread dump for the specified server.</td>
</tr>
<tr>
<td>unassign</td>
<td>Unassign applications or services from one or more destinations.</td>
</tr>
<tr>
<td>updateDomain</td>
<td>Update and save the current domain.</td>
</tr>
<tr>
<td>writeDomain</td>
<td>Write the domain configuration information to the specified directory.</td>
</tr>
<tr>
<td>writeIniFile</td>
<td>Convert WLST definitions and method declarations to a Python (.py) file.</td>
</tr>
<tr>
<td>writeTemplate</td>
<td>Writes the domain configuration information to the specified domain template.</td>
</tr>
</tbody>
</table>
This chapter provides detailed descriptions of WLST commands, command syntax, arguments and command examples. It also lists and describes the available WLST variables.

The following sections describe the WLST commands and variables in detail. Topics include:

- Section 3.1, "Overview of WLST Command Categories"
- Section 3.2, "Browse Commands"
- Section 3.3, "Control Commands"
- Section 3.4, "Customization Commands"
- Section 3.5, "Deployment Commands"
- Section 3.6, "Diagnostics Commands"
- Section 3.7, "Editing Commands"
- Section 3.8, "Information Commands"
- Section 3.9, "Life Cycle Commands"
- Section 3.10, "Node Manager Commands"
- Section 3.11, "Tree Commands"
- Section 3.12, "WLST Variable Reference"

### 3.1 Overview of WLST Command Categories

**Note:** It is recommended that you review "Syntax for WLST Commands" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool for command syntax requirements.

WLST commands are divided into the following categories.

<table>
<thead>
<tr>
<th>Command Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section 3.2, &quot;Browse Commands&quot;</td>
<td>Navigate the hierarchy of configuration or runtime beans and control the prompt display.</td>
</tr>
</tbody>
</table>
3.2 Browse Commands

Use the WLST browse commands, listed in Table 3–2, to navigate the hierarchy of configuration or runtime beans and control the prompt display.

### Table 3–2  Browse Commands for WLST Configuration

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cd</code></td>
<td>Navigate the hierarchy of configuration or runtime beans.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td><code>currentTree</code></td>
<td>Return the current location in the hierarchy.</td>
<td>Online</td>
</tr>
<tr>
<td><code>prompt</code></td>
<td>Toggle the display of path information at the prompt.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td><code>pwd</code></td>
<td>Display the current location in the hierarchy.</td>
<td>Online or Offline</td>
</tr>
</tbody>
</table>

#### 3.2.1 cd

Command Category: Browse Commands

Use with WLST: Online or Offline
3.2.1.1 Description
Navigates the hierarchy of configuration or runtime beans. This command uses a model that is similar to navigating a file system in a Windows or UNIX command shell. For example, to navigate back to a parent configuration or runtime bean, enter cd('..'). The character string .. (dot-dot), refers to the directory immediately above the current directory. To get back to the root bean after navigating to a bean that is deep in the hierarchy, enter cd('/').

You can navigate to beans in the current hierarchy and to any child or instance.

The cd command returns a stub of the configuration or runtime bean instance, if one exists. If you navigate to a type, this command returns a stub of the configuration or runtime bean instance from which you navigated. In the event of an error, the command returns a WLSTException.

---

**Note:** The cmo variable is initialized to the root of all domain configuration beans when you first connect WLST to a server instance. It reflects the parent configuration bean type until you navigate to an instance. For more information about the cmo variable, see "Changing the Current Management Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

---

3.2.1.2 Syntax

```bash
cd(mbeanName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>mbeanName</td>
<td>Path to the bean in the namespace.</td>
</tr>
</tbody>
</table>

3.2.1.3 Examples

The following example navigates the hierarchy of configuration beans. The first command navigates to the Servers configuration bean type, the second, to the myserver configuration bean instance, and the last back up two levels to the original directory location.

```bash
wls:/mydomain/serverConfig> cd('Servers')
wls:/mydomain/serverConfig/Servers> cd('myserver')
wls:/mydomain/serverConfig/Servers/myserver> cd('..//..')
wls:/mydomain/serverConfig>
```

3.2.2 currentTree

Command Category: Browse Commands
Use with WLST: Online

3.2.2.1 Description

Returns the current location in the hierarchy. This command enables you to store the current location in the hierarchy and easily return to it after browsing. In the event of an error, the command returns a WLSTException.

3.2.2.2 Syntax

```bash
currentTree()
```
### 3.2.2.3 Example

The following example stores the current location in the hierarchy in `myTree` and uses it to navigate back to the Edit MBean hierarchy from the runtime MBean hierarchy on an Administration Server instance.

```
   wls:/mydomain/edit> myTree=currentTree()
   wls:/mydomain/edit> serverRuntime()
```

Location changed to serverRuntime tree. This is a read-only tree with ServerRuntimeMBean as the root.

For more help, use help('serverRuntime')

```
   wls:/mydomain/serverRuntime> myTree()
   wls:/mydomain/edit>
```

### 3.2.3 prompt

Command Category: Browse Commands

Use with WLST: Online or Offline

#### 3.2.3.1 Description

Toggles the display of path information at the prompt, when entered without an argument. This command is useful when the prompt becomes too long due to the length of the path.

You can also explicitly specify `on` or `off` as an argument to the command. When you specify `off`, WLST hides the WLST prompt and defaults to the Jython prompt. By default, the WLST prompt displays the configuration or runtime navigation path information.

When you disable the prompt details, to determine your current location in the hierarchy, you can use the `pwd` command, as described in Section 3.2.4, "pwd".

In the event of an error, the command returns a `WLSTException`.

#### 3.2.3.2 Syntax

```
prompt(myPrompt)
```

#### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
</table>
| `myPrompt` | Optional. Hides or displays WLST prompt. Valid values include `off` or `on`.
|           | - The `off` argument hides the WLST prompt.
|           |   - If you run `prompt('off')`, when using WLST online, the prompt defaults to the Jython prompt. You can create a new prompt using Jython syntax. For more information about programming using Jython, see [http://www.jython.org](http://www.jython.org). In this case, if you subsequently enter the `prompt` command without arguments, WLST displays the WLST command prompt without the path information. To redisplay the path information, enter `prompt()` again, or enter `prompt('on')`.
|           | - The `on` argument displays the default WLST prompt, including the path information. |

#### 3.2.3.3 Examples

The following example hides and then redisplays the path information at the prompt.

```
   wls:/mydomain/serverConfig/Servers/myserver> prompt()
   wls:/> prompt()
   wls:/mydomain/serverConfig/Servers/myserver>
```
The following example hides the prompt and defaults to the Jython prompt (since the command is run using WLST online), changes the Jython prompt, and then redisplay
the WLST prompt. This example also demonstrates the use of the `pwd` command.

```
> prompt('off')
>>>
sys.ps1="myprompt>
myprompt> prompt()
wlsl> pwd()
'serverConfig:Servers/myserver'
wls:> prompt()
wls:/mydomain/serverConfig/Servers/myserver>
```

### 3.2.4 `pwd`

**Command Category:** Browse Commands  
**Use with WLST:** Online or Offline  

#### 3.2.4.1 Description

Displays the current location in the configuration or runtime bean hierarchy.  
This command is useful when you have turned off the prompt display of the path information using the `prompt` command, as described in Section 3.2.3, "prompt".  
In the event of an error, the command returns a `WLSTException`.

#### 3.2.4.2 Syntax

`pwd()`  

#### 3.2.4.3 Example

The following example displays the current location in the configuration bean hierarchy.

```
wls:/mydomain/serverConfig/Servers/myserver> pwd()
'serverConfig:Servers/myserver/Log/myserver'
```

### 3.3 Control Commands

Use the WLST control commands, listed in Table 3–3, to perform the following tasks:

- Connect to or disconnect from a server (connect and disconnect commands)
- Create a new WebLogic domain from a domain template, similar to the Configuration Wizard (createDomain, readTemplate, writeDomain, and closeTemplate commands)
- Update an existing WebLogic domain, offline (readDomain, addTemplate, updateDomain, and closeDomain commands)
- Write a domain template (writeTemplate command)
- Exit WLST

Table 3–3 lists the control commands for WLST configuration.
Table 3–3 Control Commands for WLST Configuration

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>connect</td>
<td>Connect WLST to a WebLogic Server instance.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>disconnect</td>
<td>Disconnect WLST from a WebLogic Server instance.</td>
<td>Online</td>
</tr>
<tr>
<td>createDomain</td>
<td>Create a new WebLogic domain using the specified template.</td>
<td>Offline</td>
</tr>
<tr>
<td>readTemplate</td>
<td>Open an existing domain template for domain creation.</td>
<td>Offline</td>
</tr>
<tr>
<td>writeDomain</td>
<td>Write the domain configuration information to the specified directory.</td>
<td>Offline</td>
</tr>
<tr>
<td>closeTemplate</td>
<td>Close the current domain template.</td>
<td>Offline</td>
</tr>
<tr>
<td>readDomain</td>
<td>Open an existing WebLogic domain for updating.</td>
<td>Offline</td>
</tr>
<tr>
<td>addTemplate</td>
<td>Extend the current WebLogic domain using an application or service extension template.</td>
<td>Offline</td>
</tr>
<tr>
<td>updateDomain</td>
<td>Update and save the current domain.</td>
<td>Offline</td>
</tr>
<tr>
<td>closeDomain</td>
<td>Close the current domain.</td>
<td>Offline</td>
</tr>
<tr>
<td>writeTemplate</td>
<td>Writes the configuration information to the specified domain template file.</td>
<td>Offline</td>
</tr>
<tr>
<td>exit</td>
<td>Exit WLST from the interactive session and close the scripting shell.</td>
<td>Online or Offline</td>
</tr>
</tbody>
</table>

3.3.1 addTemplate

Command Category: Control Commands
Use with WLST: Offline

3.3.1.1 Description
Extends the current WebLogic domain using an application or service extension template. Use the Template Builder to create an application or service extension template. See Oracle WebLogic Server Creating Templates Using the Domain Template Builder.

In the event of an error, the command returns a WLSTException.

3.3.1.2 Syntax
addTemplate(templateFileName)

3.3.1.3 Example
The following example opens a WebLogic domain and extends it using the specified extension template, DefaultWebApp.jar.

wls:/offline> readDomain('c:/Oracle/Middleware/user_projects/domains/wlw')
3.3.2 closeDomain

Command Category: Control Commands
Use with WLST: Offline

3.3.2.1 Description
Closes the current domain. The domain is no longer available for editing once it is closed. In the event of an error, the command returns a WLSTException.

3.3.2.2 Syntax
closeDomain()

3.3.2.3 Example
The following example closes the current domain:

```
wl\s:/offline/wlw> readDomain('c:/Oracle/Middleware/user_projects/domains/medrec')
...
wl\s:/offline/medrec> updateDomain()
wl\s:/offline/medrec> closeDomain()
wl\s:/offline>
```

3.3.3 closeTemplate

Command Category: Control Commands
Use with WLST: Offline

3.3.3.1 Description
Closes the current domain template. The domain template is no longer available once it is closed. In the event of an error, the command returns a WLSTException.

3.3.3.2 Syntax
closeTemplate()

3.3.3.3 Example
The following example opens an existing domain template, performs some operations, and then closes the current domain template.

```
wls:/offline> readTemplate('c:/Oracle/Middleware/wlserver_10.3/common/templates/domains/wls.jar')
...
wl\s:/offline/wls> closeTemplate()
wls:/offline>
```

3.3.4 connect

Command Category: Control Commands
Use with WLST: Online or Offline
3.3.4.1 Description
Connects WLST to a WebLogic Server instance.

Requires you to provide the credentials (user name and password) of a user who has been defined in the active WebLogic security realm. Once you are connected, a collection of security policies determine which configuration attributes you are permitted to view or modify. (See "Default Security Policies for MBeans" in the WebLogic Server MBean Reference.)

You can supply user credentials by doing any of the following:

- Enter the credentials on the command line. This option is recommended only if you are using WLST in interactive mode.
- Enter the credentials on the command line, then use the storeUserConfig command to create a user configuration file that contains your credentials in an encrypted form and a key file that WebLogic Server uses to unencrypt the credentials. On subsequent WLST sessions (or in WLST scripts), supply the name of the user configuration file and key file instead of entering the credentials on the command line. This option is recommended if you use WLST in script mode because it prevents you from storing unencrypted user credentials in your scripts.
- Use the credentials that are stored in the Administration Server's boot.properties file. By default, when you create an Administration Server in development mode, WebLogic Server encrypts the credentials that were used to create the server and stores them in a boot.properties file. When you create an Administration Server in production mode, no boot.properties file is created. If your production domain does not contain a boot.properties file, you can create one manually; see "Creating a Boot Identify File for an Administration Server" in Oracle Fusion Middleware Managing Server Startup and Shutdown for Oracle WebLogic Server.

When you run the connect command, if there is a boot.properties file containing the encrypted username and password for the domain, you do not have to enter the username and password to connect to the Administration Server. You do, however, have to specify the name of the Administration Server in the connect command.

Please note:

- If you run the connect command in a script without specifying the username and password or user configuration file and key file, a WSLTEException occurs. In interactive mode, you are prompted for the username and password.
- Oracle strongly recommends that you connect WLST to the server through the SSL port or administration port. If you do not, the following warning message is displayed:

  Warning: An insecure protocol was used to connect to the server. To ensure on-the-wire security, the SSL port or Admin port should be used instead.

- If you are connecting to a WebLogic Server instance through an SSL listen port on a server that is using the demonstration SSL keys and certificates, invoke WLST using the following command:

  java -Dweblogic.security.SSL.ignoreHostnameVerification=true
  -Dweblogic.security.TrustKeyStore=DemoTrust weblogic.WLST

For more information about invoking WLST, see "Main Steps for Using WLST in Interactive or Script Mode" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.
If you are connecting to a WebLogic Server instance via HTTP, ensure that the *TunnelingEnabled* attribute is set to *true* for the WebLogic Server instance. For more information, see "TunnelingEnabled" in *Oracle Fusion Middleware Oracle WebLogic Server MBean Reference*.

When trying to connect to the WebLogic Server Administration Server from WLST using localhost as the host name, the following message may be displayed if the listen-address attribute of the Administration Server has been restricted to certain IP addresses:

```
javax.naming.CommunicationException [Root exception is
java.net.ConnectException : <t3://HOST:PORT> : Destination unreachable;
nested exception is: java.net.ConnectException: Connection refused; No
available router to destination
```

You can use either of the following workarounds for this issue:

- Check that the listen-address attribute of the Administration Server has been set correctly. For example, in the domain configuration file:

  ```
  <server>
  <name>AdminServer</name>
  <ssl>
    ...
    ...
  </ssl>
  <machine>your_machine</machine>
  <!-- listen-address><your_ip_address></listen-address -->
  </server>
  ```

  - Use the hostname of the Administration Server, instead of localhost, in the WLST connect command.

After successfully connecting to a WebLogic Server instance, all the local variables are initialized.

In the event of an error, the command returns a *WLSTException*.

### 3.3.4.2 Syntax

```java
connect([username, password], [url], [timeout])
connect([userConfigFile, userKeyFile], [url], [timeout])
connect([url], [adminServerName], [timeout])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>Optional. Username of the operator who is connecting WLST to the server. If not specified, WLST processes the command as described above.</td>
</tr>
<tr>
<td>password</td>
<td>Optional. Password of the operator who is connecting WLST to the server. If not specified, WLST processes the command as described above.</td>
</tr>
<tr>
<td>url</td>
<td>Optional. Listen address and listen port of the server instance, specified using the following format: [protocol://]listen-address:listen-port. If not specified, this argument defaults to t3://localhost:7001.</td>
</tr>
</tbody>
</table>
Control Commands

3.3.4.3 Examples

The following example connects WLST to a WebLogic Server instance. In this example, the Administration Server name defaults to AdminServer. Note that a warning is displayed if the SSL or administration port is not used to connect to the server.

```
wls:/offline> connect('weblogic','welcome1','t3://localhost:8001')
Connecting to weblogic server instance running at t3://localhost:8001 as username weblogic...
Successfully connected to Admin Server 'AdminServer' that belongs to domain 'mydomain'.

Warning: An insecure protocol was used to connect to the server. To ensure on-the-wire security, the SSL port or Admin port should be used instead.
wls:/mydomain/serverConfig>
```
The following example connects WLST to a WebLogic Server instance at the specified URL. In this example, the username and password are passed as variables. This example uses a secure protocol.

```wls:/offline>
username = 'weblogic'
```

```wls:/offline>
password = 'welcome1'
```

```wls:/offline>
connect(username,password,'t3s://myhost:8001')
```

Connecting to weblogic server instance running at t3://myhost:8001 as username weblogic...

Successfully connected to Admin Server 'AdminServer' that belongs to domain 'mydomain'.

```wls:/mydomain/serverConfig>
```

The following example connects WLST to a WebLogic Server instance using a user configuration and key file to provide user credentials.

```wls:/offline>
connect(userConfigFile='c:/myfiles/myuserconfigfile.secure',
userKeyFile='c:/myfiles/myuserkeyfile.secure')
```

Connecting to t3://localhost:7001 with userid ... username ...

Successfully connected to Admin Server 'AdminServer' that belongs to domain 'mydomain'.

```wls:/mydomain/serverConfig>
```

The following example shows the prompts that are displayed in interactive mode if you run the command without parameters:

```wls:/offline>
connect()
```

Please enter your username : username
Please enter your password : 
Please enter your server URL [t3://localhost:7001] :

Connecting to t3://localhost:7001 with userid username

### 3.3.5 createDomain

Command Category: Control Commands

Use with WLST: Offline

#### 3.3.5.1 Description

Creates a WebLogic domain using the specified template.

**Note:** The `createDomain` command is similar in functionality to the `unpack` command, as described in *Creating Templates and Domains Using the pack and unpack Commands*.

In the event of an error, the command returns a `WLSTException`.

#### 3.3.5.2 Syntax

```
createDomain(domainTemplate, domainDir, user, password)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>domainTemplate</td>
<td>Name and location of the domain template from which you want to create a domain.</td>
</tr>
</tbody>
</table>
### 3.3.5.3 Example

The following example creates a new WebLogic domain using the Avitek MedRec template and sets the default username to `weblogic` and the password to `welcome1`. The domain is saved to the following directory: `c:/Oracle/Middleware/wlserver_10.3/user_projects/domains/medrec`.

```
shell> createDomain('c:/Oracle/Middleware/wlserver_10.3/common/templates/domains/wls_medrec.jar','c:/Oracle/Middleware/user_projects/domains/medrec','weblogic', 'welcome1')
```

---

### 3.3.6 disconnect

**Command Category:** Control Commands

**Use with WLST:** Online

#### 3.3.6.1 Description

Disconnects WLST from a WebLogic Server instance. The `disconnect` command does not cause WLST to exit the interactive scripting shell; it closes the current WebLogic Server instance connection and resets all the variables while keeping the interactive shell alive.

In the event of an error, the command returns a `WLSTException`.

You can connect to another WebLogic Server instance using the `connect` command, as described in Section 3.3.4, "connect".

#### 3.3.6.2 Syntax

```
disconnect(force)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>force</td>
<td>Optional. Boolean value specifying whether WLST should disconnect without waiting for the active sessions to complete. This argument defaults to <code>false</code>, indicating that all active sessions must complete before disconnect.</td>
</tr>
</tbody>
</table>

#### 3.3.6.3 Example

The following example disconnects from a running server:

```
shell> mydomain/serverConfig> disconnect()
Disconnected from weblogic server: myserver
shell> offline
```
3.3.7 exit

Command Category: Control Commands
Use with WLST: Online or Offline

3.3.7.1 Description
Exits WLST from the user session and closes the scripting shell.
If there is an edit session in progress, WLST prompts you for confirmation. To skip the prompt, set the defaultAnswer argument to y.
By default, WLST calls System.exit(0) for the current WLST JVM when exiting WLST. If you would like the JVM to exit with a different exit code, you can specify a value using the exitCode argument.

Note: When the WLST exit command is issued within an Ant script, it may also exit the execution of the Ant script. It is recommended that when invoking WLST within an Ant script, you fork a new JVM by specifying fork="true".

In the event of an error, the command returns a WLSTException.

3.3.7.2 Syntax
exit([defaultAnswer], [exitcode])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultAnswer</td>
<td>Optional. Default response, if you would prefer not to be prompted at the command line. Valid values are y and n. This argument defaults to null, and WLST prompts you for a response.</td>
</tr>
<tr>
<td>exitcode</td>
<td>Optional. Exit code to set when exiting WLST.</td>
</tr>
</tbody>
</table>

3.3.7.3 Example
The following example disconnects from the user session and closes the scripting shell.

wls:/mydomain/serverConfig> exit()
Exiting WebLogic Scripting Tool ...
c:\>

The following example disconnects from the user session, closes the scripting shell, and sets the error code to 101.

wls:/mydomain/serverConfig> exit(exitcode=101)
Exiting WebLogic Scripting Tool ...
c:\>

3.3.8 readDomain

Command Category: Control Commands
Use with WLST: Offline

3.3.8.1 Description
Opens an existing WebLogic domain for updating.
WLST offline provides read and write access to the configuration data that is persisted in the config directory for the WebLogic domain, or in a domain template JAR created using Template Builder. This data is a collection of XML documents and expresses a hierarchy of management objects.

When you open a template or WebLogic domain, WLST is placed at the root of the configuration hierarchy for that domain, and the prompt is updated to reflect the current location in the configuration hierarchy. For example:

```
wlst:offline/base_domain>
```

For more information, see "Navigating and Interrogating MBeans" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

In the event of an error, the command returns a WLSTException.

### 3.3.8.2 Syntax

```
readDomain(domainDirName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>domainDirName</td>
<td>Name of the WebLogic domain directory that you want to open.</td>
</tr>
</tbody>
</table>

### 3.3.8.3 Example

The following example opens the medrec domain for editing.

```
wls:offline> readDomain('c:/Oracle/Middleware/user_projects/domains/medrec')
wls:offline/medrec>
```

### 3.3.9 readTemplate

Command Category: Control Commands

Use with WLST: Offline

#### 3.3.9.1 Description

Opens an existing domain template for domain creation.

When you open a domain template, WLST is placed into the configuration bean hierarchy for that domain template, and the prompt is updated to reflect the current location in the configuration hierarchy. For example:

```
wls:offline/base_domain>
```

WebLogic Server configuration beans exist within a hierarchical structure. In the WLST file system, the hierarchies correspond to drives; types and instances are directories; attributes and operations are files. WLST traverses the hierarchical structure of configuration beans using commands such as cd, ls, and pwd in a similar way that you would navigate a file system in a UNIX or Windows command shell. After navigating to a configuration bean instance, you interact with the bean using WLST commands. For more information, see "Navigating and Interrogating MBeans" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

**Note:** Using WLST and a domain template, you can only create and access security information when you are creating a new WebLogic domain. When you are updating a WebLogic domain, you cannot access security information through WLST.
In the event of an error, the command returns a WLSTException.

### 3.3.9.2 Syntax

```java
readTemplate(templateFileName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>templateFileName</td>
<td>Name of the JAR file corresponding to the domain template.</td>
</tr>
</tbody>
</table>

### 3.3.9.3 Example

The following example opens the `medrec.jar` domain template for WebLogic domain creation.

```
wlst:/offline> readTemplate('c:/Oracle/Middleware/wlserver_10.3/common/templates/domains/wls_medrec.jar')
wlst:/offline/wls_medrec>
```

### 3.3.10 updateDomain

Command Category: Control Commands

Use with WLST: Offline

#### 3.3.10.1 Description

Updates and saves the current WebLogic domain. The domain continues to be editable after you update and save it.

In the event of an error, the command returns a WLSTException.

#### 3.3.10.2 Syntax

```java
updateDomain()
```

#### 3.3.10.3 Example

The following examples opens the `medrec` domain, performs some operations, and updates and saves the current domain:

```
wls:/offline> readDomain('c:/Oracle/Middleware/user_projects/domains/medrec')
...
wls:/offline/medrec> updateDomain()
```

### 3.3.11 writeDomain

Command Category: Control Commands

Use with WLST: Offline

#### 3.3.11.1 Description

Writes the domain configuration information to the specified directory.

Once you write the WebLogic domain to file system, you can continue to update the domain template object that exists in memory, and reissue the writeDomain command to store the domain configuration to a new or existing file.

By default, when you write a WebLogic domain, the associated applications are written to `WL_HOME/user_projects/applications/domainname`, where `WL_HOME` specifies the WebLogic Server home directory and `domainname` specifies the name of
the WebLogic domain. This directory must be empty; otherwise, WLST displays an error.

When you have finished using the domain template object in memory, close it using the `closeTemplate` command. If you want to edit the WebLogic domain that has been saved to disk, you can open it using the `readDomain` command.

---

**Note:** The name of the WebLogic domain is derived from the name of the domain directory. For example, for a domain saved to `c:/Oracle/Middleware/user_projects/domains/myMedrec`, the domain name is `myMedrec`.

---

Before writing the domain, you must define a password for the default user, if it is not already defined. For example:

```plaintext
cd('/Security/base_domain/User/weblogic')
cmo.setPassword('welcome1')
```

In the event of an error, the command returns a `WLSTException`.

### 3.3.11.2 Syntax

```plaintext
writeDomain(domainDir)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>domainDir</td>
<td>Name of the directory to which you want to write the domain configuration information.</td>
</tr>
</tbody>
</table>

### 3.3.11.3 Example

The following example reads the `medrec.jar` domain templates, performs some operations, and writes the domain configuration information to the `c:/Oracle/Middleware/user_projects/domains/medrec` directory.

```plaintext
wls:/offline> readTemplate('c:/Oracle/Middleware/wlserver_10.3/common/templates/domains/wls.jar')
...
wls:/offline/base_domain> writeDomain('c:/Oracle/Middleware/user_projects/domains/base_domain')
```

### 3.3.12 writeTemplate

Command Category: Control Commands

Use with WLST: Offline

#### 3.3.12.1 Description

Writes the domain configuration information to the specified domain template. You can use the domain configuration template to recreate the WebLogic domain.

Once your write the configuration information to the domain configuration template, you can continue to update the WebLogic domain or domain template object that exists in memory, and reissue the `writeDomain` or `writeTemplate` command to store the domain configuration to a new or existing WebLogic domain or domain template file. For more information, see Section 3.3.11, "writeDomain" or Section 3.3.12, "writeTemplate", respectively.
In the event of an error, the command returns a WLSTException.

Note: The writeTemplate command is similar in functionality to the pack command; see "The pack Command" in Creating Templates and Domains Using the pack and unpack Commands. However, writeTemplate does not support creating a Managed Server template.

3.3.12.2 Syntax
writeTemplate(templateName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>templateName</td>
<td>Name of the domain template to store the domain configuration information.</td>
</tr>
</tbody>
</table>

3.3.12.3 Example
The following example writes the current domain configuration to the domain template named c:/Oracle/Middleware/user_projects/templates/myTemplate.jar.

```
wls:/offline> readDomain('c:/Oracle/Middleware/user_projects/domains/mydomain')
...
wls:/offline/base_domain> writeTemplate('c:/Oracle/Middleware/user_projects/templates/myTemplate.jar')
```

3.4 Customization Commands
Use the WLST customization commands, listed in Table 3–4, to add the command group help and command help that is listed by the WLST help() and help('commandGroup') commands. For more information about adding command help to WLST, see "Adding Integrated Help for Custom Commands" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

Table 3–4 Customization Commands for WLST Configuration

<table>
<thead>
<tr>
<th>This command...</th>
<th>Enables you to...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>addHelpCommandGroup</td>
<td>Adds a new help command group to those shown by the WLST help() command.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>addHelpCommand</td>
<td>Adds new command help for a command to an existing command group. Once added to the group, the command (along with a brief description) is displayed in the command list for the group when you enter the help('commandGroup') command.</td>
<td>Online or Offline</td>
</tr>
</tbody>
</table>

3.4.1 addHelpCommandGroup
Command Category: Customization Commands
Use with WLST: Online or Offline

3.4.1.1 Description
Adds a new command help group to those shown by the WLST help() command, and specifies the resource bundle in which the help information is defined for the group.
3.4.1.2 Syntax

addHelpCommandGroup(commandGroup, resourceBundleName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>commandGroup</td>
<td>Use a unique name for the command group. Do not use a command group name that is already shown by the WLST help() command.</td>
</tr>
<tr>
<td>resourceBundleName</td>
<td>Represents either a class name or property resource file name. The resource bundle contains help text for entries for the command group using a standard pattern. The resource bundle name will be passed to ResourceBundle.getBundle(...). Multiple command groups can use the same resource bundle. The resource bundle must be present in the classpath. See &quot;Adding Integrated Help for Custom Commands&quot; in Oracle Fusion Middleware Oracle WebLogic Scripting Tool for information on how to define the help text for each command group and command. For more information on resourceBundles and localization, refer to <a href="http://download.oracle.com/javase/6/docs/api/java/util/ResourceBundle.html">http://download.oracle.com/javase/6/docs/api/java/util/ResourceBundle.html</a>.</td>
</tr>
</tbody>
</table>

3.4.1.3 Examples

The following example adds the boot command group to the list of groups shown by the help() command, and specifies that the help text is located in the property resource file 'myhelp':

wls:/offline> addHelpCommandGroup('boot','myhelp')

The following example adds the boot command group to the list of groups shown by the help() command, and specifies that the help text is located in the class foo.bar.MyResourceBundleClass:

wls:/offline> addHelpCommandGroup('boot','foo.bar.MyResourceBundleClass')

3.4.2 addHelpCommand

Command Category: Customization Commands

Use with WLST: Online or Offline

3.4.2.1 Description

Adds new command help for a command to an existing command group. Once added to the group, the command (along with a brief description) is displayed in the command list for the group when you enter the help('commandGroup') command. You can also specify whether or not the command is listed by the help('online') and help('offline') commands.

3.4.2.2 Syntax

addHelpCommand(commandName,commandGroup,[offline=false, online=false])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>commandName</td>
<td>The name of the command as defined in the command group specified by commandGroup.</td>
</tr>
<tr>
<td>commandGroup</td>
<td>The commandGroup to which the command belongs.</td>
</tr>
</tbody>
</table>
3.4.2.3 Example

The following example shows how to add the online command `bootDB` to the listing output by the `help('boot')` and `help('online')` commands:

```wls:/offline> addHelpCommand('bootDB','boot',online='true',offline='false')```

3.5 Deployment Commands

Use the WLST deployment commands, listed in Table 3–5, to:

- Deploy, undeploy, and redeploy applications and standalone modules to a WebLogic Server instance.
- Update an existing deployment plan.
- Interrogate the WebLogic Deployment Manager object.
- Start and stop a deployed application.

For more information about deploying applications, see Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server.

<table>
<thead>
<tr>
<th>This command...</th>
<th>Enables you to...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>deploy</td>
<td>Deploy an application to a WebLogic Server instance.</td>
<td>Online</td>
</tr>
<tr>
<td>distributeApplication</td>
<td>Copy the deployment bundle to the specified targets.</td>
<td>Online</td>
</tr>
<tr>
<td>getWLDMS</td>
<td>Return the WebLogic DeploymentManager object.</td>
<td>Online</td>
</tr>
<tr>
<td>listApplications</td>
<td>List all applications that are currently deployed in the WebLogic domain.</td>
<td>Online</td>
</tr>
<tr>
<td>loadApplication</td>
<td>Load an application and deployment plan into memory.</td>
<td>Online and Offline</td>
</tr>
<tr>
<td>redeploy</td>
<td>Redeploy a previously deployed application.</td>
<td>Online</td>
</tr>
<tr>
<td>startApplication</td>
<td>Start an application, making it available to users.</td>
<td>Online</td>
</tr>
<tr>
<td>stopApplication</td>
<td>Stop an application, making it unavailable to users.</td>
<td>Online</td>
</tr>
<tr>
<td>undeploy</td>
<td>Undeploy an application from the specified servers.</td>
<td>Online</td>
</tr>
<tr>
<td>updateApplication</td>
<td>Update an application configuration using a new deployment plan.</td>
<td>Online</td>
</tr>
</tbody>
</table>
3.5.1 deploy

Command Category: Deployment Commands

Use with WLST: Online

3.5.1.1 Description

Deploys an application to a WebLogic Server instance.

The `deploy` command returns a `WLSTProgress` object that you can access to check the status of the command. For more information about the `WLSTProgress` object, see "WLSTProgress Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool. In the event of an error, the command returns a `WLSTException`.

**Note:** If there is an edit session in progress, the `deploy` command does not block user interaction.

3.5.1.2 Syntax

```
deploy(appName, path, [targets], [stageMode], [planPath], [options])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the application or standalone Java EE module to be deployed.</td>
</tr>
<tr>
<td><code>path</code></td>
<td>Name of the application directory, archive file, or root of the exploded archive directory to be deployed.</td>
</tr>
<tr>
<td><code>targets</code></td>
<td>Optional. Comma-separated list of the targets. Each target may be qualified with a Java EE module name (for example, <code>module1@server1</code>) enabling you to deploy different modules of the application archive on different servers. This argument defaults to the server to which WLST is currently connected.</td>
</tr>
<tr>
<td><code>stageMode</code></td>
<td>Optional. Staging mode for the application you are deploying. Valid values are <code>stage</code>, <code>nostage</code>, and <code>external_stage</code>. For information about the staging modes, see &quot;Controlling Deployment File Copying with Staging Modes&quot; in Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server. If you do not specify a stage mode, the default stage mode is used. On the Administration Server, the default stage mode is <code>nostage</code> and on Managed Servers, it is <code>stage</code>.</td>
</tr>
<tr>
<td><code>planPath</code></td>
<td>Optional. Name of the deployment plan file. The filename can be absolute or relative to the application directory. This argument defaults to the <code>plan/plan.xml</code> file in the application directory, if one exists.</td>
</tr>
</tbody>
</table>
### Argument Definition

**options**

Optional. Comma-separated list of deployment options, specified as name-value pairs. Valid options include:

- **altDD**—Location of the alternate application deployment descriptor on the Administration Server.
- **altWlsDD**—Location of the alternate WebLogic application deployment descriptor on the Administration Server.
- **archiveVersion**—Archive version number.
- **block**—Boolean value specifying whether WLST should block user interaction until the command completes. This option defaults to `true`. If set to `false`, WLST returns control to the user after issuing the command; you can query the `WLSTProgress` object to determine the status of the command. If you are importing WLST as a Jython module, as described in "Importing WLST as a Jython Module" in *Oracle Fusion Middleware Oracle WebLogic Scripting Tool*, `block` is always set to `true`.
- **clusterDeploymentTimeout**—Time, in milliseconds, granted for a cluster deployment task on this application.
- **createPlan**—Boolean value indicating that user would like to create a default plan. This option defaults to `false`.
- **defaultSubmoduleTargets**—Boolean value indicating that targeting for qualifying JMS submodules should be derived by the system, see "Using Sub-Module Targeting with JMS Application Modules" in *Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server*. Default value is `true`.
- **deploymentPrincipalName**—String value specifying the principal for deploying the file or archive during server starts (static deployment; it does not effect the current deployment task). Make sure the user exists. This option adds `<deployment-principal-name>` to the `<app-deployment>` element in the `config.xml` file.
- **forceUndeployTimeout**—Force undeployment timeout value.
- **gracefulIgnoreSessions**—Boolean value specifying whether the graceful production to admin mode operation should ignore pending HTTP sessions. This option defaults to `false` and only applies if `gracefulProductionToAdmin` is set to `true`.
- **gracefulProductionToAdmin**—Boolean value specifying whether the production to Admin mode operation should be graceful. This option defaults to `false`.
- **libImplVersion**—Implementation version of the library, if it is not present in the manifest.
- **libraryModule**—Boolean value specifying whether the module is a library module. This option defaults to `false`. 

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Optional. Comma-separated list of deployment options, specified as name-value pairs. Valid options include:</td>
</tr>
<tr>
<td>altDD</td>
<td>Location of the alternate application deployment descriptor on the Administration Server.</td>
</tr>
<tr>
<td>altWlsDD</td>
<td>Location of the alternate WebLogic application deployment descriptor on the Administration Server.</td>
</tr>
<tr>
<td>archiveVersion</td>
<td>Archive version number.</td>
</tr>
<tr>
<td>block</td>
<td>Boolean value specifying whether WLST should block user interaction until the command completes. This option defaults to <code>true</code>. If set to <code>false</code>, WLST returns control to the user after issuing the command; you can query the <code>WLSTProgress</code> object to determine the status of the command. If you are importing WLST as a Jython module, as described in &quot;Importing WLST as a Jython Module&quot; in <em>Oracle Fusion Middleware Oracle WebLogic Scripting Tool</em>, <code>block</code> is always set to <code>true</code>.</td>
</tr>
<tr>
<td>clusterDeploymentTimeout</td>
<td>Time, in milliseconds, granted for a cluster deployment task on this application.</td>
</tr>
<tr>
<td>createPlan</td>
<td>Boolean value indicating that user would like to create a default plan. This option defaults to <code>false</code>.</td>
</tr>
<tr>
<td>defaultSubmoduleTargets</td>
<td>Boolean value indicating that targeting for qualifying JMS submodules should be derived by the system, see &quot;Using Sub-Module Targeting with JMS Application Modules&quot; in <em>Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server</em>. Default value is <code>true</code>.</td>
</tr>
<tr>
<td>deploymentPrincipalName</td>
<td>String value specifying the principal for deploying the file or archive during server starts (static deployment; it does not effect the current deployment task). Make sure the user exists. This option adds <code>&lt;deployment-principal-name&gt;</code> to the <code>&lt;app-deployment&gt;</code> element in the <code>config.xml</code> file.</td>
</tr>
<tr>
<td>forceUndeployTimeout</td>
<td>Force undeployment timeout value.</td>
</tr>
<tr>
<td>gracefulIgnoreSessions</td>
<td>Boolean value specifying whether the graceful production to admin mode operation should ignore pending HTTP sessions. This option defaults to <code>false</code> and only applies if <code>gracefulProductionToAdmin</code> is set to <code>true</code>.</td>
</tr>
<tr>
<td>gracefulProductionToAdmin</td>
<td>Boolean value specifying whether the production to Admin mode operation should be graceful. This option defaults to <code>false</code>.</td>
</tr>
<tr>
<td>libImplVersion</td>
<td>Implementation version of the library, if it is not present in the manifest.</td>
</tr>
<tr>
<td>libraryModule</td>
<td>Boolean value specifying whether the module is a library module. This option defaults to <code>false</code>.</td>
</tr>
</tbody>
</table>
### 3.5.1.3 Example

The following example deploys the `businessApp` application located at `c:/myapps/business`, A default deployment plan is created.

The `deploy` command returns a `WLSTProgress` object that you can access to check the status of the command. The `WLSTProgress` object is captured in a user-defined variable, in this case, `progress`.

```
  wls:/mydomain/serverConfig/Servers> progress= deploy(appName='businessApp', path='c:/myapps/business',createplan='true')
```

The previous example stores the `WLSTProgress` object returned in a user-defined variable, in this case, `progress`. You can then use the `progress` variable to print the status of the `deploy` command. For example:

```
  wls:/mydomain/serverConfig/Servers> progress.printStatus()
  Current Status of your Deployment:
  Deployment command type: deploy
  Deployment State       : completed
  Deployment Message     : null
  wls:/mydomain/serverConfig/Servers>
```
For more information about the WLSTProgress object, see "WLSTProgress Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

The following example deploys the demoApp application in the archive file located at c:/myapps/demos/app/demoApp.ear, targeting the application modules to myserver, and using the deployment plan file located in c:/myapps/demos/app/plan/plan.xml. WLST waits 120,000 ms for the process to complete.

wls:/mydomain/serverConfig/Servers> deploy('demoApp', 'c:/myapps/demos/app/demoApp.ear', targets='myserver', planPath='c:/myapps/demos/app/plan/plan.xml', timeout=120000)

The following example deploys the jmsApp application located at c:/myapps/demos/jmsApp/demo-jms.xml, targeting the application module to a specific target.

wls:/mydomain/serverConfig/Servers> deploy('jmsApp', path='c:/myapps/demos/jmsApps/demo-jms.xml', subModuleTargets='jmsApp@managed1')

The following example shows how to set the application version (appVersion) to a unique identifier to support production (side-by-side) redeployment. This example deploys the demoApp application in the archive file located at c:/myapps/demos/app/demoApp.ear, and sets the application and archive version numbers to the specified values.

wls:/mydomain/serverConfig> deploy('demoApp', 'c:/myapps/demos/app/demoApp.ear', archiveVersion='901-101', appVersion='901-102')

For more information about production redeployment strategies, see "Redeploying Applications in a Production Environment" in Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server.

3.5.2 distributeApplication

Command Category: Deployment Commands

Use with WLST: Online

3.5.2.1 Description

Copies the deployment bundle to the specified targets. The deployment bundle includes module, configuration data, and any additional generated code. The distributeApplication command does not start deployment.

The distributeApplication command returns a WLSTProgress object that you can access to check the status of the command. For more information about the WLSTProgress object, see "WLSTProgress Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool. In the event of an error, the command returns a WLSTException.

3.5.2.2 Syntax

distributeApplication(appPath, [planPath], [targets], [options])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appPath</td>
<td>Name of the archive file or root of the exploded archive directory to be deployed.</td>
</tr>
</tbody>
</table>
### 3.5.2.3 Example
The following example loads the BigApp application located in the `c:/myapps` directory, and stores the `WLSTProgress` object in a user-defined variable, in this case, `progress`.

The following example distributes the `c:/myapps/BigApp` application to the `myserver`, `oamserver1`, and `oamcluster` servers, using the deployment plan defined at `c:/deployment/BigApp/plan.xml`.

```wls
progress=distributeApplication( 'c:/myapps/BigApp', 'c:/deployment/BigApp/plan.xml', 'myserver,oamserver1,oamcluster')
```

Distributing Application and Plan ...
Successfully distributed the application.

The previous example stores the `WLSTProgress` object in a user-defined variable, in this case, `progress`. You can then use the `progress` variable to determine if the `distributeApplication` command has completed. For example:

```wls
progress.isCompleted()
```

1

For more information about the `WLSTProgress` object, see "WLSTProgress Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

---

### 3.5.3 getWLDM

**Command Category:** Deployment Commands

**Use with WLST:** Online

**3.5.3.1 Description**
Returns the WebLogic DeploymentManager object. You can use the object methods to configure and deploy applications. WLST must be connected to an Administration Server to run this command. In the event of an error, the command returns a `WLSTException`.

**3.5.3.2 Syntax**

```java
getWLDM()
```

**3.5.3.3 Example**
The following example gets the `WebLogicDeploymentManager` object and stores it in the `wldm` variable.
3.5.4 listApplications

Command Category: Deployment Commands

Use with WLST: Online

3.5.4.1 Description

Lists all applications that are currently deployed in the WebLogic domain.

In the event of an error, the command returns a WLSTException.

3.5.4.2 Syntax

```
listApplications()
```

3.5.4.3 Example

The following example lists all the applications currently deployed in mydomain.

```
wls:/mydomain/serverConfig> listApplications()
SamplesSearchWebApp
asyncServletEar
jspSimpleTagEar
ejb30
webservicesJwsSimpleEar
ejb20BeanMgedEar
xmlBeanEar
extServletAnnotationsEar
examplesWebApp
apache_xbean.jar
mainWebApp
jdbcRowSetsEar
```

3.5.5 loadApplication

Command Category: Deployment Commands

Use with WLST: Online and Offline

3.5.5.1 Description

Loads an application and deployment plan into memory. When used in online mode, you can connect only to the Administration Server; you cannot connect to a Managed Server.

The loadApplication command returns a WLSTPlan object that you can access to make changes to the deployment plan. For more information about the WLSTPlan object, see "WLSTPlan Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool. In the event of an error, the command returns a WLSTException.

3.5.5.2 Syntax

```
loadApplication(appPath, [planPath], [createPlan])
```
3.5.5.3 Example
The following example loads the c:/myapps/myejb.jar application using the plan file at c:/myplans/myejb/plan.xml.

```wls:offline>
myPlan=loadApplication('c:/myapps/myejb.jar', 'c:/myplans/myejb/plan.xml')
```

Loading application from c:/myapps/myejb.jar and deployment plan from c:/myplans/myejb/plan.xml ...
Successfully loaded the application.

The previous example stores the WLSTPlan object returned in the myPlan variable. You can then use myPlan variable to display information about the plan, such as the variables. For example:

```wls:offline>
myPlan.showVariables()
```

MyEJB jndi.ejb
MyWAR app.foo

For more information about the WLSTPlan object, see "WLSTPlan Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

3.5.6 redeploy

Command Category: Deployment Commands

Use with WLST: Online

3.5.6.1 Description
Reolds classes and redeploy a previously deployed application.

The redeploy command returns a WLSTProgress object that you can access to check the status of the command. For more information about the WLSTProgress object, see "WLSTProgress Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

In the event of an error, the command returns a WLSTException.

For more information about redeploying applications, see "Overview of Common Deployment Scenarios" in Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server.

3.5.6.2 Syntax

```java
redeploy(appName, [planPath], [options])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application to be redeployed.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>planPath</td>
<td>Optional. Name of the deployment plan file. The filename can be absolute or relative to the application directory. This argument defaults to the plan/plan.xml file in the application directory, if one exists.</td>
</tr>
<tr>
<td>createPlan</td>
<td>Optional. Boolean value specifying whether WLST should create a plan in the application directory if the specified plan does not exist. This argument defaults to true.</td>
</tr>
</tbody>
</table>
3.5.6.3 Example

The following example redeploy an application using the plan.xml file located in the c:/myapps directory.

```
wlst:/mydomain/serverConfig> progress=redeploy('myApp' 'c:/myapps/plan.xml')
Redeploying application 'myApp' ...
Redeployment of 'myApp' is successful
wlst:/mydomain/serverConfig>
```

The previous example stores the WLSTProgress object returned in a user-defined variable, in this case, progress. You can then use the progress variable to access the state of the redeploy command. For example:

```
wlst:/mydomain/serverConfig/Servers> progress.getState()
'completed'
wlst:/mydomain/serverConfig/Servers>
```

For more information about the WLSTProgress object, see "WLSTProgress Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>planPath</td>
<td>Optional. Name of the deployment plan file. The filename can be absolute or relative to the application directory. This argument defaults to the plan/plan.xml file in the application directory, if one exists.</td>
</tr>
<tr>
<td>options</td>
<td>Optional. Comma-separated list of deployment options, specified as name-value pairs. For a list of valid deployment options, see options argument description in Section 3.5.1, “deploy”. In addition, the following deployment option can be specified for the redeploy command:</td>
</tr>
<tr>
<td></td>
<td>■ appPath—Name of the archive file or root of the exploded archive directory to be redeployed.</td>
</tr>
<tr>
<td></td>
<td>■ deploymentPrincipalName—String value specifying the principal for redeploying the file or archive during server starts. You can use this option to overwrite the current &lt;deployment-principal-name&gt; in the config.xml file.</td>
</tr>
</tbody>
</table>

3.5.7 startApplication

Command Category: Deployment Commands

Use with WLST: Online

3.5.7.1 Description

Starts an application, making it available to users. The application must be fully configured and available in the WebLogic domain.

The startApplication command returns a WLSTProgress object that you can access to check the status of the command. For more information about the WLSTProgress object, see "WLSTProgress Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool. In the event of an error, the command returns a WLSTException.

3.5.7.2 Syntax

```
startApplication(appName, [options])
```
3.5.7.3 Example
The following example starts the BigApp application with the specified deployment options.

```
wlst:/mydomain/serverConfig/Servers> progress=startApplication('BigApp',
stageMode='NOSTAGE', testMode='false')
Starting the application...
Successfully started the application.
```

The previous example stores the WLSTProgress object returned in a user-defined variable, in this case, progress. You can then use the progress variable to access the state of the startApplication command. For example:

```
wls:/mydomain/serverConfig/Servers> progress.getState()
'completed'
wls:/mydomain/serverConfig/Servers>
```

For more information about the WLSTProgress object, see "WLSTProgress Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

3.5.8 stopApplication

Command Category: Deployment Commands

Use with WLST: Online

3.5.8.1 Description
Stops an application, making it unavailable to users. The application must be fully configured and available in the WebLogic domain.

The stopApplication command returns a WLSTProgress object that you can access to check the status of the command. For more information about the WLSTProgress object, see "WLSTProgress Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

In the event of an error, the command returns a WLSTException.

3.5.8.2 Syntax
```
stopApplication(appName, [options])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application to stop, as specified in the plan.xml file.</td>
</tr>
<tr>
<td>options</td>
<td>Optional. Comma-separated list of deployment options, specified as name-value pairs. For a list of valid deployment options, see options argument description in Section 3.5.1, &quot;deploy&quot;.</td>
</tr>
</tbody>
</table>

3.5.8.3 Example
The following example stops the BigApp application.

```
wls:/mydomain/serverConfig/Servers> stopApplication('BigApp',
stageMode='NOSTAGE', testMode='false')
```

For more information about the WLSTProgress object, see "WLSTProgress Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.
wls:/offline> progress=stopApplication('BigApp')
Stopping the application...
Successfully stopped the application.

The previous example stores the WLSTProgress object returned in a user-defined variable, in this case, progress. You can then use the progress variable to check whether stopApplication command is running. For example:

wls:/mydomain/serverConfig/Servers> progress.isRunning()
0
wls:/mydomain/serverConfig/Servers>

For more information about the WLSTProgress object, see "WLSTProgress Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

3.5.9 undeploy

Command Category: Deployment Commands
Use with WLST: Online

3.5.9.1 Description
Undeploys an application from the specified servers.

The undeploy command returns a WLSTProgress object that you can access to check the status of the command. For more information about the WLSTProgress object, see "WLSTProgress Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool. In the event of an error, the command returns a WLSTException.

For more information about deploying and undeploying applications, see "Overview of Common Deployment Scenarios" in Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server.

3.5.9.2 Syntax
undeploy(appName, [targets], [options])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Deployment name for the deployed application.</td>
</tr>
<tr>
<td>targets</td>
<td>Optional. List of the target servers from which the application will be removed. If not specified, defaults to all current targets.</td>
</tr>
<tr>
<td>options</td>
<td>Optional. Comma-separated list of deployment options, specified as name-value pairs. For a list of valid deployment options, see options argument description in Section 3.5.1, &quot;deploy&quot;.</td>
</tr>
</tbody>
</table>

3.5.9.3 Example
The following example removes the businessApp application from all target servers. WLST waits 60,000 ms for the process to complete.

wls:/mydomain/serverConfig> undeploy('businessApp', timeout=60000)
Undeploying application businessApp ...  
<Jul 20, 2005 9:34:15 AM EDT> <Info> <J2EE Deployment SPI> <BEA-260121>  
<Initiating undeploy operation for application, businessApp [archive: null],  
to AdminServer .>  
Completed the undeployment of Application with status  
Current Status of your Deployment:  
Deployment command type: undeploy
Deployment State : completed
Deployment Message : no message
wls:/mydomain/serverConfig>

3.5.10 updateApplication

Command Category: Deployment Commands

Use with WLST: Online

3.5.10.1 Description

Updates an application configuration using a new deployment plan. The application
must be fully configured and available in the WebLogic domain.

The updateApplication command returns a WLSTProgress object that you can access
to check the status of the command. For more information about the WLSTProgress
object, see "WLSTProgress Object" in Oracle Fusion Middleware Oracle WebLogic Scripting
Tool. In the event of an error, the command returns a WLSTException.

3.5.10.2 Syntax

updateApplication(appName, [planPath], [options])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application, as specified in the current plan.xml file.</td>
</tr>
<tr>
<td>planPath</td>
<td>Optional. Name of the new deployment plan file. The filename can be absolute or relative to the application directory.</td>
</tr>
<tr>
<td>options</td>
<td>Optional. Comma-separated list of deployment options, specified as name-value pairs. For a list of valid deployment options, see options argument description in Section 3.5.1, &quot;deploy&quot;.</td>
</tr>
</tbody>
</table>

3.5.10.3 Example

The following example updates the application configuration for BigApp using the
plan.xml file located in c:/myapps/BigApp/newPlan.

wls:/offline> progress=updateApplication('BigApp',
'c:/myapps/BigApp/newPlan/plan.xml', stageMode='STAGE', testMode='false')
Updating the application...
Successfully updated the application.

The previous example stores the WLSTProgress object returned in a user-defined
variable, in this case, progress. You can then use the progress variable to access the
state of the updateApplication command. For example:

wls:/mydomain/serverConfig/Servers> progress.getState()
'completed'
wls:/mydomain/serverConfig/Servers>

For more information about the WLSTProgress object, see "WLSTProgress Object" in
Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

3.6 Diagnostics Commands

Use the WLST diagnostics commands, listed in Table 3-6, to retrieve diagnostics data
by executing queries against the WebLogic Diagnostics Framework (WLDF) data
stores. For more information about WLDF, see Oracle Fusion Middleware Configuring and Using the Diagnostics Framework for Oracle WebLogic Server.

### Table 3–6 Diagnostic Command for WLST Configuration

<table>
<thead>
<tr>
<th>This command...</th>
<th>Enables you to...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>exportDiagnosticData</td>
<td>Execute a query against the specified log file.</td>
<td>Offline</td>
</tr>
<tr>
<td>exportDiagnosticDataFromServer</td>
<td>Executes a query on the server side and retrieves the exported WebLogic Diagnostic Framework (WLDF) data.</td>
<td>Online</td>
</tr>
<tr>
<td>getAvailableCapturedImages</td>
<td>Returns a list of the previously captured diagnostic images.</td>
<td>Online</td>
</tr>
<tr>
<td>saveDiagnosticImageCapture File</td>
<td>Downloads the specified diagnostic image capture.</td>
<td>Online</td>
</tr>
<tr>
<td>saveDiagnosticImageCapture EntryFile</td>
<td>Downloads a specific entry from the diagnostic image capture.</td>
<td>Online</td>
</tr>
</tbody>
</table>

### 3.6.1 exportDiagnosticData

Command Category: Diagnostics Commands

Use with WLST: Offline

#### 3.6.1.1 Description

Executes a query against the specified log file. The results are saved to an XML file. For more information about the WebLogic Server Diagnostic Service, see Oracle Fusion Middleware Configuring and Using the Diagnostics Framework for Oracle WebLogic Server.

In the event of an error, the command returns a WLSTException.

#### 3.6.1.2 Syntax

```bash
exportDiagnosticData([options])
```
3.6.1.3 Example
The following example executes a query against the ServerLog named myserver.log and stores the results in the file named myExport.xml.

```
wls:/offline/mydomain> exportDiagnosticData(logicalName='ServerLog', logName='myserver.log', exportFileName='myExport.xml')
{'elfFields': '', 'logName': 'myserver.log', 'logRotationDir': '.', 'endTimestamp': 9223372036854775807L, 'exportFileName': 'export.xml', 'storeDir': '../data/store/diagnostics', 'logicalName': 'ServerLog', 'query': '', 'beginTimestamp': 0}
```

Exporting diagnostic data to export.xml

```
Aug 2, 2005 6:58:21 PM EDT <Info> <Store> <BEA-280050> <Persistent store
"WLS_DIAGNOSTICS" opened: directory="c:\Oracle\Middleware\wls\wls\10.3\server\data\store\diagnostics"
writePolicy="Disabled" blockSize=512 directIO=false driver="wlfileio2"
```

3.6.2 exportDiagnosticDataFromServer

Command Category: Diagnostics Commands

Use with WLST: Online

3.6.2.1 Description
Executes a query on the server side and retrieves the exported WebLogic Diagnostic Framework (WLDF) data. The results are saved to an XML file. This command requires that you have a secure connection to the Managed Server.
For more information about the WebLogic Server Diagnostic Service, see *Oracle Fusion Middleware Configuring and Using the Diagnostics Framework for Oracle WebLogic Server*.

In the event of an error, the command returns a `WLSTException`.

### 3.6.2.2 Syntax

```python
exportDiagnosticDataFromServer([options])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Optional. Comma-separated list of export diagnostic options, specified as name-value pairs. Valid options include:</td>
</tr>
<tr>
<td></td>
<td>■ <code>beginTimestamp</code>—Timestamp (inclusive) of the earliest record to be added to the result set. This option defaults to 0.</td>
</tr>
<tr>
<td></td>
<td>■ <code>endTimestamp</code>—Timestamp (exclusive) of the latest record to be added to the result set. This option defaults to <code>Long.MAX_VALUE</code>.</td>
</tr>
<tr>
<td></td>
<td>■ <code>exportFileName</code>—Name of the file to which the data is exported. This option defaults to <code>export.xml</code>.</td>
</tr>
<tr>
<td></td>
<td>■ <code>logicalName</code>—Logical name of the log file being read. Valid values include: <code>HarvestedDataArchive</code>, <code>EventsDataArchive</code>, <code>ServerLog</code>, <code>DomainLog</code>, <code>HTTPAccessLog</code>, <code>WebAppLog</code>, <code>ConnectorLog</code>, and <code>JMSMessageLog</code>. This option defaults to <code>ServerLog</code>.</td>
</tr>
<tr>
<td></td>
<td>■ <code>query</code>—Expression specifying the filter condition for the data records to be included in the result set. This option defaults to &quot;&quot; (empty string), which returns all data.</td>
</tr>
</tbody>
</table>

### 3.6.2.3 Example

The following example executes a query against the `HTTPAccessLog` and stores the results in the file named `myExport.xml`.

```python
wls:/mydomain/serverRuntime>
exportDiagnosticDataFromServer(logicalName="HTTPAccessLog",
exportFileName="myExport.xml")
```

### 3.6.3 `getAvailableCapturedImages`

**Command Category:** Diagnostics Commands

**Use with WLST:** Online

#### 3.6.3.1 Description

Returns, as an array of strings, a list of the previously captured diagnostic images that are stored in the image destination directory configured on the server. The default directory is `SERVER\logs\diagnostic_images`.

This command is useful for identifying a diagnostic image capture that you want to download, or for identifying a diagnostic image capture from which you want to download a specific entry.

For more information about the WebLogic Server Diagnostic Service, see *Oracle Fusion Middleware Configuring and Using the Diagnostics Framework for Oracle WebLogic Server*.

In the event of an error, the command returns a `WLSTException`.

#### 3.6.3.2 Syntax

```python
getAvailableCapturedImages()
```
3.6.3.3 Example
The following example returns an array of strings named `images`, which contains a list of the diagnostic image capture files available in the image destination directory, and prints the entries contained in the diagnostic image named `diagnostic_image_myserver_2009_06_15_14_58_36.zip`.

```
wls:/mydomain/serverRuntime> images=getAvailableCapturedImages()
Connecting to http://localhost:7001 with userid weblogic ...
wls:/mydomain/serverRuntime> print images [ 'diagnostic_image_myserver_2009_06_15_14_58_36.zip' ]
```

3.6.4 `saveDiagnosticImageCaptureFile`
Command Category: Diagnostics Commands
Use with WLST: Online

3.6.4.1 Description
Downloads the specified diagnostic image capture from the server to which WLST is currently connected.

For more information about the WebLogic Server Diagnostic Service, see *Oracle Fusion Middleware Configuring and Using the Diagnostics Framework for Oracle WebLogic Server*.

In the event of an error, the command returns a `WLSTException`.

3.6.4.2 Syntax
```
saveDiagnosticImageCaptureFile(imageName, [outputFile])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>imageName</code></td>
<td>The name of the diagnostic image capture to download.</td>
</tr>
<tr>
<td><code>outputFile</code></td>
<td>Optional. Local path and file name in which the retrieved diagnostic image capture is to be stored. If not specified, this argument defaults to the value of <code>imageName</code> and the current working directory.</td>
</tr>
</tbody>
</table>

3.6.4.3 Example
The following example retrieves the list of the diagnostic image captures that are stored in the image destination directory on the server. It then shows two uses of the `saveDiagnosticImageCaptureFile` command. In the first use, the first diagnostic image capture in the list is downloaded to the local machine using the default output file name. In the second use, the first diagnostic image capture in the list is downloaded to the local machine in the file `mylocalimg.zip`.

```
wls:/mydomain/serverRuntime> images=getAvailableCapturedImages()
Connecting to http://localhost:7001 with userid weblogic ...
wls:/mydomain/serverConfig> saveDiagnosticImageCaptureFile(images[0])
Retrieving diagnostic_image_myserver_2009_06_25_12_12_50.zip to local path diagnostic_image_myserver_2009_06_25_12_12_50.zip
Connecting to http://localhost:7001 with userid weblogic ...
wls:/mydomain/serverConfig> saveDiagnosticImageCaptureFile(images[0], 'mylocalimg.zip')
Retrieving diagnostic_image_myserver_2009_06_25_12_12_50.zip to local path mylocalimg.zip
Connecting to http://localhost:7001 with userid weblogic ...
```
3.6.5 saveDiagnosticImageCaptureEntryFile

Command Category: Diagnostics Commands
Use with WLST: Online

3.6.5.1 Description
Downloads a specific entry from the diagnostic image capture that is located on the server to which WLST is currently connected.

For more information about the WebLogic Server Diagnostic Service, see Oracle Fusion Middleware Configuring and Using the Diagnostics Framework for Oracle WebLogic Server.

In the event of an error, the command returns a WLSTException.

3.6.5.2 Syntax
saveDiagnosticImageCaptureEntryFile(imageName, imageEntryName, [outputFile])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageName</td>
<td>Name of the diagnostic image capture containing the desired entry.</td>
</tr>
<tr>
<td>imageEntryName</td>
<td>Name of the specific entry to be retrieved from the diagnostic image capture. This can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>image.summary</td>
</tr>
<tr>
<td></td>
<td>JTA.img</td>
</tr>
<tr>
<td></td>
<td>JRockitFlightRecorder.jfr</td>
</tr>
<tr>
<td></td>
<td>FlightRecording.jfr</td>
</tr>
<tr>
<td></td>
<td>WatchSource.img</td>
</tr>
<tr>
<td></td>
<td>configuration.img</td>
</tr>
<tr>
<td></td>
<td>WORK_MANAGER.img</td>
</tr>
<tr>
<td></td>
<td>JNDI_IMAGE_SOURCE.img</td>
</tr>
<tr>
<td></td>
<td>APPLICATION.img</td>
</tr>
<tr>
<td></td>
<td>InstrumentationImageSource.img</td>
</tr>
<tr>
<td></td>
<td>SAF.img</td>
</tr>
<tr>
<td></td>
<td>Logging.img</td>
</tr>
<tr>
<td></td>
<td>PERSISTENT_STORE.img</td>
</tr>
<tr>
<td></td>
<td>JDBC.img</td>
</tr>
<tr>
<td></td>
<td>PathService.img</td>
</tr>
<tr>
<td></td>
<td>JMS.img</td>
</tr>
<tr>
<td></td>
<td>Deployment.img</td>
</tr>
<tr>
<td></td>
<td>JVM.img</td>
</tr>
<tr>
<td></td>
<td>CONNECTOR.img</td>
</tr>
<tr>
<td>outputFile</td>
<td>Optional. Local path and file name in which the entry retrieved from the diagnostic image capture is to be stored. If not specified, this argument defaults to the value of imageEntryName and the current working directory.</td>
</tr>
</tbody>
</table>

3.6.5.3 Example
The following example gets the list of diagnostic image captures, then uses the saveDiagnosticImageCaptureEntryFile twice. In the first use, this example retrieves the image summary to the local machine using the default output file name. In the second use, it retrieves the image summary to the local machine in the file myimage.summary.

wls:/mydomain/serverRuntime>images=getAvailableCapturedImages()
Connecting to http://localhost:7001 with userid weblogic ...
wls:/mydomain/serverConfig>saveDiagnosticImageCaptureEntryFile(images[0], 'image.summary')
3.7 Editing Commands

Use the WLST editing commands, listed in Table 3–7, to interrogate and edit configuration beans.

Note: To edit configuration beans, you must be connected to an Administration Server, and you must navigate to the edit tree and start an edit session, as described in Section 3.11.5, "edit" and Section 3.7.17, "startEdit", respectively.

If you connect to a Managed Server, WLST functionality is limited to browsing the configuration bean hierarchy. While you cannot use WLST to change the values of MBeans on Managed Servers, it is possible to use the Management APIs to do so. Oracle recommends that you change only the values of configuration MBeans on the Administration Server. Changing the values of MBeans on Managed Servers can lead to an inconsistent domain configuration.

For more information about editing configuration beans, see "Using WLST Online to Update an Existing Domain" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

| Table 3–7  Editing Commands for WLST Configuration |
| --- | --- | --- |
| This command... | Enables you to... | Use with WLST... |
| activate | Activate changes saved during the current editing session but not yet deployed. | Online or Offline |
| assign | Assign resources to one or more destinations. | Offline |
| cancelEdit | Cancel an edit session, release the edit lock, and discard all unsaved changes. This operation can be called by any user with administrator privileges, even if the user did not start the edit session. | Online |
| create | Create a configuration bean of the specified type for the current bean. | Online or Offline |
| delete | Delete an instance of a configuration for the current configuration bean. | Online or Offline |
| encrypt | Encrypt the specified string. | Online |
| get | Return the value of the specified attribute. | Online or Offline |
| getActivationTask | Return the latest ActivationTask MBean on which a user can get status. | Online |
| invoke | Invokes a management operation on the current configuration bean. | Online |
3.7.1 activate

Command Category: Editing Commands

Use with WLST: Online

3.7.1.1 Description

Activates changes saved during the current editing session but not yet deployed. This command prints a message if a server restart is required for the changes that are being activated.

The activate command returns the latest ActivationTask MBean which reflects the state of changes that a user is currently making or has made recently. You can then invoke methods to get information about the latest Configuration Manager activate task in progress or just completed. In the event of an error, the command returns a WLSTException.

3.7.1.2 Syntax

activate([timeout], [block])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>timeout</td>
<td>Optional. Time (in milliseconds) that WLST waits for the activation of configuration changes to complete before canceling the operation. A value of -1 indicates that the operation will not time out. This argument defaults to 300,000 ms (or 5 minutes).</td>
</tr>
</tbody>
</table>
### 3.7.3 Example

The following example activates the changes made during the current edit session that have been saved to disk, but that have not yet been activated. WLST waits for 100,000 ms for the activation to complete, and 200,000 ms before the activation is stopped.

```
wlst:/mydomain/edit !> activate(200000, block='true')
Activating all your changes, this may take a while ...
The edit lock associated with this edit session is released once the activation is completed.
Action completed.
wls:/mydomain/edit>
```

### 3.7.2 assign

Command Category:Editing Commands

Use with WLST: Offline

#### 3.7.2.1 Description

Assigns resources to one or more destinations.

In the event of an error, the command returns a **WLSTException**.

#### 3.7.2.2 Syntax

```
assign(sourceType, sourceName, destinationType, destinationName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>sourceType</strong></td>
<td>Type of configuration bean to be assigned. This value can be set to one of the following values:</td>
</tr>
<tr>
<td></td>
<td>- AppDeployment</td>
</tr>
<tr>
<td></td>
<td>- Library</td>
</tr>
<tr>
<td></td>
<td>- securityType (such as User)</td>
</tr>
<tr>
<td></td>
<td>- Server</td>
</tr>
<tr>
<td></td>
<td>- service (such as JDBCSystemResource)</td>
</tr>
<tr>
<td></td>
<td>- service.SubDeployment, where service specifies the service type of the SubDeployment (such as JMSSystemResource.SubDeployment); you can also specify nested subdeployments (such as AppDeployment.SubDeployment.SubDeployment)</td>
</tr>
</tbody>
</table>

Guidelines for setting this value are provided below.
Use the following guidelines for setting the `sourceType` and `destinationType`:

- When assigning application deployments, set the values as follows:
  - `sourceType`: AppDeployment
  - `destinationType`: Target

- When assigning libraries, set the values as follows:
  - `sourceType`: Library
  - `destinationType`: Target

- When assigning services, set the values as follows:
  - `sourceType`: Name of the specific server, such as `JDBCSystemResource`
  - `destinationType`: Target

- When assigning servers to clusters, set the values as follows:
  - `sourceType`: Server
  - `destinationType`: Cluster

- When assigning subdeployments, set the values as follows:
  - `sourceType`: service.SubDeployment, where `service` specifies the parent of the SubDeployment, such as `JMSSystemResource`. You can also specify nested subdeployments (such as `AppDeployment.SubDeployment.SubDeployment`).
  - `destinationType`: Target

- When assigning security types, set the values as follows:
  - `sourceType`: Name of the security type, such as `User`
  - `destinationType`: Name of the destination security type, such as `Group`

### 3.7.2.3 Example

The following examples:

- Assign the servers `myServer` and `myServer2` to the cluster `myCluster`.
  
  ```bash
  wls:/offline/mydomain> assign("Server", "myServer,myServer2", "Cluster",
  ```
"myCluster"

- Assign all servers to the cluster myCluster.
  
  ```
  wls:/offline/mydomain> assign("Server", "*", "Cluster", "myCluster")
  ```

- Assign the application deployment myAppDeployment to the target server newServer.
  
  ```
  wls:/offline/mydomain> assign("AppDeployment", "myAppDeployment", "Target", "newServer")
  ```

- Assign the user newUser to the group Monitors.
  
  ```
  wls:/offline/mydomain> assign("User", "newUser", "Group", "Monitors")
  ```

- Assign the SubDeployment myQueueSubDeployment, which is a child of the JMS resource myJMSResource, to the target server newServer.
  
  ```
  wls:/offline/mydomain> assign('JMSSystemResource.SubDeployment', 'myJMSResource.myQueueSubDeployment', 'Target', 'newServer')
  ```

- Assign the nested SubDeployment MedRecAppScopedJMS.MedRecJMSServer, which is a child of the AppDeployment AppDeployment, to the target server AdminServer.
  
  ```
  ```

### 3.7.3 cancelEdit

**Command Category:** Editing Commands

**Use with WLST:** Online

**3.7.3.1 Description**

Cancels an edit session, releases the edit lock, and discards all unsaved changes.

The user issuing this command does not have to be the current editor; this allows an administrator to cancel an edit session, if necessary, to enable other users to start an edit session.

In the event of an error, the command returns a `WLSTException`.

**3.7.3.2 Syntax**

```
cancelEdit([defaultAnswer])
```

**Argument** | **Definition**
---|---
`defaultAnswer` | Optional. Default response, if you would prefer not to be prompted at the command line. Valid values are `y` and `n`. This argument defaults to `null`, and WLST prompts you for a response.

**3.7.3.3 Example**

The following example cancels the current editing session. WLST prompts for verification before canceling.

```
wls:/mydomain/edit !> cancelEdit()
Sure you would like to cancel the edit session? (y/n)y
Edit session is cancelled successfully
wls:/mydomain/edit>
```
3.7.4 create

Command Category: Editing Commands

Use with WLST: Online or Offline

3.7.4.1 Description

Creates a configuration bean of the specified type for the current bean.

The create command returns a stub for the newly created configuration bean. In the event of an error, the command returns a WLSTException.

Note: Child types must be created under an instance of their parent type. You can only create configuration beans that are children of the current Configuration Management Object (cmo) type. For more information about the cmo variable, see "Changing the Current Management Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

Please note the following when using the create command with WLST online:

- You must be connected to an Administration Server. You cannot use the create command for runtime MBeans or when WLST is connected to a Managed Server instance.
- You must navigate to the edit configuration MBean hierarchy using the edit command before issuing this command. See Section 3.11.5, "edit".
- You can use the create command to create a WebLogic Server configuration MBean that is a child of the current MBean type.

Please note the following when using the create command with WLST offline:

- When using WLST offline, the following characters are not valid in object names: period (.), forward slash (/), or backward slash (\).

For more information about:

- Creating MBeans, see "Understanding WebLogic Server MBeans" in Developing Custom Management Utilities with JMX.
- Examples of creating specific types of MBean resources, for example, a JMS or JDBC system resource, refer to the WLST sample scripts installed with your product, as described in "WLST Sample Scripts" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.
- MBeans, their child types, attributes, and operations, see Oracle Fusion Middleware Oracle WebLogic Server MBean Reference.

3.7.4.2 Syntax

create(name, childMBeanType, [baseProviderType])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the configuration bean that you are creating.</td>
</tr>
<tr>
<td>childMBeanType</td>
<td>Type of configuration bean that you are creating. You can create instances of any type defined in the config.xml file except custom security types. For more information about valid configuration beans, see Oracle Fusion Middleware Oracle WebLogic Server MBean Reference.</td>
</tr>
</tbody>
</table>
### 3.7.4.3 Example

The following example creates a child configuration bean of type `Server` named `newServer` for the current configuration bean, storing the stub as `server1`:

```wls:
server1=create('newServer', 'Server')
Server with name 'newServer' has been created successfully.
server1.getName()
'newServer'
```

The following example creates an authentication provider security provider called `myProvider`:

```wls:
cd('SecurityConfiguration/mydomain/Realms/myrealm')
create('myProvider', 'weblogic.security.providers.authentication.SQLAuthenticator', 'AuthenticationProvider')
cd('AuthenticationProviders/myProvider')
set('ControlFlag', 'REQUIRED')
```

The following example creates a machine named `highsec_nm` and sets attributes for the associated Node Manager.

```wls:
create('highsec_nm', 'Machine')
cd('Machine/highsec_nm/NodeManager/highsec_nm')
set('DebugEnabled', 'true')
set('ListenAddress', 'innes')
set('NMType', 'SSL')
set('ShellCommand', '')
```

### 3.7.5 delete

**Command Category:** Editing Commands

**Use with WLST:** Online or Offline

#### 3.7.5.1 Description

Deletes an instance of a configuration bean of the specified type for the current configuration bean.

In the event of an error, the command returns a `WLSTException`.

**Note:** You can only delete configuration beans that are children of current Configuration Management Object (cmo) type. For more information about the `cmo` variable, see "Changing the Current Management Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

#### 3.7.5.2 Syntax

```wls:
delete(name, childMBeanType)
```
3.7.5.3 Example
The following example deletes the configuration bean of type Server named newServer:

```
wlst:/mydomain/edit !> delete('newServer','Server')
Server with name 'newServer' has been deleted successfully.
wlst:/mydomain/edit !>
```

3.7.6 encrypt

Command Category: Editing Commands
Use with WLST: Online

3.7.6.1 Description
Encrypts the specified string. You can then use the encrypted string in your configuration file or as an argument to a command.

You must invoke this command once for each WebLogic domain in which you want to use the encrypted string. The string can be used only in the WebLogic domain for which it was originally encrypted.

In the event of an error, the command returns a WLSTException.

3.7.6.2 Syntax
```
encrypt(obj, [domainDir])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>obj</td>
<td>String that you want to encrypt.</td>
</tr>
<tr>
<td>domainDir</td>
<td>Optional. Absolute path name of a WebLogic domain directory. The encrypted</td>
</tr>
<tr>
<td></td>
<td>string can be used only by the WebLogic domain that is contained within the</td>
</tr>
<tr>
<td></td>
<td>specified directory. If you do not specify this argument, the command</td>
</tr>
<tr>
<td></td>
<td>encrypts the string for use in the WebLogic domain to which WLST is currently connected.</td>
</tr>
</tbody>
</table>

3.7.6.3 Example
The following example encrypts the specified string using the `security/SerializedSystemIni.dat` file in the specified WebLogic domain directory.

```
wlst:/mydomain/serverConfig>
es=encrypt('myPassword','c:/Oracle/Middleware/domains/mydomain')
```

3.7.7 get

Command Category: Editing Commands
Use with WLST: Online or Offline
3.7.7.1 Description
Returns the value of the specified attribute. For more information about the MBean attributes that can be viewed, see Oracle WebLogic Server MBean Reference. In the event of an error, the command returns a WLSTException.

Note: You can list all attributes and their current values by entering `ls('a')`. For more information, see Section 3.8.12, "ls".

Alternatively, you can use the cmo variable to perform any get method on the current configuration bean. For example:

```java
cmo.getListenPort()
```

For more information about the cmo variable, see "Changing the Current Management Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

3.7.7.2 Syntax
`get(attrName)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>attrName</td>
<td>Name of the attribute to be displayed. You can specify the full pathname of the attribute. If no pathname is specified, the attribute is displayed for the current configuration object.</td>
</tr>
</tbody>
</table>

3.7.7.3 Example
The following example returns the value of the AdministrationPort for the current configuration bean.

```java
wls:/mydomain/serverConfig> get('AdministrationPort')
9002
```

Alternatively, you can use the cmo variable:

```java
cmo.getAdministrationPort()
```

3.7.8 getActivationTask

Command Category: Editing Commands

Use with WLST: Online

3.7.8.1 Description
Return the latest ActivationTask MBean on which a user can get status. The ActivationTask MBean reflects the state of changes that a user has made recently in WLST. You can then invoke methods to get information about the latest Configuration Manager activate task in progress or just completed. In the event of an error, the command returns a WLSTException.

Note: If you have activated changes outside of WLST, use the ConfigurationManagerMBean getActivationTasks() method to get access to Activation Tasks created in other tools.
3.7.8.2 Syntax
getActivationTask()

3.7.8.3 Example
The following example returns the latest ActivationTask MBean on which a user can get status and stores it within the task variable.

wls:/mydomain/edit> task=getActivationTask()
wls:/mydomain/edit> if task!=None:
...   task.getState()
...

3.7.9 invoke

Command Category: Editing Commands
Use with WLST: Online

3.7.9.1 Description
Invokes a management operation on the current configuration bean. Typically, you use this command to invoke operations other than the get and set operations that most WebLogic Server configuration beans provide. The class objects are loaded through the same class loader that is used for loading the configuration bean on which the action is invoked.

You cannot use the invoke command when WLST is connected to a Managed Server instance.

If successful, the invoke command returns the object that is returned by the operation invoked. In the event of an error, the command returns a WLSTException.

3.7.9.2 Syntax
invoke(methodName, parameters, signatures)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>methodName</td>
<td>Name of the method to be invoked.</td>
</tr>
<tr>
<td>parameters</td>
<td>An array of parameters to be passed to the method call.</td>
</tr>
<tr>
<td>signatures</td>
<td>An array containing the signature of the action.</td>
</tr>
</tbody>
</table>

3.7.9.3 Example
The following example invokes the lookupServer method on the current configuration bean.

wls:/mydomain/config> objs = jarray.array([java.lang.String("oamserver")],java.lang.Object)
wls:/mydomain/edit> strs = jarray.array(["java.lang.String"],java.lang.String)
wls:/mydomain/edit> invoke('lookupServer',objs,strs)
true
wls:/mydomain/edit>

3.7.10 isRestartRequired

Command Category: Editing Commands
Use with WLST: Online

3.7.10.1 Description
Determines whether a server restart is required.

If you invoke this command while an edit session is in progress, the response is based on the edits that are currently in progress. If you specify the name of an attribute, WLST indicates whether a server restart is required for that attribute only.

In the event of an error, the command returns a WLSTException.

3.7.10.2 Syntax
isRestartRequired([attributeName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>attributeName</td>
<td>Optional. Name of a specific attribute for which you want to check if a server restart is required.</td>
</tr>
</tbody>
</table>

3.7.10.3 Example
The following example specifies whether a server restart is required for all changes made during the current WLST session.

```
wls:/mydomain/edit !> isRestartRequired()
Server re-start is REQUIRED for the set of changes in progress.
```

The following attribute(s) have been changed on MBeans that require server re-start.
MBean Changed : mydomain:Name=mydomain,Type=Domain
Attributes changed : AutoConfigurationSaveEnabled

The following example specifies whether a server restart is required if you edit the ConsoleEnabled attribute.

```
wls:/mydomain/edit !> isRestartRequired("ConsoleEnabled")
Server re-start is REQUIRED if you change the attribute ConsoleEnabled
wls:/mydomain/edit !>
```

3.7.11 loadDB
Command Category: Editing Commands
Use with WLST: Offline

3.7.11.1 Description
 Loads SQL files into a database.

The loadDB command loads the SQL files from a template file. This command can only be issued after a domain template or extension template has been loaded into memory (see Section 3.3.8, "readDomain" and Section 3.3.9, "readTemplate").

Before executing this command, ensure that the following conditions are true:

- The appropriate database is running.
- SQL files exist for the specified database and version.
To verify that the appropriate SQL files exist, open the domain template and locate
the relevant SQL file list, jdbc.index, in the _jdbc_ directory. For example, for
Oracle 9i, the SQL file list is located at _jdbc_\Oracle\9i\jdbc.index.

The command fails if the above conditions are not met.
In the event of an error, the command returns a WLSTException.

### 3.7.11.2 Syntax

loadDB(dbVersion, datasourceName, dbCategory)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbVersion</td>
<td>Version of the database for which the SQL files are intended to be used.</td>
</tr>
<tr>
<td>datasourceName</td>
<td>Name of the JDBC data source to be used to load SQL files.</td>
</tr>
<tr>
<td>dbCategory</td>
<td>Optional. Database category associated with the specified data source. For more information about the jdbc.index file and database categories, see &quot;Files Typically Included in a Template&quot; in the Oracle WebLogic Server Domain Template Reference.</td>
</tr>
</tbody>
</table>

### 3.7.11.3 Example

The following example loads SQL files related to Drop/Create P13N Database Objects intended for version 5.1 of the database, using the p13nDataSource JDBC data source.

wls:/offline/mydomain> loadDB('5.1', 'p13nDataSource', 'Drop/Create P13N Database Objects')

### 3.7.12 loadProperties

Command Category: Editing Commands
Use with WLST: Online and Offline

### 3.7.12.1 Description

Loads property values from a file and makes them available in the WLST session.

This command cannot be used when you are importing WLST as a Jython module, as described in "Importing WLST as a Jython Module" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

In the event of an error, the command returns a WLSTException.

### 3.7.12.2 Syntax

loadProperties(fileName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>fileName</td>
<td>Properties file pathname.</td>
</tr>
</tbody>
</table>

### 3.7.12.3 Example

This example gets and sets the properties file values.

wls:/mydomain/serverConfig> loadProperties('c:/temp/myLoad.properties')
3.7.13 save

Command Category: Editing Commands
Use with WLST: Online

3.7.13.1 Description
Saves the edits that have been made but have not yet been saved. This command is only valid when an edit session is in progress. For information about starting an edit session, see Section 3.7.17, "startEdit".

In the event of an error, the command returns a WLSTException.

3.7.13.2 Syntax
save()

3.7.13.3 Example
The following example saves the edits that have not yet been saved to disk.

```
wls:/mydomain/edit !> save()
Saving all your changes ...
Saved all your changes successfully.
wls:/mydomain/edit !>
```

3.7.14 set

Command Category: Editing Commands
Use with WLST: Online or Offline

3.7.14.1 Description
Sets the value of a specified attribute in the current management object. When using WLST offline, this command writes the attribute value to the domain configuration files. When using WLST online, this command sets the value of an MBean attribute. Online changes are written to the domain configuration file when you activate your edits.

In the event of an error, the command returns a WLSTException.

For information about setting encrypted attributes (all encrypted attributes have names that end with Encrypted), see "Writing and Reading Encrypted Configuration Values" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

Note the following when using WLST online:

- You must be in an edit session to use this command. See Section 3.7.17, "startEdit".
- You cannot use this command when WLST is connected to a Managed Server.
- As an alternative to this command, you can use the cmo variable with the following syntax:

  cmo.setAttribute(value)

  For example, instead of using set('ListenPort', 7011), you can use:
  cmo.setListenPort(7011)

  For more information about the cmo variable, see "Changing the Current Management Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.
3.7.14.2 Syntax

set(attrName, value)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>attrName</td>
<td>Name of the attribute to be set.</td>
</tr>
<tr>
<td>value</td>
<td>Value of the attribute to be set.</td>
</tr>
</tbody>
</table>

Note: This value should not be enclosed in single or double quotes. See the examples.

3.7.14.3 Example

The following example sets the ArchiveConfigurationCount attribute of DomainMBean to 10:

wls:/mydomain/serverConfig> set('ArchiveConfigurationCount', 10)

The following example sets the long value of the T1TimerInterval attribute of a custom Mbean to 123:

wls:/mydomain/serverConfig> set('T1TimerInterval', Long(123))

The following example sets the boolean value of the MyBooleanAttribute attribute of a custom Mbean to true:

wls:/mydomain/serverConfig> set('MyBooleanAttribute', Boolean(true))

3.7.15 setOption

Command Category: Editing Commands
Use with WLST: Offline

3.7.15.1 Description
Sets options related to a WebLogic domain creation or update. In the event of an error, the command returns a WLSTException.

3.7.15.2 Syntax

setOption(optionName, optionValue)
### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>optionName</td>
<td>Name of the option to set.</td>
</tr>
</tbody>
</table>

#### Available options for domain creation include:

- **CreateStartMenu**—Boolean value specifying whether to create a Start Menu shortcut on a Windows platform. This option defaults to `true`.
  
  **Note:** If a user with Administrator privileges installed the software and chose to create the Start menu entries in the All Users folder, only users with Administrator privileges can create Start menu entries in the same folder when creating a WebLogic domain using the Configuration Wizard or WLST. That is, if a user without Administrator privileges uses the Configuration Wizard or WLST from this installation to create domains, Start menu shortcuts to the domains are not created. In this case, the users can manually create shortcuts in their local Start menu folder, if desired.

- **DomainName**—Name of the WebLogic domain. By default, the name of the WebLogic domain is derived from the name of the domain directory. For example, for a WebLogic domain saved to `c:/Oracle/Middleware/user_projects/domains/myMedrec`, the domain name is `myMedrec`. By setting `DomainName`, the name of the created domain will be independent of the domain directory name.

- **JavaHome**—Home directory for the JVM to be used when starting the server. The default for this option depends on the platform on which you install WebLogic Server.

- **OverwriteDomain**—Boolean value specifying whether to allow an existing WebLogic domain to be overwritten. This option defaults to `false`.

- **ServerStartMode**—Mode to use when starting the server for the newly created WebLogic domain. This value can be `dev` (development) or `prod` (production). This option defaults to `dev`.

#### Available options for domain updates include:

- **AllowCasualUpdate**—Boolean value specifying whether to allow a WebLogic domain to be updated without adding an extension template. This option defaults to `true`.

- **ReplaceDuplicates**—Boolean value specifying whether to keep original configuration elements in the WebLogic domain or replace the elements with corresponding ones from an extension template when there is a conflict. This option defaults to `true`.

#### Available options for both domain creation and domain updates include:

- **AppDir**—Application directory to be used when a separate directory is desired for applications, as specified by the template. This option defaults to `WL_HOME/user_projects/applications/domainname`, where `WL_HOME` specifies the WebLogic Server home directory and `domainname` specifies the name of the WebLogic domain.

- **AutoAdjustSubDeploymentTarget**—Boolean value specifying whether WLST automatically adjusts targets for the subdeployments of AppDeployments. This option defaults to `true`. To deactivate this feature, set the option to `false` and explicitly set the targeting for AppDeployment subdeployments before writing or updating the WebLogic domain or domain template.

- **AutoDeploy**—Boolean value specifying whether to activate auto deployment when a cluster or multiple Managed Servers are created. This option defaults to `true`. To deactivate this feature, set the option to `false` on the first line of your script.

<table>
<thead>
<tr>
<th>optionValue</th>
<th>Value for the option.</th>
</tr>
</thead>
</table>

**Note:** Boolean values can be specified as a String (`true`, `false`) or integer (0, 1).
3.7.15.3 Example
The following example sets the CreateStartMenu option to false:

```
wlst:/offline> setOption('CreateStartMenu', 'false')
```

### 3.7.16 showChanges

**Command Category:** Editing Commands

**Use with WLST:** Online

#### 3.7.16.1 Description
Shows the changes made to the configuration by the current user during the current edit session. In the event of an error, the command returns a `WLSTException`.

#### 3.7.16.2 Syntax

```
showChanges([onlyInMemory])
```

#### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>onlyInMemory</td>
<td>Optional. Boolean value specifying whether to display only the changes that have not yet been saved. This argument defaults to <code>false</code>, indicating that all changes that have been made from the start of the session are displayed.</td>
</tr>
</tbody>
</table>

#### 3.7.16.3 Example

The following example shows all of the changes made by the current user to the configuration since the start of the current edit session.

```
wls:/mydomain/edit !> showChanges()
```

Changes that are in memory and saved to disc but not yet activated are:

```
MBean Changed : com.bea:Name=basicWLSDomain,Type=Domain
Operation Invoked : add
Attribute Modified : Machines
Attributes Old Value : null
Attributes New Value : Mach1
Server Restart Required : false
```

```
MBean Changed : com.bea:Name=basicWLSDomain,Type=Domain
Operation Invoked : add
Attribute Modified : Servers
Attributes Old Value : null
Attributes New Value : myserver
Server Restart Required : false
```

### 3.7.17 startEdit

**Command Category:** Editing Commands

**Use with WLST:** Online

#### 3.7.17.1 Description

Starts a configuration edit session on behalf of the currently connected user. You must navigate to the edit configuration MBean hierarchy using the `edit` command before issuing this command. For more information, see Section 3.11.5, "edit".
This command must be called prior to invoking any command to modify the WebLogic domain configuration.

In the event of an error, the command returns a WLSTException.

**Note:** WLST automatically starts an edit session if it detects that there is an edit session that is already in progress by the same user, which may have been started via the Administration Console or another WLST session.

### 3.7.17.2 Syntax

```
startEdit([waitTimeInMillis], [timeoutInMillis], [exclusive])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>waitTimeInMillis</td>
<td>Optional. Time (in milliseconds) that WLST waits until it gets a lock, in the event that another user has a lock. This argument defaults to 0 ms.</td>
</tr>
<tr>
<td>timeoutInMillis</td>
<td>Optional. Timeout (in milliseconds) that WLST waits to release the edit lock. This argument defaults to -1 ms, indicating that this edit session never expires.</td>
</tr>
<tr>
<td>exclusive</td>
<td>Optional. Specifies whether the edit session should be an exclusive session. If set to true, if the same owner enters the startEdit command, WLST waits until the current edit session lock is released before starting the new edit session. The exclusive lock times out according to the time specified in timeoutInMillis. This argument defaults to false.</td>
</tr>
</tbody>
</table>

### 3.7.17.3 Example

The following example starts an edit session with a 1-minute timeout waiting for a lock and a 2-minute timeout waiting to release the edit lock:

```
wls:/mydomain/edit> startEdit(60000, 120000)
Starting an edit session ...
Started edit session, please be sure to save and activate your changes once you are done.
wls:/mydomain/edit !>
```

### 3.7.18 stopEdit

Command Category: Editing Commands

Use with WLST: Online

#### 3.7.18.1 Description

 Stops the current edit session, releases the edit lock, and discards unsaved changes.

In the event of an error, the command returns a WLSTException.

#### 3.7.18.2 Syntax

```
stopEdit([defaultAnswer])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultAnswer</td>
<td>Optional. Default response, if you would prefer not to be prompted at the command line. Valid values are y and n. This argument defaults to null, and WLST prompts you for a response.</td>
</tr>
</tbody>
</table>
3.7.18.3 Example
The following example stops the current editing session. WLST prompts for verification before canceling.

```bash
wls:/mydomain/edit !> stopEdit()
Sure you would like to stop your edit session? (y/n)
Y
Edit session has been stopped successfully.
wls:/mydomain/edit>
```

3.7.19 unassign

Command Category: Editing Commands

Use with WLST: Offline

3.7.19.1 Description
Unassign applications or resources from one or more destinations.

In the event of an error, the command returns a WLSTException.

3.7.19.2 Syntax

```bash
unassign(sourceType, sourceName, destinationType, destinationName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sourceType</code></td>
<td>Type of configuration bean to be unassigned. This value can be set to one of the following values:</td>
</tr>
<tr>
<td></td>
<td>■ AppDeployment</td>
</tr>
<tr>
<td></td>
<td>■ Library</td>
</tr>
<tr>
<td></td>
<td>■ <code>securityType</code> (such as User)</td>
</tr>
<tr>
<td></td>
<td>■ Server</td>
</tr>
<tr>
<td></td>
<td>■ <code>service</code> (such as JDBCSystemResource)</td>
</tr>
<tr>
<td></td>
<td>■ <code>service.SubDeployment</code>, where <code>service</code> specifies the service type of the SubDeployment (such as JMSSystemResource.SubDeployment); you can also specify nested subdeployments (such as AppDeployment.SubDeployment.SubDeployment)</td>
</tr>
<tr>
<td><code>sourceName</code></td>
<td>Name of the application or resource to be unassigned. Multiple names can be specified, separated by commas, or you can use the wildcard (*) character to specify all resources of the specified type. Specify subdeployments using the following format: <code>service.subDeployment</code>, where <code>service</code> specifies the parent service and <code>subDeployment</code> specifies the name of the subdeployment. For example, myJMSResource.myQueueSubDeployment. You can also specify nested subdeployments, such as MedRecEAR.MedRecAppScopedJMS.MedRecJMSServer.</td>
</tr>
<tr>
<td><code>destinationType</code></td>
<td>Type of destination. Guidelines for setting this value are provided below.</td>
</tr>
<tr>
<td><code>destinationName</code></td>
<td>Name of the destination. Multiple names can be specified, separated by commas.</td>
</tr>
</tbody>
</table>

Use the following guidelines for setting the `sourceType` and `destinationType`:

- When unassigning application deployments, set the values as follows:
Editing Commands

- `sourceType`: AppDeployment
- `destinationType`: Target

■ When unassigning **libraries**, set the values as follows:
  - `sourceType`: Library
  - `destinationType`: Target

■ When unassigning **security types**, set the values as follows:
  - `sourceType`: Name of the security type, such as User
  - `destinationType`: Name of the destination security type, such as Group

■ When unassigning **servers** from **clusters**, set the values as follows:
  - `sourceType`: Server
  - `destinationType`: Cluster

■ When unassigning **services**, set the values as follows:
  - `sourceType`: Name of the specific server, such as JDBCSystemResource
  - `destinationType`: Target

■ When unassigning **subdeployments**, set the values as follows:
  - `sourceType`: service.SubDeployment, where `service` specifies the parent of the SubDeployment, such as JMSSystemResource.SubDeployment; you can also specify nested subdeployments (such as AppDeployment.SubDeployment.SubDeployment)
  - `destinationType`: Target

### 3.7.19.3 Example

The following examples:

■ Unassign the servers **myServer** and **myServer2** from the cluster **myCluster**.

```bash
wls:/offline/medrec> unassign("Server", "myServer,myServer2", "Cluster", "myCluster")
```

■ Unassign all servers from the cluster **myCluster**.

```bash
wls:/offline/mydomain> unassign("Server", ",", "Cluster", "myCluster")
```

■ Unassign the user **newUser** from the group **Monitors**.

```bash
wls:/offline/medrec> unassign("User", "newUser", "Group", "Monitors")
```

■ Unassign the application deployment **myAppDeployment** from the target server **newServer**.

```bash
wls:/offline/mydomain> unassign("AppDeployment", "myAppDeployment", "Target", "newServer")
```

■ Unassign the nested SubDeployment **MedRecAppScopedJMS.MedRecJMSServer**, which is a child of the AppDeployment **AppDeployment**, from the target server **AdminServer**.

```bash
```
3.7.20 undo

Command Category: Editing Commands
Use with WLST: Online

3.7.20.1 Description
Reverts all unsaved or unactivated edits.
You specify whether to revert all unactivated edits (including those that have been saved to disk), or all edits made since the last save operation. This command does not release the edit session.
In the event of an error, the command returns a WLSTException.

3.7.20.2 Syntax
undo([unactivatedChanges], [defaultAnswer])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>unactivatedChanges</td>
<td>Optional. Boolean value specifying whether to undo all unactivated changes, including edits that have been saved to disk. This argument defaults to false, indicating that all edits since the last save operation are reverted.</td>
</tr>
<tr>
<td>defaultAnswer</td>
<td>Optional. Default response, if you would prefer not to be prompted at the command line. Valid values are y and n. This argument defaults to null, and WLST prompts you for a response.</td>
</tr>
</tbody>
</table>

3.7.20.3 Example
The following example reverts all changes since the last save operation. WLST prompts for verification before reverting.

```
wlst:/mydomain/edit !> undo()
Sure you would like to undo your changes? (y/n)
y
Discarded your in-memory changes successfully.
wls:/mydomain/edit>
```

The following example reverts all unactivated changes. WLST prompts for verification before reverting.

```
wls:/mydomain/edit !> undo('true')
Sure you would like to undo your changes? (y/n)
y
Discarded all your changes successfully.
wls:/mydomain/edit>
```

3.7.21 validate

Command Category: Editing Commands
Use with WLST: Online

3.7.21.1 Description
Validates the changes that have been made but have not yet been saved. This command enables you to verify that all changes are valid before saving them.
In the event of an error, the command returns a WLSTException.
3.7.21.2 Syntax

validate()

3.7.21.3 Example

The following example validates all changes that have been made but have not yet been saved.

```
wls:/mydomain/edit --> validate()
Validating changes ...
Validated the changes successfully
```

3.8 Information Commands

Use the WLST information commands, listed in Table 3–8, to interrogate domains, servers, and variables, and provide configuration bean, runtime bean, and WLST-related information.

<table>
<thead>
<tr>
<th>This command...</th>
<th>Enables you to...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>addListener</td>
<td>Add a JMX listener to the specified MBean.</td>
<td>Online</td>
</tr>
<tr>
<td>configToScript</td>
<td>Convert an existing server configuration (config directory) to an executable WLST script</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>dumpStack</td>
<td>Display stack trace from the last exception that occurred while performing a WLST action, and reset the stack trace.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>dumpVariables</td>
<td>Display all variables used by WLST, including their name and value.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>find</td>
<td>Find MBeans and attributes in the current hierarchy.</td>
<td>Online</td>
</tr>
<tr>
<td>getConfigManager</td>
<td>Return the latest ConfigurationManagerBean MBean which manages the change process.</td>
<td>Online</td>
</tr>
<tr>
<td>getMBean</td>
<td>Return the MBean by browsing to the specified path.</td>
<td>Online</td>
</tr>
<tr>
<td>getMBI</td>
<td>Return the MBeanInfo for the specified MBeanType or the cmo variable.</td>
<td>Online</td>
</tr>
<tr>
<td>getPath</td>
<td>Return the MBean path for the specified MBean instance.</td>
<td>Online</td>
</tr>
<tr>
<td>listChildTypes</td>
<td>List all the children MBeans that can be created or deleted for the cmo type.</td>
<td>Online</td>
</tr>
<tr>
<td>lookup</td>
<td>Look up the specified MBean.</td>
<td>Online</td>
</tr>
<tr>
<td>ls</td>
<td>List all child beans and/or attributes for the current configuration or runtime bean.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>man</td>
<td>Display help from MBeanInfo for the current MBean or its specified attribute.</td>
<td>Online</td>
</tr>
<tr>
<td>redirect</td>
<td>Redirect WLST output to the specified filename.</td>
<td>Online</td>
</tr>
<tr>
<td>removeListener</td>
<td>Remove a listener that was previously defined.</td>
<td>Online</td>
</tr>
</tbody>
</table>
3.8.1 addListener

Command Category: Information Commands

Use with WLST: Online

3.8.1.1 Description

Adds a JMX listener to the specified MBean. Any changes made to the MBean are reported to standard out and/or are saved to the specified configuration file.

In the event of an error, the command returns a WLSTException.

3.8.1.2 Syntax

addListener(mbean, [attributeNames], [logFile], [listenerName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>mbean</td>
<td>Name of the MBean or MBean object to listen on.</td>
</tr>
<tr>
<td>attributeNames</td>
<td>Optional. Comma-separated list of all attribute names on which you would like to add a JMX listener. This argument defaults to null, and adds a JMX listener for all attributes.</td>
</tr>
<tr>
<td>logFile</td>
<td>Optional. Name and location of the log file to which you want to write listener information. This argument defaults to standard out.</td>
</tr>
<tr>
<td>listenerName</td>
<td>Optional. Name of the JMX listener. This argument defaults to a WLST-generated name.</td>
</tr>
</tbody>
</table>

3.8.1.3 Example

The following example defines a JMX listener on the cmo MBean for the Notes and ArchiveConfigurationCount attributes. The listener is named domain-listener and is stored in ./listeners/domain.log.
3.8.2 configToScript

Command Category: Information Commands

Use with WLST: Online or Offline

Converts an existing server configuration (config directory) to an executable WLST script. You can use the resulting script to re-create the resources on other servers.

---

**Note:** If you use `configToScript` for a domain that contains other Fusion Middleware components in addition to WebLogic Server, be aware that `configToScript` does not include the configuration for those components in the resulting WLST script. Only the WebLogic Server configuration is included in the script. If your domain contains other Fusion Middleware components in addition to WebLogic Server, Oracle recommends that you use `pack` and `unpack` to create the domain on remote servers. See *Creating Templates and Domains Using the Pack and Unpack Commands*.

---

The `configToScript` command creates the following files:

- A WLST script that contains the commands needed to recreate the configuration.
- A properties file that contains domain-specific values. You can update the values in this file to create new domains that are similar to the original configuration.
- A user configuration file and an associated key file to store encrypted attributes. The user configuration file contains the encrypted information. The key file contains a secret key that is used to encrypt and decrypt the encrypted information.

When you run the generated script:

- If a server is currently running, WLST will try to connect using the values in the properties file and then run the script commands to create the server resources.
- If no server is currently running, WLST will start a server with the values in the properties file, run the script commands to create the server resources, and shutdown the server. This may cause WLST to exit from the command shell.

In the event of an error, the command returns a `WLSTException`.

### 3.8.2.1 Syntax

```python
configToScript([configPath], [pyPath], [overwrite], [propertiesFile], [createDeploymentScript])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>configPath</code></td>
<td>Optional. Path to the domain directory that contains the configuration that you want to convert. This argument defaults to the directory from which you start WLST (./).</td>
</tr>
<tr>
<td><code>pyPath</code></td>
<td>Optional. Path and filename to which you want to write the converted WLST script. This argument defaults to ./config/config.py.</td>
</tr>
</tbody>
</table>
### 3.8.2.2 Example

The following example converts the configuration to a WLST script `config.py`. By default, the configuration file is loaded from `./config`, the script file is saved to `./config/config.py`, and the properties file is saved to `./config/config.py.properties`.

```bash
wls:/offline> configToScript()
configToScript is loading configuration from c:\Oracle\Middleware\user_projects\domains\wls\config\config.xml ...
Completed configuration load, now converting resources to wlst script...
configToScript completed successfully
The WLST script is written to c:\Oracle\Middleware\user_projects\domains\wls\config\config.py
and the properties file associated with this script is written to c:\Oracle\Middleware\user_projects\domains\wls\config\config.py.properties
wls:/offline>
```

The following example converts server resources configured in the file `c:\Oracle\Middleware\user_projects\domains\mydomain\config` directory to a WLST script `c:\Oracle\Middleware\myscripts\config.py`.

```bash
wls:/offline> configToScript('c:\Oracle\Middleware\user_projects\domains\mydomain','c:\Oracle\Middleware\myscripts')
configToScript is loading configuration from c:\Oracle\Middleware\user_projects\domains\mydomain\config\config.xml ...
Completed configuration load, now converting resources to wlst script...
configToScript completed successfully
The WLST script is written to c:\Oracle\Middleware\myscripts\config.py
and the properties file associated with this script is written to c:\Oracle\Middleware\mydomain\config.py.properties
wls:/offline>
```

### 3.8.3 dumpStack

**Command Category:** Information Commands

**Use with WLST:** Online or Offline

#### 3.8.3.1 Description

Displays the stack trace from the last exception that occurred while performing a WLST action, and resets the stack trace.

If successful, the `dumpstack` command returns the Throwable object. In the event of an error, the command returns a `WLSTException`.

---

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>overwrite</td>
<td>Optional. Boolean value specifying whether the script file should be overwritten if it already exists. This argument defaults to <code>true</code>, indicating that the script file is overwritten.</td>
</tr>
<tr>
<td>propertiesFile</td>
<td>Optional. Path to the directory in which you want WLST to write the properties files. This argument defaults to the pathname specified for the <code>scriptPath</code> argument.</td>
</tr>
<tr>
<td>createDeploymentScript</td>
<td>Optional. Boolean value specifying whether WLST creates a script that performs deployments only. This argument defaults to <code>false</code>, indicating that a deployment script is not created.</td>
</tr>
</tbody>
</table>
3.8.3.2 Syntax

dumpStack()

3.8.3.3 Example

This example displays the stack trace.

wls:/myserver/serverConfig> dumpStack()
com.bea.plateng.domain.script.jython.WLSTException: java.lang.reflect.InvocationTargetException
... 

3.8.4 dumpVariables

Command Category: Information Commands
Use with WLST: Online or Offline

3.8.4.1 Description

Displays all the variables used by WLST, including their name and value. In the event of an error, the command returns a WLSTException.

3.8.4.2 Syntax

dumpVariables()

3.8.4.3 Example

This example displays all the current variables and their values.

wls:/mydomain/serverConfig> dumpVariables()
adminHome   weblogic.rmi.internal.BasicRemoteRef - hostID:
   ‘-1 108080150904263937S:localhost:[7001,8001,-1,-1,-1,-1,-1]
cmgr   [MBeanServerInvocationHandler]com.bea:Name=ConfigurationManager,
       Type=weblogic.management.mbeanservers.edit.ConfigurationManagerMBean
   cmo   [MBeanServerInvocationHandler]com.bea:Name=mydomain,Type=Domain
connected true
domainName mydomain
...

wls:/mydomain/serverConfig>

3.8.5 find

Command Category: Information Commands
Use with WLST: Online

3.8.5.1 Description

Finds MBeans and attributes in the current hierarchy.

WLST returns the pathname to the MBean that stores the attribute and/or attribute type, and its value. If searchInstancesOnly is set to false, this command also searches the MBeanType paths that are not instantiated in the server, but that can be created. In the event of an error, the command returns a WLSTException.

3.8.5.2 Syntax

find([name], [type], [searchInstancesOnly])
3.8.5.3 Example

The following example searches for an attribute named javaCompiler in the current configuration hierarchy.

```bash
wls:/mydomain/serverConfig> find(name = 'JavaCompiler')
Finding 'JavaCompiler' in all registered MBean instances ...
/Servers/AdminServer  JavaCompilerPreClassPath  null
/Servers/AdminServer  JavaCompiler              java
/Servers/AdminServer  JavaCompilerPostClassPath null
wls:/mydomain/serverConfig>
```

The following example searches for an attribute of type JMSRuntime in the current configuration hierarchy.

```bash
wls:/mydomain/serverRuntime> find(type='JMSRuntime')
Finding MBean of type 'JMSRuntime' in all the instances ...
/JMSRuntime/AdminServer.jms
wls:/mydomain/serverRuntime>
```

The following example searches for an attribute named execute in the current configuration hierarchy. The searchInstancesOnly argument is set to false, indicating to also search MBeanTypes that are not instantiated in the server.

```bash
wls:/mydomain/serverConfig> find(name='execute', searchInstancesOnly='false')
Finding 'execute' in all registered MBean instances ...
/Servers/AdminServer  ExecuteQueues [Ljava.lang.Object;@1aa7dbc
/Servers/AdminServer  Use81StyleExecuteQueues  false
Now finding 'execute' in all MBean Types that can be instantiated ...
/Servers  ExecuteQueues
/Servers  Use81StyleExecuteQueues
wls:/mydomain/serverConfig>
```

3.8.6 getConfigManager

Command Category: Information Commands

Use with WLST: Online

3.8.6.1 Description

Returns the latest ConfigurationManager MBean which manages the change process. You can then invoke methods to manage configuration changes across a WebLogic domain. In the event of an error, the command returns a WLSTException.

3.8.6.2 Syntax

getConfigManager()
3.8.6.3 Example
The following example returns the latest ConfigurationManagerBean MBean and stores it in a cm variable.

```wls:/mydomain/serverConfig> cm=getConfigManager()
wls:/mydomain/serverConfig> cm.getType()
'weblogic.management.mbeanservers.edit.ConfigurationManagerMBean'
```

3.8.7 getMBean

Command Category: Information Commands

Use with WLST: Online

3.8.7.1 Description
Returns the MBean by browsing to the specified path. In the event of an error, the command returns a WLSTException.

**Note:** No exception is thrown if the MBean is not found.

3.8.7.2 Syntax
getMBean(mbeanPath)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>mbeanPath</td>
<td>Path name to the MBean in the current hierarchy.</td>
</tr>
</tbody>
</table>

3.8.7.3 Example
The following example returns the MBean specified by the path.

```wls:/mydomain/edit !> com=getMBean('Servers/myserver/COM/myserver')
wls:/mydomain/edit !> com.getType()
'Server'
```

3.8.8 getMBI

Command Category: Information Commands

Use with WLST: Online

3.8.8.1 Description
Returns the MBeanInfo for the specified MBeanType or the cmo variable. In the event of an error, the command returns a WLSTException.

3.8.8.2 Syntax
getMBI([mbeanType])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>mbeanType</td>
<td>Optional. MBeanType for which the MBeanInfo is displayed.</td>
</tr>
</tbody>
</table>

3.8.8.3 Example
The following example gets the MBeanInfo for the specified MBeanType and stores it in the variable svrMbi.
3.8.9 getPath

Command Category: Information Commands

Use with WLST: Online

3.8.9.1 Description

Returns the MBean path for the specified MBean instance or ObjectName for the MBean in the current tree. In the event of an error, the command returns a WLSTException.

3.8.9.2 Syntax

getPath(mbean)

Argument | Definition
--- | ---
mbean | MBean instance or ObjectName for the MBean in the current tree for which you want to return the MBean path.

3.8.9.3 Example

The following example returns the MBean specified by the path.

```bash
wls:/mydomain/edit !> path=getPath('com.bea:Name=myserver,Type=Server')
```

```
'_servers/myserver'
```

3.8.10 listChildTypes

Command Category: Information Commands

Use with WLST: Online

3.8.10.1 Description

Lists all the child MBeans that can be created or deleted for the cmo. The cmo variable specifies the configuration bean instance to which you last navigated using WLST. For more information about the cmo variable, see "Changing the Current Management Object" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

In the event of an error, the command returns a WLSTException.

3.8.10.2 Syntax

listChildTypes([parent])

Argument | Definition
--- | ---
parent | Optional. Parent type for which you want the children types listed.

3.8.10.3 Example

The following example lists the children MBeans that can be created or deleted for the cmo type.

```bash
wls:/mydomain/serverConfig> listChildTypes()
AppDeployments
```

```
```
3.8.11 lookup

Command Category: Information Commands

Use with WLST: Online

3.8.11.1 Description

Looks up the specified MBean. The MBean must be a child of the current MBean. In the event of an error, the command returns a WLSTException.

3.8.11.2 Syntax

lookup(name, [childMBeanType])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the MBean that you want to lookup.</td>
</tr>
<tr>
<td>childMBeanType</td>
<td>Optional. The type of the MBean that you want to lookup.</td>
</tr>
</tbody>
</table>

3.8.11.3 Example

The following example looks up the specified server, myserver, and stores the returned stub in the sbean variable.

```wls::mydomain/serverConfig> sbean=lookup('myserver','Server')
```

```wls::mydomain/serverConfig> sbean.getType()
'Server'
```

3.8.12 ls

Command Category: Information Commands

Use with WLST: Online or Offline

3.8.12.1 Description

Lists the attributes, operations, and child management objects of the specified management object.

In the event of an error, the command returns a WLSTException.

By default, the output is returned as a string and is arranged in three columns:
Information Commands

---

**Note:** By default, the `ls()` function echos its output so that it appears on the console in which you are running WLST. You can disable this by including the following command in your WLST script:

```java
WLS.setShowLSResult(flag)
```

`flag` is a Boolean value. If it is 0 (false), then output from `ls()` to stdout is disabled and will not appear on the console. If it is set to 1 (true), then output from `ls()` to stdout is enabled.

- The first column displays a set of codes that describe the listed item. See Table 3–9.
- The second column displays the item name.
- When the item is an attribute, the third column displays the attribute value. If an attribute is encrypted, the third column displays asterisks instead of the value. (See “Writing and Reading Encrypted Configuration Values” in *Oracle Fusion Middleware Oracle WebLogic Scripting Tool.*)
- When the item is an operation, the third column uses the following pattern to display the operation’s return type and input parameters: `returnType: parameterType(parameterName)`

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>d</td>
<td>Indicates that the item is a child management object.</td>
</tr>
<tr>
<td></td>
<td>Like a directory in a UNIX or Windows file system, you can use the <code>cd</code> command to make the child object the current management object.</td>
</tr>
<tr>
<td>r</td>
<td>Indicates that the item is a child management object or an attribute that is readable, assuming that current user has been given read permission by the security realm’s policies. (See “Default Security Policies for MBeans” in <em>Oracle Fusion Middleware Oracle WebLogic Server MBean Reference.</em>)</td>
</tr>
<tr>
<td>w</td>
<td>Indicates that the item is an attribute that is writable, assuming that current user has been given write permission by the security realm’s policies. (See “Default Security Policies for MBeans” in <em>Oracle WebLogic Server MBean Reference.</em>)</td>
</tr>
<tr>
<td>x</td>
<td>Indicates that the item is an operation that can be executed, assuming that current user has been given execute permission by the security realm’s policies. (See “Default Security Policies for MBeans” in <em>Oracle Fusion Middleware Oracle WebLogic Server MBean Reference.</em>)</td>
</tr>
</tbody>
</table>

By default, the output lists all attributes, operations, and child management objects of the current management object. To filter the output or to see a list for a different management object, you can specify a command argument.
3.8.12.2 Syntax

\texttt{ls( [ a | c | o ] [ moPath ])}

\texttt{ls( [ moPath ] returnMap [ returnType ] )}

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Optional. Displays only the attributes of the specified management object (suppresses the display of other items).</td>
</tr>
<tr>
<td>c</td>
<td>Optional. Displays only the child management objects of the specified management object (suppresses the display of other items).</td>
</tr>
<tr>
<td>o</td>
<td>Optional. Displays only the operations that can be invoked on the specified management object (suppresses the display of other items). This argument is only applicable for WLST online.</td>
</tr>
</tbody>
</table>
| moPath      | Optional. Path name to the management object for which you want to list attributes, operations, and child management objects. You can specify a path name that is relative to your current location in the hierarchy or an absolute path name. With WLST offline, use the forward-slash character (/) to specify the root of the configuration document. With WLST online, you can list the contents of MBeans in any management hierarchy (see Section 3.11, "Tree Commands"). Use the following syntax to specify the root of a hierarchy: 
\texttt{root-name:/}
For example, to list the root of the server runtime hierarchy:
\texttt{ls('serverRuntime:/')}
If you do not specify this argument, the command lists items for the current management object. |
| returnMap   | Optional. Boolean value that determines whether the command returns values as a map. This argument defaults to false, which causes this command to return a String. |
| returnType  | Optional. Controls the output returned in the map. Specify \texttt{a}, \texttt{c}, or \texttt{o}, which filter the output as described at the top of this table. This argument is valid only if \texttt{returnMap} is set to true. This argument defaults to \texttt{c}. |
### 3.8.12.3 Example

The following example displays all the child configuration beans, and attribute names and values for the examples domain, which has been loaded into memory, in WLST offline mode:

```
wls:/offline/mydomain > ls()
```

```
-  AppDeployments
-  BridgeDestinations
-  Clusters
-  CustomResources
-  DeploymentConfiguration
-  Deployments
-  EmbeddedLDAP
-  ErrorHandlings
-  FileStores
-  InternalAppDeployments
-  InternalLibraries
-  JDBCDataSourceFactories
-  JDBCStores
-  JDBCSystemResources
-  JMSBridgeDestinations
-  JMSInteropModules
-  JNNServers
-  JNNSystemResources
-  JMX
...
```

```
wls:/offline/examples>
```

The following example displays all the attribute names and values in `DomainMBean`:

```
wls:/mydomain/serverConfig> ls('a')
```

```
-AdminServerName AdminServer
-AdministrationMBeanAuditingEnabled false
-AdministrationPort 9002
-AdministrationPortEnabled false
-AdministrationProtocol t3s
-ArchiveConfigurationCount 0
-ClusterConstraintsEnabled false
-ConfigBackupEnabled false
-ConfigurationAuditType none
-ConfigurationVersion 9.0.0.0
-ConsoleContextPath console
-ConsoleEnabled true
-ConsoleExtensionDirectory console-ext
-DomainVersion 9.0.0.0
-LastModificationTime 0
-Name basicWLSDomain
-Notes null
-Parent null
-ProductionModeEnabled false
-RootDirectory .
-Type Domain
```

```
wls:/mydomain/serverConfig>
```

The following example displays all the child beans and attribute names and values in `Servers MBean`:

```
wls:/mydomain/serverConfig> ls('Servers')
```

```
AdminServer
```
The following example displays the attribute names and values for the specified MBean path and returns the information in a map:

```
svrAttrList = ls('edit:/Servers/myserver', 'true', 'a')
```

```
-wr- AcceptBacklog                                50
-wr- AdminReconnectIntervalSeconds                10
-wr- AdministrationPort                           9002
-wr- AdministrationProtocol                       t3s
-wr- AutoKillIfFailed                             false
-wr- AutoMigrationEnabled                         false
-wr- AutoRestart                                  true
-wr- COMEnabled                                   false
-wr- ClasspathServletDisabled                     false
-wr- ClientCertProxyEnabled                       false
-wr- Cluster                                      null
-wr- ClusterRuntime                               null
-wr- ClusterWeight                                100
```

```
wls:/mydomain/serverConfig>
```

### 3.8.13 man

**Command Category:** Information Commands

**Use with WLST:** Online

#### 3.8.13.1 Description

Displays help from MBeanInfo for the current MBean or its specified attribute. In the event of an error, the command returns a WLSTException.

#### 3.8.13.2 Syntax

```
man([attrName])
```

#### 3.8.13.3 Example

The following example displays help from MBeanInfo for the ServerMBean bean.

```
wls:/mydomain/serverConfig> man('Servers')
dynamic : true
creator : createServer
destroyer : destroyServer
description : <p>Returns the ServerMBeans representing the servers that have been configured to be part of this domain.</p>
descriminatorType : Attribute
Name : Servers
interfaceClassName : [Lweblogic.management.configuration.ServerMBean;
displayName : Servers
relationship : containment
```

### 3.8.14 redirect

**Command Category:** Information Commands

**Use with WLST:** Online
3.8.14.1 Description
Redirects WLST information, error, and debug messages to the specified filename. Also redirects the output of the dumpStack() and dumpVariables() commands to the specified filename.

In the event of an error, the command returns a WLSTException.

3.8.14.2 Syntax
redirect(outputFile, [toStdOut])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>outputFile</td>
<td>Name of the file to which you want to record the WLST commands. The filename can be absolute or relative to the directory from which you started WLST.</td>
</tr>
<tr>
<td>toStdOut</td>
<td>Optional. Boolean value specifying whether the output should be sent to stdout. This argument defaults to true, indicating that the output will be sent to stdout.</td>
</tr>
</tbody>
</table>

3.8.14.3 Example
The following example begins redirecting WLST output to the logs/wlst.log file:

wls:/mydomain/serverConfig> redirect('./logs/wlst.log')

3.8.15 removeListener
Command Category: Information Commands
Use with WLST: Online

3.8.15.1 Description
Removes a listener that was previously defined. If you do not specify an argument, WLST removes all listeners defined for all MBeans. For information about setting a listener, see Section 3.8.1, "addListener".

In the event of an error, the command returns a WLSTException.

3.8.15.2 Syntax
removeListener([mbean], [listenerName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>mbean</td>
<td>Optional. Name of the MBean or MBean object for which you want to remove the previously defined listeners.</td>
</tr>
<tr>
<td>listenerName</td>
<td>Optional. Name of the listener to be removed.</td>
</tr>
</tbody>
</table>

3.8.15.3 Example
The following example removes the listener named mylistener.

wls:/mydomain/serverConfig> removeListener(listenerName="mylistener")

3.8.16 showListeners
Command Category: Information Commands
Use with WLST: Online
3.8.16.1 Description
Shows all listeners that are currently defined. For information about setting a listener, see Section 3.8.1, "addListener".

In the event of an error, the command returns a WLSTException.

3.8.16.2 Syntax
showListeners()

3.8.16.3 Example
The following example shows all listeners that are currently defined.

wls:/mydomain/serverConfig> showListeners()

3.8.17 startRecording
Command Category: Information Commands
Use with WLST: Online or Offline

3.8.17.1 Description
Records all user interactions with WLST. This command is useful for capturing commands for replay.

In the event of an error, the command returns a WLSTException.

This command cannot be used when you are importing WLST as a Jython module, as described in "Importing WLST as a Jython Module" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

3.8.17.2 Syntax
startRecording(recordFile, [recordAll])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>recordFile</td>
<td>Name of the file to which you want to record the WLST commands. The filename can be absolute or relative to the directory from which you invoked WLST.</td>
</tr>
<tr>
<td>recordAll</td>
<td>Optional. Boolean value specifying whether to capture all user interactions in the file. This argument defaults to false, indicating that only WLST commands are captured, and not WLST command output.</td>
</tr>
</tbody>
</table>

3.8.17.3 Example
The following example begins recording WLST commands in the record.py file:

wls:/mydomain/serverConfig> startRecording('c:/myScripts/record.py')
Starting recording to c:/myScripts/record.py

3.8.18 state
Command Category: Information Commands
Use with WLST: Online
3.8.18.1 Description
Using Node Manager, returns a map of servers or clusters and their state. Node Manager must be running.

For more information about server states, see "Understanding Server Life Cycle" in Oracle Fusion Middleware Managing Server Startup and Shutdown for Oracle WebLogic Server.

In the event of an error, the command returns a WLSTException.

3.8.18.2 Syntax
state(name, [type])

3.8.18.3 Example
The following example returns the state of the Managed Server, managed1.

wls:/mydomain/serverConfig> state('managed1', 'Server')
Current state of "managed1": SUSPENDED
wls:/mydomain/serverConfig>

The following example returns the state of the cluster, mycluster.

wls:/mydomain/serverConfig> state('mycluster', 'Cluster')
There are 3 server(s) in cluster: mycluster

States of the servers are
MServer1---SHUTDOWN
MServer2---SHUTDOWN
MServer3---SHUTDOWN
wls:/mydomain/serverConfig>

3.8.19 stopRecording

Command Category: Information Commands
Use with WLST: Online or Offline

3.8.19.1 Description
Stops recording WLST commands. For information about starting a recording, see Section 3.8.17, "startRecording".

In the event of an error, the command returns a WLSTException.

3.8.19.2 Syntax
stopRecording()

3.8.19.3 Example
The following example stops recording WLST commands.

wls:/mydomain/serverConfig> stopRecording()
3.8.20 stopRedirect

**Command Category:** Information Commands

**Use with WLST:** Online or Offline

**3.8.20.1 Description**

Stops the redirection of WLST output to a file, if redirection is in progress.

In the event of an error, the command returns a `WLSTException`.

**3.8.20.2 Syntax**

`stopRedirect()`

**3.8.20.3 Example**

The following example stops the redirection of WLST output to a file:

```plaintext
wls:/mydomain/serverConfig> stopRedirect()
WLST output will not be redirected to myfile.txt any more
```

3.8.21 storeUserConfig

**Command Category:** Information Commands

**Use with WLST:** Online

**3.8.21.1 Description**

Creates a user configuration file and an associated key file. The user configuration file contains an encrypted username and password. The key file contains a secret key that is used to encrypt and decrypt the username and password.

Only the key file that originally encrypted the username and password can be used to decrypt the values. If you lose the key file, you must create a new user configuration and key file pair.

In the event of an error, the command returns a `WLSTException`.

**3.8.21.2 Syntax**

`storeUserConfig([userConfigFile], [userKeyFile], [nm])`
3.8.21.3 Example

The following example creates and stores a user configuration file and key file in the default location.

```
wlst:/mydomain/serverConfig> storeUserConfig()
Creating the key file can reduce the security of your system if it is not kept in a secured location after it is created. Do you want to create the key file? y or n
y
The username and password that were used for this current WLS connection are stored in C:\Documents and Settings\pat\pat-WebLogicConfig.properties and C:\Documents and Settings\pat\pat-WebLogicKey.properties.
```

The following example creates and stores a user configuration file and key file in the specified locations.

```
wlst:/mydomain/serverConfig> storeUserConfig('c:/myFiles/myuserconfigfile.secure', 'c:/myFiles/myuserkeyfile.secure')
Creating the key file can reduce the security of your system if it is not kept in a secured location after it is created. Do you want to create the key file? y or n
y
The username and password that were used for this current WLS connection are stored in c:/myFiles/myuserconfigfile.secure and c:/myFiles/myuserkeyfile.secure
```

### Argument | Definition
--- | ---
userConfigFile | Optional. Name of the file to store the user configuration. The pathname can be absolute or relative to the file-system directory from which you started WLST.

If you do not specify this argument, the command stores the file in your home directory as determined by your JVM. The location of the home directory depends on the SDK and type of operating system on which WLST is running. The default filename is based on the following pattern:

```
username-WebLogicConfig.properties
```

where `username` is the user name that you used to log in to the operating system.

The command also prints to standard out the location in which it created the file.

userKeyFile | Optional. Name of the file to store the key information that is associated with the user configuration file that you specify. The pathname can be absolute or relative to the file-system directory from which you started WLST.

If you do not specify this argument, the command stores the file in your home directory as determined by your JVM. The location of the home directory depends on the SDK and type of operating system on which WLST is running. The default filename is based on the following pattern:

```
username-WebLogicKey.properties
```

where `username` is the user name that you used to log in to the operating system.

The command also prints to standard out the location in which it created the file.

nm | Optional. Boolean value specifying whether to store the username and password for Node Manager or WebLogic Server. If set to true, the Node Manager username and password is stored. This argument default to false.
3.8.22 threadDump

Command Category: Information Commands
Use with WLST: Online or Offline

3.8.22.1 Description
Displays a thread dump for the specified server. In the event of an error, the command returns a WLSTException.

3.8.22.2 Syntax
threadDump([writeToFile], [fileName], [serverName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>writeToFile</td>
<td>Optional. Boolean value specifying whether to save the output to a file. This argument defaults to true, indicating that output is saved to a file.</td>
</tr>
<tr>
<td>fileName</td>
<td>Optional. Name of the file to which the output is written. The filename can be absolute or relative to the directory where WLST is running. This argument defaults to Thread_Dump_serverName file, where serverName indicates the name of the server. This argument is valid only if writeToFile is set to true.</td>
</tr>
<tr>
<td>serverName</td>
<td>Optional. Server name for which the thread dump is requested. This argument defaults to the server to which WLST is connected. If you are connected to an Administration Server, you can display a thread dump for the Administration Server and any Managed Server that is running in the WebLogic domain. If you are connected to a Managed Server, you can only display a thread dump for that Managed Server.</td>
</tr>
</tbody>
</table>

3.8.22.3 Example
The following example displays the thread dump for the current server and saves the output to the Thread_Dump_serverName file.

```
wls:/mydomain/serverConfig> threadDump()
```

The following example displays the thread dump for the server managedServer. The information is not saved to a file.

```
wls:/mydomain/serverConfig> threadDump(writeToFile='false', serverName='managedServer')
```

3.8.23 viewMBean

Command Category: Information Commands
Use with WLST: Online

3.8.23.1 Description
Displays information about an MBean, such as the attribute names and values, and operations. In the event of an error, the command returns a WLSTException.

3.8.23.2 Syntax
viewMBean(mbean)
3.8.23.3 Example

The following example displays information about the current MBean, `cmo`.

```plaintext
wls:/mydomain/serverConfig> cmo.getType()
'Domain'
wls:/mydomain/serverConfig> viewMBean(cmo)
```

**Attribute Names and Values**

```
XMLEntityCaches   null
Targets   javax.management.ObjectName[com.bea
:Name=MedRecJMSServer,Type=JMSServer,
   com.bea:Name=WSStoreForwardInternalJMSServerMedRecServer,Type=JMSServer,
   com.bea:Name=MedRec%seeJMSServer,Type=JMSServer,
   com.bea:Name=PhysNSEEJMSServer,Type=JMSServer,
   com.bea:Name=MedRecSAFAgent,Type=SAFAgent,
   com.bea:Name=AdminServer,Type=Server]
RootDirectory                                .
EmbeddedLDAP                          com.bea:Name=OOTB_medrec,Type=EmbeddedLDAP
RemoteSAFContexts  null
Libraries   javax.management.ObjectName[com.bea
... 
```

3.8.24 writeIniFile

**Command Category:** Information Commands

**Use with WLST:** Online

**3.8.24.1 Description**

Converts WLST definitions and method declarations to a Python (.py) file to enable advanced users to import them as a Jython module. After importing, the definitions and method declarations are available to other Jython modules and can be accessed directly using Jython syntax. For more information, see “Importing WLST as a Jython Module” in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

In the event of an error, the command returns a `WLSTException`.

**3.8.24.2 Syntax**

```
writeIniFile(filePath)
```

**Argument**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>filePath</td>
<td>Full pathname to the file that you want to save the converted information.</td>
</tr>
</tbody>
</table>

**3.8.24.3 Example**

The following example converts WLST to a Python file named `wl.py`.

```plaintext
wls:/offline> writeIniFile("wl.py")
The Ini file is successfully written to wl.py
wls:/offline>
```
3.9 Life Cycle Commands

Use the WLST life cycle commands, listed in Table 3-10, to manage the life cycle of a server instance.

For more information about the life cycle of a server instance, see "Understanding Server Life Cycle" in Oracle Fusion Middleware Managing Server Startup and Shutdown for Oracle WebLogic Server.

Table 3-10 Life Cycle Commands for WLST Configuration

<table>
<thead>
<tr>
<th>This command</th>
<th>Enables you to...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>migrate</td>
<td>Migrate services to a target server within a cluster.</td>
<td>Online</td>
</tr>
<tr>
<td>resume</td>
<td>Resume a server instance that is suspended or in ADMIN state.</td>
<td>Online</td>
</tr>
<tr>
<td>shutdown</td>
<td>Gracefully shut down a running server instance or cluster.</td>
<td>Online</td>
</tr>
<tr>
<td>start</td>
<td>Start a Managed Server instance or a cluster using Node Manager.</td>
<td>Online</td>
</tr>
<tr>
<td>startServer</td>
<td>Start the Administration Server.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>suspend</td>
<td>Suspend a running server.</td>
<td>Online</td>
</tr>
</tbody>
</table>

3.9.1 migrate

Command Category: Life Cycle Commands

Use with WLST: Online

3.9.1.1 Description

Migrates the specified services (JTA, JMS, or Server) to a targeted server within a cluster. In the event of an error, the command returns a WLSTException.

For information about migrating services, see "Service Migration" in Oracle Fusion Middleware Using Clusters for Oracle WebLogic Server.

3.9.1.2 Syntax

migrate(sname, destinationName, [sourceDown], [destinationDown], [migrationType])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>sname</td>
<td>Name of the migratable target which should be migrated.</td>
</tr>
<tr>
<td>destinationName</td>
<td>Name of the machine or server to which you want to migrate the services.</td>
</tr>
<tr>
<td>sourceDown</td>
<td>Optional. Boolean value specifying whether the source server is down.</td>
</tr>
<tr>
<td></td>
<td>This argument defaults to true, indicating that the source server is not running.</td>
</tr>
<tr>
<td></td>
<td>When migrating JTA services, the sourceDown argument is ignored, if specified, and defaults to true. The source server must be down in order for the migration of JTA services to succeed.</td>
</tr>
</tbody>
</table>
3.9.1.3 Example

The following example migrates all JMS and JTA services on server1 to the server server2. The boolean arguments specify that the source server is down and the destination server is running.

```
wls:/mydomain/edit !> migrate('server1','server2', 'true', 'false', 'all')
Migrating all JMS and JTA services from 'server1' to destination 'server2' ...
wls:/mydomain/edit !>
```

The following example migrates all Server services on server1 to the server server2. The boolean arguments specify that the source server is down and the destination server is running.

```
wls:/mydomain/edit !> migrate('server1','server2', 'true', 'false', 'Server')
Migrating singleton server services from 'server1' to machine 'server2' ...
wls:/mydomain/edit !>
```

3.9.2 resume

Command Category: Life Cycle Commands

Use with WLST: Online

3.9.2.1 Description

Resumes a server instance that is suspended or in ADMIN state. This command moves a server to the RUNNING state. For more information about server states, see "Understanding Server Life Cycle" in Oracle Fusion Middleware Managing Server Startup and Shutdown for Oracle WebLogic Server.

In the event of an error, the command returns a WLSTException.

3.9.2.2 Syntax

```
resume([sname], [block])
```
3.9.2.3 Example
The following example resumes a Managed Server instance.

```
wls:/mydomain/serverConfig> resume('managed1', block='true')
Server 'managed1' resumed successfully.
wls:/mydomain/serverConfig>
```

3.9.3 shutdown

Command Category: Life Cycle Commands
Use with WLST: Online

3.9.3.1 Description
Gracefully shuts down a running server instance or a cluster. The `shutdown` command waits for all the in-process work to be completed before shutting down the server or cluster.

You shut down a server to which WLST is connected by entering the `shutdown` command without any arguments.

When connected to a Managed Server instance, you only use the `shutdown` command to shut down the Managed Server instance to which WLST is connected; you cannot shut down another server while connected to a Managed Server instance.

WLST uses Node Manager to shut down a Managed Server. When shutting down a Managed Server, Node Manager must be running.

In the event of an error, the command returns a `WLSTException`.

3.9.3.2 Syntax
```
shutdown([name], [entityType], [ignoreSessions], [timeOut], [force], [block])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Optional. Name of the server or cluster to shutdown. This argument defaults to the server to which WLST is currently connected.</td>
</tr>
<tr>
<td>entityType</td>
<td>Optional. Type, Server or Cluster. This argument defaults to Server. When shutting down a cluster, you must set this argument explicitly to Cluster, or the command will fail.</td>
</tr>
<tr>
<td>ignoreSessions</td>
<td>Optional. Boolean value specifying whether WLST should drop all HTTP sessions immediately or wait for HTTP sessions to complete or timeout while shutting down. This argument defaults to false, indicating that all HTTP sessions must complete or timeout.</td>
</tr>
</tbody>
</table>
3.9.3.3 Example

The following example instructs WLST to shutdown the server to which you are connected:

```bash
wls:/mydomain/serverConfig> shutdown()
Shutting down the admin server that you are currently connected to .......
Disconnected from weblogic server: AdminServer
```

The following example instructs WLST to wait 1000 seconds for HTTP sessions to complete or timeout (at 1000 seconds) before shutting down myserver:

```bash
wls:/mydomain/serverConfig> shutdown('myserver','Server','false',1000,
block='false')
```

The following example instructs WLST to drop all HTTP sessions immediately while connected to a Managed Server instance:

```bash
wls:/mydomain/serverConfig> shutdown('MServer1','Server','true',1200)
Shutting down a managed server that you are connected to ...
Disconnected from weblogic server: MServer1
```

The following example instructs WLST to shutdown the cluster mycluster:

```bash
wls:/mydomain/serverConfig> shutdown('mycluster','Cluster')
Shutting down the cluster with name mycluster
Shutdown of cluster mycluster has been issued, please refer to the logs to check if the cluster shutdown is successful. Use the state(<server-name>) or state(<cluster-name>,”Cluster”) to check the status of the server or cluster
wls:/mydomain/serverConfig> state('mycluster','Cluster')
There are 3 server(s) in cluster: mycluster

States of the servers are
MServer1----SHUTDOWN
MServer2----SHUTDOWN
MServer3----SHUTDOWN
wls:/mydomain/serverConfig>
```

---

### Table: Arguments and Definitions

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>timeOut</code></td>
<td>Optional. Time (in seconds) that WLST waits for subsystems to complete in-process work and suspend themselves before shutting down the server. This argument defaults to 0 seconds, indicating that there is no timeout.</td>
</tr>
<tr>
<td><code>force</code></td>
<td>Optional. Boolean value specifying whether WLST should terminate a server instance or a cluster without waiting for the active sessions to complete. This argument defaults to <code>false</code>, indicating that all active sessions must complete before shutdown.</td>
</tr>
<tr>
<td><code>block</code></td>
<td>Optional. Boolean value specifying whether WLST should block user interaction until the server is shutdown. This argument defaults to <code>false</code>, indicating that user interaction is not blocked. In this case, WLST returns control to the user after issuing the command and assigns the task MBean associated with the current task to a variable that you can use to check its status. If you are importing WLST as a Jython module, as described in &quot;Importing WLST as a Jython Module&quot; in Oracle Fusion Middleware Oracle WebLogic Scripting Tool, <code>block</code> is always set to <code>true</code>.</td>
</tr>
</tbody>
</table>
3.9.4 start

Command Category: Life Cycle Commands

Use with WLST: Online

3.9.4.1 Description

Starts a Managed Server instance or a cluster using Node Manager. WLST must be connected to the Administration Server and Node Manager must be running.

For more information about WLST commands used to connect to and use Node Manager, see Section 3.10, "Node Manager Commands".

In the event of an error, the command returns a WLSTException.

3.9.4.2 Syntax

start(name, [type], [url], [block])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the Managed Server or cluster to start.</td>
</tr>
<tr>
<td>type</td>
<td>Optional. Type, Server or Cluster. This argument defaults to Server. When starting a cluster, you must set this argument explicitly to Cluster, or the command will fail.</td>
</tr>
<tr>
<td>url</td>
<td>Optional. Listen address and listen port of the server instance, specified using the following format: [protocol://]listen-address:listen-port. If not specified, this argument defaults to t3://localhost:7001.</td>
</tr>
<tr>
<td>block</td>
<td>Optional. Boolean value specifying whether WLST should block user interaction until the server or cluster is started. This argument defaults to false, indicating that user interaction is not blocked. In this case, WLST returns control to the user after issuing the command and assigns the task MBean associated with the current task to a variable that you can use to check its status. If you are importing WLST as a Jython module, as described &quot;Importing WLST as a Jython Module&quot; in Oracle Fusion Middleware Oracle WebLogic Scripting Tool, block is always set to true.</td>
</tr>
</tbody>
</table>

3.9.4.3 Example

The following example instructs Node Manager to start a Managed Server instance; the listen address is localhost and listen port is 8801. WLST returns control to the user after issuing this command, as block is set to false.

wls:/mydomain/serverConfig> start('myserver', 'Server', block='false')
Starting server myserver ...
Server with name myserver started successfully.

wls:/mydomain/serverConfig>

The following example instructs Node Manager to start a cluster. WLST block user interaction until the cluster is started, as block defaults to true.

wls:/mydomain/serverConfig> start('mycluster', 'Cluster')
Starting the following servers in Cluster, mycluster: MS1, MS2, MS3...

All servers in the cluster mycluster are started successfully.

wls:/mydomain/serverConfig>

3.9.5 startServer

Command Category: Life Cycle Commands
3.9.5.1 Description
Starts the Administration Server. In the event of an error, the command returns a WLSTException.

Note: You can use startServer only to start a WebLogic Administration Server, by running WLST from the WL_HOME/common/bin directory. You cannot use startServer to start an integrated WebLogic Administration Server (that is, an Administration Server for a Fusion Middleware Suite product installed in an ORACLE_HOME directory).

To start the Administration server for a Fusion Middleware Suite product other than WebLogic Server, use either of the following methods:

- Execute the server startup script for the associated WebLogic domain.
- Start the server using Node Manager. If you use this method, make sure that the startScriptEnabled property is set to true in Node Manager.

3.9.5.2 Syntax

```
startServer([adminServerName], [domainName], [url], [username], [password], [domainDir], [block], [timeout], [serverLog], [systemProperties], [jvmArgs] [spaceAsJvmArgsDelimiter])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adminServerName</td>
<td>Optional. Name of the Administration Server to start. This argument defaults to myserver.</td>
</tr>
<tr>
<td>domainName</td>
<td>Optional. Name of the WebLogic domain to which the Administration Server belongs. This argument defaults to mydomain.</td>
</tr>
<tr>
<td>url</td>
<td>Optional. URL of the Administration Server. The URL supplied with the startServer command will override the listen address and port specified in the config.xml file. If not specified on the command line or in the config.xml file, this argument defaults to t3://localhost:7001.</td>
</tr>
<tr>
<td>username</td>
<td>Optional. Username use to connect WLST to the server. This argument defaults to weblogic.</td>
</tr>
<tr>
<td>password</td>
<td>Optional. Password used to connect WLST to the server. This argument defaults to welcome1.</td>
</tr>
<tr>
<td>domainDir</td>
<td>Optional. Domain directory in which the Administration Server is being started. This argument directory to the directory from which you started WLST.</td>
</tr>
<tr>
<td>block</td>
<td>Optional. Boolean value specifying whether WLST blocks user interaction until the server is started. When block is set to false, WLST returns control to the user after issuing the command. This argument defaults to true, indicating that user interaction is blocked. If you are importing WLST as a Jython module, as described in &quot;Importing WLST as a Jython Module&quot; in Oracle Fusion Middleware Oracle WebLogic Scripting Tool, block is always set to true.</td>
</tr>
</tbody>
</table>
### 3.9.5.3 Example
The following example starts the Administration Server named `demoServer` in the `demoDomain`.

```
wlstalk://offline> startServer('demoServer','demoDomain','t3://localhost:8001', 'myweblogic','wlstdomain','c:/mydomains/wlst','false', 60000, jvmArgs='-XX:MaxPermSize=75m, -Xmx512m, -XX:+UseParallelGC')
wlstalk://offline>
```

### 3.9.6 suspend

Command Category: Life Cycle Commands

Use with WLST: Online

#### 3.9.6.1 Description
Suspends a running server. This command moves a server from the `RUNNING` state to the `ADMIN` state. For more information about server states, see "Understanding Server Life Cycle" in Oracle Fusion Middleware Managing Server Startup and Shutdown for Oracle WebLogic Server.

In the event of an error, the command returns a `WLSTException`.

#### 3.9.6.2 Syntax

```
suspend([sname], [ignoreSessions], [timeOut], [force], [block])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>sname</td>
<td>Optional. Name of the server to suspend. The argument defaults to the server to which WLST is currently connected.</td>
</tr>
<tr>
<td>ignoreSessions</td>
<td>Optional. Boolean value specifying whether WLST should drop all HTTP sessions immediately or wait for HTTP sessions to complete or time out while suspending. This argument defaults to <code>false</code>, indicating that HTTP sessions must complete or time out.</td>
</tr>
<tr>
<td>timeOut</td>
<td>Optional. Time (in seconds) the WLST waits for the server to complete in-process work before suspending the server. This argument defaults to 0 seconds, indicating that there is no timeout.</td>
</tr>
</tbody>
</table>
### 3.9.6.3 Example

The following example suspends a Managed Server instance:

```
wls:/mydomain/serverConfig> suspend('managed1')
Server 'managed1' suspended successfully.
wls:/mydomain/serverConfig>
```

### 3.10 Node Manager Commands

Use the WLST Node Managers commands, listed in Table 3–11, to start, shut down, restart, and monitor WebLogic Server instances.

Note: Node Manager must be running before you can execute the commands within this category.

For more information about Node Manager, see "Using Node Manager" in the Oracle Fusion Middleware Node Manager Administrator’s Guide for Oracle WebLogic Server.

<table>
<thead>
<tr>
<th>This command...</th>
<th>Enables you to...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>nm</code></td>
<td>Determine whether WLST is connected to Node Manager.</td>
<td>Online</td>
</tr>
<tr>
<td><code>nmConnect</code></td>
<td>Connect WLST to Node Manager to establish a session.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td><code>nmDisconnect</code></td>
<td>Disconnect WLST from a Node Manager session.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td><code>nmEnroll</code></td>
<td>Enables the Node Manager on the current computer to manage servers in a specified WebLogic domain.</td>
<td>Online</td>
</tr>
<tr>
<td><code>nmGenBootStartupProps</code></td>
<td>Generates the Node Manager property files, <code>boot.properties</code> and <code>startup.properties</code>, for the specified server.</td>
<td>Online</td>
</tr>
<tr>
<td><code>nmKill</code></td>
<td>Kill the specified server instance that was started with Node Manager.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td><code>nmLog</code></td>
<td>Return the Node Manager log.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td><code>nmServerLog</code></td>
<td>Return the server output log of the server that was started with Node Manager.</td>
<td>Online or Offline</td>
</tr>
</tbody>
</table>
### 3.10.1 nm

**Command Category:** Node Manager Commands  
**Use with WLST:** Online or Offline

#### 3.10.1.1 Description

Determines whether WLST is connected to Node Manager. Returns `true` or `false` and prints a descriptive message. Node Manager must be running before you can execute this command.

In the event of an error, the command returns a `WLSTException`.

#### 3.10.1.2 Syntax

```
nm()
```

#### 3.10.1.3 Example

The following example indicates that WLST is currently connected to Node Manager that is monitoring `mydomain`.

```
wls:/mydomain/serverConfig> nm()
Currently connected to Node Manager that is monitoring the domain "mydomain"
wls:/mydomain/serverConfig>
```

The following example indicates that WLST is not currently connected to Node Manager.

```
wls:/mydomain/serverConfig> nm()
Not connected to any Node Manager
wls:/mydomain/serverConfig>
```

### 3.10.2 nmConnect

**Command Category:** Node Manager Commands  
**Use with WLST:** Online or Offline

#### 3.10.2.1 Description

Connects WLST to Node Manager to establish a session. After connecting to Node Manager, you can invoke any Node Manager commands via WLST. Node Manager must be running before you can execute this command.

---

**Table 3–11 (Cont.) Node Manager Commands for WLST Configuration**

<table>
<thead>
<tr>
<th>This command...</th>
<th>Enables you to...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>nmServerStatus</td>
<td>Return the status of the server that was started with Node Manager.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>nmStart</td>
<td>Start a server in the current WebLogic domain using Node Manager.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>nmVersion</td>
<td>Return the Node Manager version.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>startNodeManager</td>
<td>Starts Node Manager on the same computer that is running WLST.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>stopNodeManager</td>
<td>Stops Node Manager.</td>
<td>Online or Offline</td>
</tr>
</tbody>
</table>
Once connected, the WLST prompt displays as follows, where `domainName` indicates the name of the WebLogic domain that is being managed: `wls:/nm/domainName>`. If you then connect WLST to a WebLogic Server instance, the prompt is changed to reflect the WebLogic Server instance. You can use the `nm` command to determine whether WLST is connected to Node Manager, as described in Section 3.10.1, "nm".

In the event of an error, the command returns a `WLSTException`.

### 3.10.2.2 Syntax

```
nmConnect([username, password], [host], [port], [domainName], [domainDir] [nmType], [verbose])

nmConnect([userConfigFile, userKeyFile], [host], [port], [domainName], [domainDir] [nmType], [verbose])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
</table>
| `username`   | Username of the operator who is connecting WLST to Node Manager. The username defaults to `weblogic`.  
  **Note:** When running a server in production mode, you must specify the username and password explicitly on the command line to ensure that the appropriate username and password are used when connecting to Node Manager. |
| `password`   | Password of the operator who is connecting WLST to Node Manager. The password defaults to `welcome1`.  
  **Note:** When running a server in production mode, you must specify the username and password explicitly on the command line to ensure that the appropriate username and password are used when connecting to Node Manager. |
| `host`       | Optional. Host name of Node Manager. This argument defaults to `localhost`. |
| `port`       | Optional. Port number of Node Manager. This argument defaults to a value that is based on the Node Manager type, as follows:  
  - For `plain` type, defaults to 5556  
  - For `rsh` type, defaults to 514  
  - For `ssh` type, defaults to 22  
  - For `ssl` type, defaults to 5556 |
| `domainName` | Optional. Name of the WebLogic domain that you want to manage. This argument defaults to `mydomain`. |
| `domainDir`  | Optional. Path of the domain directory to which you want to save the Node Manager secret file (`nm_password.properties`) and `SerializedSystemIni.dat` file. This argument defaults to the directory in which WLST was started. |
### Example

The following example connects WLST to Node Manager to monitor the oamdomain domain using the default host and port numbers and plain Node Manager type.

```bash
wls:/myserver/serverConfig> nmConnect('weblogic', 'welcome1', 'localhost', '5555', 'oamdomain', 'c:/Oracle/Middleware/user_projects/domains/oamdomain','ssl')
Connecting to Node Manager Server ...
Successfully connected to Node Manager.
wls:/nm/oamdomain>
```

The following example connects WLST to a Node Manager Server instance using a user configuration and key file to provide user credentials.

```bash
wls:/myserver/serverConfig> nmConnect(userConfigFile='c:/myfiles/myuserconfigfile.secure', userKeyFile='c:/myfiles/myuserkeyfile.secure', host='172.18.137.82', port=26106, domainName='mydomain', domainDir='c:/myfiles/mydomain', mType='ssl')
Connecting to Node Manager Server ...
Successfully connected to Node Manager.
wls:/nm/mydomain>
```

### 3.10.3 nmDisconnect

Command Category: Node Manager Commands

Use with WLST: Online or Offline

WLST must be connected to Node Manager to run this command.
3.10.3.1 Description
Disconnects WLST from a Node Manager session.
In the event of an error, the command returns a WLSTException.

3.10.3.2 Syntax
nmDisconnect()

3.10.3.3 Example
The following example disconnects WLST from a Node Manager session.

```
wls:/nm/oamdomain> nmDisconnect()
Successfully disconnected from Node Manager
wls:/myserver/serverConfig>
```

3.10.4 nmEnroll
Command Category: Node Manager Commands
Use with WLST: Online

3.10.4.1 Description
Enrolls the machine on which WLST is currently running. WLST must be connected to an Administration Server to run this command; WLST does not need to be connected to Node Manager.

This command downloads the following files from the Administration Server:
- Node Manager secret file (nm_password.properties), which contains the encrypted username and password that is used for server authentication
- SerializedSystemIni.dat file

This command also updates the nodemanager.domains file under the WL_HOME/common/nodemanager directory with the domain information, where WL_HOME refers to the top-level installation directory for WebLogic Server.

You must run this command once per WebLogic domain per machine unless that domain shares the root directory of the Administration Server.

If the machine is already enrolled when you run this command, the Node Manager secret file (nm_password.properties) is refreshed with the latest information from the Administration Server.

In the event of an error, the command returns a WLSTException.

3.10.4.2 Syntax
nmEnroll([domainDir], [nmHome])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>domainDir</td>
<td>Optional. Path of the domain directory to which you want to save the Node Manager secret file (nm_password.properties) and SerializedSystemIni.dat file. This argument defaults to the directory in which WLST was started.</td>
</tr>
<tr>
<td>nmHome</td>
<td>Optional. Path to the Node Manager home. The nodemanager.domains file, containing the domain information, is written to this directory. This argument defaults to WL_HOME/common/nodemanager, where WL_HOME refers to the top-level installation directory for WebLogic Server.</td>
</tr>
</tbody>
</table>
3.10.4.3 Example
The following example enrolls the current machine with Node Manager and saves the Node Manager secret file (nm_password.properties) and SerializedSystemIni.dat file to c:/Oracle/Middleware/mydomain/common/nodemanager/nm_password.properties. The nodemanager.domains file is written to WL_HOME/common/nodemanager by default.

wls:/mydomain/serverConfig> nmEnroll('c:/Oracle/Middleware/mydomain/common/nodemanager')
Enrolling this machine with the domain directory at c:\Oracle\Middleware\mydomain\common\nodemanager....
Successfully enrolled this machine with the domain directory at C:\Oracle\Middleware\mydomain\common\nodemanager
wls:/mydomain/serverConfig>

3.10.5 nmGenBootStartupProps
Command Category: Node Manager Commands
Use with WLST: Online

3.10.5.1 Description
Generates the Node Manager property files, boot.properties and startup.properties, for the specified server. The Node Manager property files are stored relative to the root directory of the specified server. The target root directory must be on the same machine on which you are running the command.

You must specify the name of a server; otherwise, the command will fail.

In the event of an error, the command returns a WLSTException.

3.10.5.2 Syntax
nmGenBootStartupProps(serverName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverName</td>
<td>Name of the server for which Node Manager property files are generated.</td>
</tr>
</tbody>
</table>

3.10.5.3 Example
The following example generates boot.properties and startup.properties in the root directory of the specified server, ms1.

wls:/mydomain/serverConfig> nmGenBootStartupProps('ms1')
Successfully generated boot.properties at c:\Oracle\Middleware\mydomain\servers\ms1\data\nodemanager\boot.properties
Successfully generated startup.properties at c:\Oracle\Middleware\mydomain\servers\ms1\data\nodemanager\startup.properties
wls:/mydomain/serverConfig>

3.10.6 nmKill
Command Category: Node Manager Commands
Use with WLST: Online or Offline
WLST must be connected to Node Manager to run this command.
3.10.6.1 Description
Kills the specified server instance that was started with Node Manager.

If you do not specify a server name using the `serverName` argument, the argument defaults to `myServer`, which must match your server name or the command will fail.

If you attempt to kill a server instance that was not started using Node Manager, the command displays an error.

In the event of an error, the command returns a `WLSTException`.

3.10.6.2 Syntax
```
mrKill([serverName], [serverType])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>serverName</code></td>
<td>Optional. Name of the server to be killed. This argument defaults to <code>myServer</code>.</td>
</tr>
<tr>
<td><code>serverType</code></td>
<td>Optional. The type of server to start. This argument defaults to <code>WebLogic</code>. Another valid option is <code>Coherence</code>.</td>
</tr>
</tbody>
</table>

3.10.6.3 Example
The following example kills the server named `oamserver`.

```
wls:/nm/oamdomain> nmrKill('oamserver')
Killing server 'oamserver' ...
Server oamServer killed successfully.
wls:/nm/oamdomain>
```

3.10.7 `nmLog`
Command Category: Node Manager Commands
Use with WLST: Online or Offline
WLST must be connected to Node Manager to run this command.

3.10.7.1 Description
Returns the Node Manager log.

In the event of an error, the command returns a `WLSTException`.

3.10.7.2 Syntax
```
mrLog([writer])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>writer</code></td>
<td>Optional. <code>java.io.Writer</code> object to which you want to stream the log output. This argument defaults to the WLST writer stream.</td>
</tr>
</tbody>
</table>

3.10.7.3 Example
The following example displays the Node Manager log.

```
wls:/nm/oamdomain> nmrLog()
Successfully retrieved the Node Manager log and written.
wls:/nm/oamdomain>
```
3.10.8 nmServerLog

Command Category: Node Manager Commands

Use with WLST: Online or Offline
WLST must be connected to Node Manager to run this command.

3.10.8.1 Description
Returns the server output log of the server that was started with Node Manager.
In the event of an error, the command returns a WLSTException.

3.10.8.2 Syntax
nmServerLog([serverName], [writer], [serverType])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverName</td>
<td>Optional. Name of the server for which you want to display the server output log. This argument defaults to myserver.</td>
</tr>
<tr>
<td>writer</td>
<td>Optional. java.io.Writer object to which you want to stream the log output. This argument defaults to the WLSTInterpreter standard out, if not specified.</td>
</tr>
<tr>
<td>serverType</td>
<td>Optional. The type of server to start. This argument defaults to WebLogic. Another valid option is Coherence.</td>
</tr>
</tbody>
</table>

3.10.8.3 Example
The following example displays the server output log for the oamserver server and writes the log output to myWriter.

```
wlsh:/nm/oamdomain> nmServerLog('oamserver',myWriter)
Successfully retrieved the server log and written.
wlsh:/nm/oamdomain>
```

3.10.9 nmServerStatus

Command Category: Node Manager Commands

Use with WLST: Online or Offline
WLST must be connected to Node Manager to run this command.

3.10.9.1 Description
Returns the status of the server that was started with Node Manager.
In the event of an error, the command returns a WLSTException.

3.10.9.2 Syntax
nmServerStatus([serverName], [serverType])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverName</td>
<td>Optional. Name of the server for which you want to display the status. This argument defaults to myserver.</td>
</tr>
<tr>
<td>serverType</td>
<td>Optional. The type of server to start. This argument defaults to WebLogic. Another valid option is Coherence.</td>
</tr>
</tbody>
</table>
3.10.9.3 Example
The following example displays the status of the server named oamserver, which was started with Node Manager.

```
wls:/nm/oamdomain> nmServerStatus('oamserver')
RUNNING
wls:/nm/oamdomain>
```

3.10.10 nmStart

Command Category: Node Manager Commands
Use with WLST: Online or Offline
WLST must be connected to Node Manager to run this command.

3.10.10.1 Description
Starts a server in the current WebLogic domain using Node Manager.

In the event of an error, the command returns a WLSTException.

**Note:** boot.properties must exist in order to start a server with nmStart. If this is the first time you are starting a server, you must manually create it in order to use nmStart.

Alternatively, you can use the nmStartprops argument to provide user credentials (after connecting to Node Manager):

```
prps = makePropertiesObject("username=weblogic, password=welcome1")
nmStart("AdminServer", props=prps)
```

3.10.10.2 Syntax

```
nmStart([serverName], [domainDir], [props], [writer], [serverType])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverName</td>
<td>Optional. Name of the server to be started.</td>
</tr>
<tr>
<td>domainDir</td>
<td>Optional. Domain directory of the server to be started. This argument defaults to the directory from which you started WLST.</td>
</tr>
<tr>
<td>props</td>
<td>Optional. System properties to apply to the new server.</td>
</tr>
<tr>
<td>writer</td>
<td>Optional. java.io.Writer object to which the server output is written. This argument defaults to the WLST writer.</td>
</tr>
<tr>
<td>serverType</td>
<td>Optional. The type of server to start. This argument defaults to WebLogic. Another valid option is Coherence.</td>
</tr>
</tbody>
</table>

3.10.10.3 Example
The following example starts the managed1 server in the current WebLogic domain using Node Manager.

```
wls:/nm/mydomain> nmStart("managed1")
Starting server managed1 ...
Server managed1 started successfully
wls:/nm/mydomain>
```
The following example starts the Administration Server in the specified WebLogic domain using Node Manager. In this example, the `prps` variable stores the system property settings and is passed to the command using the `props` argument.

```
wls:/nm/mydomain> prps = makePropertiesObject("weblogic.ListenPort=8001")
wls:/nm/mydomain> nmStart("AdminServer",props=prps)
Starting server AdminServer...
Server AdminServer started successfully
wls:/nm/mydomain>
```

### 3.10.11 nmVersion

Command Category: Node Manager Commands

Use with WLST: Online or Offline

WLST must be connected to Node Manager to run this command.

#### 3.10.11.1 Description

Returns the Node Manager version.

In the event of an error, the command returns a `WLSTException`.

#### 3.10.11.2 Syntax

```
nmVersion()
```

#### 3.10.11.3 Example

The following example displays the Node Manager version.

```
wls:/nm/oamdomain> nmVersion()
The Node Manager version that you are currently connected to is 9.0.0.0
wls:/nm/oamdomain>
```

### 3.10.12 startNodeManager

Command Category: Node Manager Commands

Use with WLST: Online or Offline

#### 3.10.12.1 Description

Starts Node Manager on the same computer that is running WLST.

---

**Notes:**  The WebLogic Server custom installation process optionally installs and starts Node Manager as a Windows service on Windows systems. For more information, see "About Installing Node Manager as a Windows Service" in the *Oracle Fusion Middleware Installation Guide for Oracle WebLogic Server*. In this case, you do not need to start the Node Manager manually.

In production environments, Oracle recommends that you do not use the `startNodeManager` command to start Node Manager. The recommended approach is to install Node Manager as a service or daemon, or to use the `startNodeManager` script (`startNodeManager.sh` or `startNodeManager.cmd`).
If Node Manager is already running when you invoke the `startNodeManager` command, the following message is displayed:

A Node Manager has already been started. Cannot start another Node Manager process via WLST

In the event of an error, the command returns a `WLSTException`.

### 3.10.12.2 Syntax

```bash
startNodeManager([verbose], [nmProperties])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>verbose</code></td>
<td>Optional. Boolean value specifying whether WLST starts Node Manager in verbose mode. This argument defaults to <code>false</code>, disabling verbose mode.</td>
</tr>
<tr>
<td><code>nmProperties</code></td>
<td>Optional. Comma-separated list of Node Manager properties, specified as name-value pairs. Node Manager properties include, but are not limited to, the following: <code>NodeManagerHome, ListenAddress, ListenPort, and PropertiesFile</code>.</td>
</tr>
</tbody>
</table>

#### 3.10.12.3 Example

The following example displays the Node Manager server version.

```bash
wls:/mydomain/serverConfig> startNodeManager(verbos='true', NodeManagerHome='c:/Oracle/Middleware/wlserver_10.3/common/nodemanager', ListenPort='6666', ListenAddress='myhost'))
Launching Node Manager ...
Successfully launched the Node Manager.
The Node Manager process is running independent of the WLST process
Exiting WLST will not stop the Node Manager process. Please refer to the Node Manager logs for more information.
The Node Manager logs will be under c:\Oracle\Middleware\wlserver_10.3\common\nodemanager.
wls:/mydomain/serverConfig>
```

### 3.10.13 stopNodeManager

Command Category: Node Manager Commands

Use with WLST: Online or Offline

#### 3.10.13.1 Description

Stops the Node Manager process.

---

**Note:** In order to stop the Node Manager process, you must have either started Node Manager with `startNodeManager`, or Node Manager must have been started with the property `QuitEnabled=true`. You can configure this property in `$WLS_HOME/common/nodemanager.properties`. This allows you to connect to the Node Manager to shut it down.

---

If the Node Manager is not running when you invoke the `stopNodeManager` command, the following message is displayed:

Cannot stop the Node Manager unless you are connected to it.
3.10.13.2 Syntax
stopNodeManager()

3.10.13.3 Example
The following example stops the Node Manager process for the base_domain domain.

```
wls:/nm/base_domain> stopNodeManager()
Stopped Node Manager Process successfully
wls:/offline>
```

3.11 Tree Commands

Use the WLST tree commands, listed in Table 3–12, to navigate among MBean hierarchies.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>custom</td>
<td>Navigate to the root of custom MBeans that are registered in the server.</td>
<td>Online</td>
</tr>
<tr>
<td>domainConfig</td>
<td>Navigate to the last MBean to which you navigated in the domain configuration hierarchy or to the root of the hierarchy, DomainMBean.</td>
<td>Online</td>
</tr>
<tr>
<td>domainCustom</td>
<td>Navigate to the root of custom MBeans that are registered in the Domain Runtime MBean Server</td>
<td>Online</td>
</tr>
<tr>
<td>domainRuntime</td>
<td>Navigate to the last MBean to which you navigated in the domain runtime hierarchy or to the root of the hierarchy, DomainRuntimeMBean.</td>
<td>Online</td>
</tr>
<tr>
<td>edit</td>
<td>Navigate to the last MBean to which you navigated in the edit configuration MBean hierarchy or to the root of the hierarchy, DomainMBean.</td>
<td>Online</td>
</tr>
<tr>
<td>jndi</td>
<td>Navigates to the JNDI tree for the server to which WLST is currently connected.</td>
<td>Online</td>
</tr>
<tr>
<td>serverConfig</td>
<td>Navigate to the last MBean to which you navigated in the configuration MBean hierarchy or to the root of the hierarchy, DomainMBean.</td>
<td>Online</td>
</tr>
<tr>
<td>serverRuntime</td>
<td>Navigate to the last MBean to which you navigated in the runtime MBean hierarchy or to the root of the hierarchy, ServerRuntimeMBean.</td>
<td>Online</td>
</tr>
</tbody>
</table>

3.11.1 custom

Command Category: Tree Commands

Use with WLST: Online

3.11.1.1 Description
Navigates to the root of custom MBeans that are registered in the Runtime MBean Server. WLST navigates, interrogates, and edits custom MBeans as it does domain MBeans; however, custom MBeans cannot use the cmo variable because a stub is not available.
The `custom` command is available when WLST is connected to an Administration Server instance or a Managed Server instance. When connected to a WebLogic Integration or WebLogic Portal server, WLST can interact with all the WebLogic Integration or WebLogic Portal server MBeans.

For more information about custom MBeans, see *Oracle Fusion Middleware Developing Custom Management Utilities With JMX for Oracle WebLogic Server*.

In the event of an error, the command returns a `WLSTException`.

---

**Note:** When navigating to the `custom` tree, WLST queries all MBeans in the compatibility MBean server, the runtime MBean server, and potentially the JVM platform MBean server to locate the custom MBeans. Depending on the number of MBeans in the current WebLogic domain, this process may take a few minutes, and WLST may not return a prompt right away.

---

**3.11.2 Syntax**

custom()

**3.11.3 Example**

The following example navigates from the configuration MBean hierarchy to the custom MBean hierarchy on a Administration Server instance.

```
wlst:/mydomain/serverConfig> custom()
Location changed to custom tree. This is a writeable tree with No root. For more help, use help('custom')
wls:/mydomain/custom>
```

---

**3.11.2 domainConfig**

Command Category: Tree Commands

Use with WLST: Online

**3.11.2.1 Description**

Navigates to the last MBean to which you navigated in the domain Configuration hierarchy or to the root of the hierarchy, `DomainMBean`. This read-only hierarchy stores the configuration MBeans that represent your current WebLogic domain.

In the event of an error, the command returns a `WLSTException`.

**3.11.2.2 Syntax**

domainConfig()

**3.11.2.3 Example**

The following example navigates from the configuration MBean hierarchy to the WebLogic domain Configuration hierarchy on an Administration Server instance.

```
wls:/mydomain/serverConfig> domainConfig()
```
Location changed to domainConfig tree. This is a read-only tree with DomainMBean as the root.
For more help, use help('domainConfig')
wls:/mydomain/domainConfig> ls()
dr-- AppDeployments
dr-- BridgeDestinations
dr-- Clusters
dr-- CustomResources
dr-- DeploymentConfiguration
dr-- Deployments
dr-- EmbeddedLDAP
dr-- ErrorHandlings
dr-- FileStores
dr-- InternalAppDeployments
dr-- InternalLibraries
dr-- JDBCDataSourceFactories
dr-- JDBCStores
dr-- JDBCSystemResources
dr-- JMSBridgeDestinations
dr-- JMSInteropModules
dr-- JMSServers
dr-- JMSSystemResources
...
wls:/mydomain/domainConfig>

3.11.3 domainCustom

Command Category: Tree Commands
Use with WLST: Online

3.11.3.1 Description
Navigates to the domain custom tree of custom MBeans that are registered in the Domain Runtime MBean Server. WLST navigates, interrogates, and edits domain custom MBeans as it does domain MBeans; however, domain custom MBeans cannot use the cmo variable because a stub is not available.

Note: When navigating to the domainCustom tree, WLST queries all MBeans in the Domain Runtime MBean Server, the Runtime MBean Servers on each server, and potentially the JVM platform MBean server to locate the custom MBeans. Depending on the number of MBeans in the current WebLogic domain, this process may take a few minutes, and WLST may not return a prompt right away. It is recommended that a JMX query Object Name Pattern be specified to limit the amount of searching performed.

The domainCustom command is available only when WLST is connected to an Administration Server instance.

For more information about the Domain Runtime MBean Server, see "Understanding WebLogic Server MBeans" in Oracle Fusion Middleware Developing Custom Management Utilities With JMX for Oracle WebLogic Server.

In the event of an error, the command returns a WLSTException.
3.11.3.2 Syntax

\[ \text{domainCustom}(\text{ObjectNamePattern}) \]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ObjectNamePattern</td>
<td>A JMX query pattern, such as sip:* or <em>:</em> The default value is null or <em>:</em>.*</td>
</tr>
</tbody>
</table>

3.11.3.3 Example

The following example navigates from the configuration MBean hierarchy to the domain custom MBean hierarchy on an Administration Server instance:

\[ wls:/mydomain/serverConfig> \text{domainCustom()} \]

Location changed to domain custom tree. This is a writeable tree with No root. For more help, use help('domainCustom').

\[ wls:/mydomain/domainCustom \]

3.11.4 domainRuntime

Command Category: Tree Commands

Use with WLST: Online

3.11.4.1 Description

Navigates to the last MBean to which you navigated in the domain Runtime hierarchy or to the root of the hierarchy, DomainRuntimeMBean. This read-only hierarchy stores the runtime MBeans that represent your current WebLogic domain.

In the event of an error, the command returns a WLSTException.

3.11.4.2 Syntax

\[ \text{domainRuntime}() \]

3.11.4.3 Example

The following example navigates from the configuration MBean hierarchy to the domain Runtime hierarchy on an Administration Server instance.

\[ wls:/mydomain/serverConfig> \text{domainRuntime()}
\]

\[ wls:/mydomain/domainRuntime> \text{ls()}
\]

| dr--  | AppRuntimeStateRuntime |
| dr--  | DeployerRuntime        |
| dr--  | DomainServices         |
| dr--  | LogRuntime             |
| dr--  | MessageDrivenControlEJBRuntime |
| dr--  | MigratableServiceCoordinatorRuntime |
| dr--  | MigrationDataRuntimes  |
| dr--  | SNMPAgentRuntime       |
| dr--  | ServerLifeCycleRuntimes|
| dr--  | ServerRuntimes         |
| dr--  | ServerServices         |
| -r--  | ActivationTime         | Mon Aug 01 11:41:25 EDT 2005 |
| -r--  | Clusters               | null                        |
| -r--  | MigrationDataRuntimes  | null                        |
| -r--  | Name                   | sampleMedRecDomain          |
| -rw-  | Parent                 | null                        |
| -r--  | SNMPAgentRuntime       | null                        |
Command Category: Tree Commands

### 3.11.5 Description
Navigates to the last MBean to which you navigated in the edit configuration MBean hierarchy or to the root of the hierarchy, DomainMBean. This writable hierarchy stores all of the configuration MBeans that represent your current WebLogic domain.

---

**Note:** To edit configuration beans, you must be connected to an Administration Server. If you connect to a Managed Server, WLST functionality is limited to browsing the configuration bean hierarchy. While you cannot use WLST to change the values of MBeans on Managed Servers, it is possible to use the Management APIs to do so. Oracle recommends that you change only the values of configuration MBeans on the Administration Server. Changing the values of MBeans on Managed Servers can lead to an inconsistent domain configuration.

For more information about editing configuration beans, see "Using WLST Online to Update an Existing Domain" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

---

In the event of an error, the command returns a WLSTException.

### 3.11.5.2 Syntax
edit()

### 3.11.5.3 Example
The following example illustrates how to navigate from the server configuration MBean hierarchy to the editable copy of the domain configuration MBean hierarchy, in an Administration Server instance.

```
wlsh:/myserver/serverConfig> edit()
Location changed to edit tree. This is a writeable tree with DomainMBean as the root.
For more help, use help('edit')
wls:/myserver/edit !> ls()
dr-- AppDeployments
dr-- BridgeDestinations
dr-- Clusters
dr-- DeploymentConfiguration
dr-- Deployments
dr-- EmbeddedLDAP
...
wls:/myserver/edit !>
```
### 3.11.6 jndi

Command Category: Tree Commands

Use with WLST: Online

#### 3.11.6.1 Description
Navigates to the JNDI tree for the server to which WLST is currently connected. This read-only tree holds all the elements that are currently bound in JNDI.

In the event of an error, the command returns a WLSTException.

#### 3.11.6.2 Syntax
```
jndi()
```

#### 3.11.6.3 Example
The following example navigates from the runtime MBean hierarchy to the Domain JNDI tree on an Administration Server instance.
```
wls:/myserver/runtime> jndi()
Location changed to jndi tree. This is a read-only tree with No root. For more help, use help('jndi')
wls:/myserver/jndi> ls()
dr--  .ejb
dr--   javax
dr--   jms
dr--   weblogic
...
```

### 3.11.7 serverConfig

Command Category: Tree Commands

Use with WLST: Online

#### 3.11.7.1 Description
Navigates to the last MBean to which you navigated in the configuration MBean hierarchy or to the root of the hierarchy, DomainMBean.

This read-only hierarchy stores the configuration MBeans that represent the server to which WLST is currently connected. The MBean attribute values include any command-line overrides that a user specified while starting the server.

In the event of an error, the command returns a WLSTException.

For more information, see "Navigating Among MBean Hierarchies" in Oracle Fusion Middleware Oracle WebLogic Scripting Tool.

#### 3.11.7.2 Syntax
```
serverConfig()
```

#### 3.11.7.3 Example
The following example navigates from the domain runtime MBean hierarchy to the configuration MBean hierarchy on an Administration Server instance.
```
wls:/mydomain/domainRuntime> serverConfig()
wls:/mydomain/serverConfig>
```
3.11.8 serverRuntime

Command Category: Tree Commands
Use with WLST: Online

3.11.8.1 Description
Navigates to the last MBean to which you navigated in the runtime MBean hierarchy or to the root of the hierarchy, ServerRuntimeMBean. This read-only hierarchy stores the runtime MBeans that represent the server to which WLST is currently connected.

In the event of an error, the command returns a WLSTException.

3.11.8.2 Syntax
serverRuntime()

3.11.8.3 Example
The following example navigates from the configuration MBean hierarchy to the runtime MBean hierarchy on an Administration Server instance.

wls:/mydomain/serverConfig> serverRuntime()
Location changed to serverRuntime tree. This is a read-only tree with ServerRuntimeMBean as the root.
For more help, use help('serverRuntime')
wls:/mydomain/serverRuntime>

3.12 WLST Variable Reference

Table 3–13 describes WLST variables and their common usage. All variables are initialized to default values at the start of a user session and are changed according to the user interaction with WLST.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>cmgr</td>
<td>The cmgr variable is set to the ConfigurationManagerMBean. You can use this variable to get the current value of any ConfigurationManagerMBean attribute.</td>
<td>wls:/mydomain/edit&gt; cmgr.getCurrentEditor() 'weblogic'</td>
</tr>
<tr>
<td>cmo</td>
<td>Current Management Object. The cmo variable is set to the bean instance to which you navigate using WLST. You can use this variable to perform any get, set, or invoke method on the current bean instance. WLST sets the variable to the current WLST path. For example, when you change to the serverConfig hierarchy, cmo is set to DomainMBean. When you change to the serverRuntime hierarchy, cmo is set to ServerRuntimeMBean. The variable is available in all WLST hierarchies except custom and jndi.</td>
<td>wls:/mydomain/edit&gt; cmo.setAdministrationPort(9092)</td>
</tr>
<tr>
<td>connected</td>
<td>Boolean value specifying whether WLST is connected to a running server. WLST sets this variable to true when connected to a running server; otherwise, WLST sets it to false.</td>
<td>wls:/mydomain/serverConfig&gt; print connected false</td>
</tr>
</tbody>
</table>
### Table 3–13 (Cont.) WLST Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>domainName</td>
<td>Name of the WebLogic domain to which WLST is connected.</td>
<td><code>wls:/mydomain/serverConfig&gt; print domainName mydomain</code></td>
</tr>
<tr>
<td>domainRuntimeService</td>
<td>DomainRuntimeServiceMBean. This variable is available only when WLST is connected to the Administration Server.</td>
<td><code>wls:/mydomain/serverConfig&gt; domainService.getServerName() 'myserver'</code></td>
</tr>
<tr>
<td>editService</td>
<td>EditServiceMBean. This variable is available only when WLST is connected to the Administration Server.</td>
<td><code>wls:/mydomain/edit&gt; dc = editService.getDomainConfiguration()</code></td>
</tr>
<tr>
<td>exitonerror</td>
<td>Boolean value specifying whether WLST terminates script execution when it encounters an exception. This variable defaults to true, indicating that script execution is terminated when WLST encounters an error. This variable is not applicable when running WLST in interactive mode.</td>
<td><code>wls:/mydomain/serverConfig&gt; print exitonerror true</code></td>
</tr>
<tr>
<td>home</td>
<td>Represents the local MBeanHome.</td>
<td><code>wls:/mydomain/serverConfig&gt; print home</code></td>
</tr>
<tr>
<td>isAdminServer</td>
<td>Boolean value specifying whether WLST is connected to a WebLogic Administration Server instance. WLST sets this variable to true if WLST is connected to a WebLogic Administration Server; otherwise, WLST sets it to false.</td>
<td><code>wls:/mydomain/serverConfig&gt; print isAdminServer true</code></td>
</tr>
<tr>
<td>mbs</td>
<td>MBeanServerConnection object that corresponds to the current location in the hierarchy.</td>
<td><code>wls:/mydomain/serverConfig&gt; print mbs</code></td>
</tr>
<tr>
<td>recording</td>
<td>Boolean value specifying whether WLST is recording commands. WLST sets this variable to true when the startRecording command is entered; otherwise, WLST sets this variable to false.</td>
<td><code>wls:/mydomain/serverConfig&gt; print recording true</code></td>
</tr>
<tr>
<td>runtimeService</td>
<td>RuntimeServiceMBean MBean.</td>
<td><code>wls:/mydomain/serverConfig&gt; print runtimeService.getServerRuntime()</code></td>
</tr>
<tr>
<td>serverName</td>
<td>Name of the server to which WLST is connected.</td>
<td><code>wls:/mydomain/serverConfig&gt; print serverName myserver</code></td>
</tr>
<tr>
<td>typeService</td>
<td>TypeServiceMBean MBean.</td>
<td><code>wls:/mydomain/serverConfig&gt; print typeService</code></td>
</tr>
<tr>
<td>username</td>
<td>Name of user currently connected to WLST.</td>
<td><code>wls:/mydomain/serverConfig&gt; print username</code></td>
</tr>
<tr>
<td>version</td>
<td>Current version of the running server to which WLST is connected.</td>
<td><code>wls:/mydomain/serverConfig&gt; print version</code></td>
</tr>
</tbody>
</table>

**WebLogic Server 9.0 Thu Aug 31 12:15:50 PST 2005 778899**
This chapter provides detailed descriptions of custom WLST commands for infrastructure security, including command syntax, arguments and command examples.

**Note:** If you installed an IDM 11gR2 release, for supported WLST commands, refer to the appropriate WLST command reference in the Oracle online library for your release. For IDM 11gR2 releases prior to PS3, refer to “Infrastructure Security Custom WLST Commands” in *WebLogic Scripting Tool Command Reference*. For IDM 11gR2 PS3 or higher releases, refer to *WebLogic Scripting Tool Command Reference for Identity and Access Management*.

The following sections describe the Oracle Fusion Middleware Infrastructure Security custom WLST commands in detail. Topics include:

- Section 4.1, "Overview of WLST Security Commands"
- Section 4.2, "Audit Configuration Commands"
- Section 4.3, "SSL Configuration Commands"
- Section 4.4, "Oracle Identity Federation Commands"
- Section 4.5, "Directory Integration Platform Commands"
- Section 4.6, "Security Commands"
- Section 4.7, "Oracle Access Manager Commands"
- Section 4.8, "Oracle Security Token Service"
- Section 4.9, "Oracle Keystore Service"
- Section 4.10, "Library Oracle Virtual Directory (libOVD) Commands"
- Section 4.11, "Identity Directory Service Commands"

For additional information about Oracle Platform Security Services, see *Oracle Fusion Middleware Application Security Guide*. 
4.1 Overview of WLST Security Commands

WLST security commands are divided into the following categories:

<table>
<thead>
<tr>
<th>Command Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audit Configuration Commands</td>
<td>View and manage audit policies and the audit repository configuration.</td>
</tr>
<tr>
<td>SSL Configuration Commands</td>
<td>View and manage wallets, JKS keystores, and SSL configuration for Oracle HTTP Server, Oracle WebCache, Oracle Internet Directory, and Oracle Virtual Directory components.</td>
</tr>
<tr>
<td>Oracle Identity Federation Commands</td>
<td>View and manage configuration for Oracle Identity Federation</td>
</tr>
<tr>
<td>Directory Integration Platform Commands</td>
<td>For information on DIP tools, see &quot;Directory Integration Platform Tools&quot; in the Oracle Fusion Middleware Reference for Oracle Identity Management</td>
</tr>
<tr>
<td>Security Commands</td>
<td>Manage domain and credential domain stores and migrate domain policy store.</td>
</tr>
<tr>
<td>Oracle Access Manager Commands</td>
<td>Manage OAM-related components, such as authorization providers, identity asserters, and SSO providers.</td>
</tr>
<tr>
<td>Library Oracle Virtual Directory (libOVD) Commands</td>
<td>View and manage Library Oracle Virtual Directory (libOVD) configurations associated with a particular OPSS context.</td>
</tr>
<tr>
<td>Identity Directory Service Commands</td>
<td>Manage Identity Directory Service entity attributes, entity definitions, relationships, and default operational configurations.</td>
</tr>
</tbody>
</table>

4.2 Audit Configuration Commands

Use the WLST commands listed in Table 4-2 to view and manage audit policies and the audit repository configuration.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>getNonJavaEEAuditMBeanName</td>
<td>Display the mBean name for a non-Java EE component.</td>
<td>Online</td>
</tr>
<tr>
<td>getAuditPolicy</td>
<td>Display audit policy settings.</td>
<td>Online</td>
</tr>
<tr>
<td>setAuditPolicy</td>
<td>Update audit policy settings.</td>
<td>Online</td>
</tr>
<tr>
<td>getAuditRepository</td>
<td>Display audit repository settings.</td>
<td>Online</td>
</tr>
<tr>
<td>setAuditRepository</td>
<td>Update audit repository settings.</td>
<td>Online</td>
</tr>
<tr>
<td>listAuditEvents</td>
<td>List audit events for one or all components.</td>
<td>Online</td>
</tr>
<tr>
<td>exportAuditConfig</td>
<td>Export a component's audit configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>importAuditConfig</td>
<td>Import a component's audit configuration.</td>
<td>Online</td>
</tr>
</tbody>
</table>
### Audit Configuration Commands

For more information, see the *Oracle Fusion Middleware Application Security Guide*.

#### 4.2.1 `getNonJavaEEAuditMBeanName`

Online command that displays the mbean name for non-Java EE components.

##### 4.2.1.1 Description

This command displays the mbean name for non-Java EE components given the instance name, component name, component type, and the name of the Oracle WebLogic Server on which the component's audit mbean is running. The mbean name is a required parameter to other audit WLST commands when managing a non-Java EE component.

##### 4.2.1.2 Syntax

`getNonJavaEEAuditMBeanName(instName, compName, compType, svrName)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>instName</code></td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td><code>compName</code></td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td><code>compType</code></td>
<td>Specifies the type of component. Valid values are ohs, oid, ovd, and WebCache.</td>
</tr>
<tr>
<td><code>svrName</code></td>
<td>Specifies the name of the Oracle WebLogic Server.</td>
</tr>
</tbody>
</table>

##### 4.2.1.3 Example

The following interactive command displays the mBean name for an Oracle Internet Directory:

```
wls:/mydomain/serverConfig> getNonJavaEEAuditMBeanName(instName='inst1', compName='oid1', compType='oid', svrName='AdminServer')
```

#### 4.2.2 `getAuditPolicy`

Online command that displays the audit policy settings.

##### 4.2.2.1 Description

This command displays audit policy settings including the filter preset, special users, custom events, maximum log file size, and maximum log directory size. The component mbean name is required for non-Java EE components like Oracle Internet Directory and Oracle Virtual Directory.
### 4.2.2.2 Syntax

`getAuditPolicy([mbeanName, componentType])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mbeanName</code></td>
<td>Specifies the name of the component audit MBean for non-Java EE components.</td>
</tr>
<tr>
<td><code>componentType</code></td>
<td>Requests the audit policy for a specific component registered in the audit store. If not specified, the audit policy in jps-config.xml is returned.</td>
</tr>
</tbody>
</table>

### 4.2.3 Examples

The following command displays the audit settings for a Java EE component:

```
    wls:/mydomain/serverConfig> getAuditPolicy()
    Location changed to domainRuntime tree. This is a read-only tree with DomainMBean as the root.
    For more help, use help(domainRuntime)

    FilterPreset:All
    Max Log File Size:104857600
    Max Log Dir Size:0
```

The following command displays the audit settings for MBean `CSAuditProxyMBean`:

```
    wls:/mydomain/serverConfig> getAuditPolicy(on='oracle.security.audit.test:type=CSAuditMBean, name=CSAuditProxyMBean')
```

### 4.2.3 setAuditPolicy

Online command that updates an audit policy.

#### 4.2.3.1 Description

Online command that configures the audit policy settings. You can set the filter preset, add or remove users, and add or remove custom events. The component mbean name is required for non-Java EE components like Oracle Internet Directory and Oracle Virtual Directory.

**Note:** You can obtain a non-Java EE component's MBean name using the `getNonJavaEEAuditMBeanName` command.

#### 4.2.3.2 Syntax

`setAuditPolicy([mbeanName], [filterPreset], [addSpecialUsers], [removeSpecialUsers], [addCustomEvents], [removeCustomEvents], [componentType], [maxDirSize], [maxFileSize], [andCriteria], [orCriteria], [componentEventsFile])`
4.2.3.3 Examples

The following interactive command sets audit policy to None level, and adds users user2 and user3 while removing user1 from the policy:

```
wls:/mydomain/serverConfig> setAuditPolicy (filterPreset='None',addSpecialUsers='user2,user3',removeSpecialUsers='user1')
```

```
wls:/mydomain/serverConfig> getAuditPolicy();
Already in Domain Runtime Tree
FilterPreset:None
Special Users:user2,user3
Max Log File Size:104857600
Max Log Dir Size:0
```

The following interactive command adds login events while removing logout events from the policy:

```
wls:/mydomain/serverConfig> setAuditPolicy(filterPreset='Custom',addCustomEvents='UserLogin',removeCustomEvents='UserLogout')
```

The following interactive command sets audit policy to a Low level:

```
wls:/IDMDomain/domainRuntime> setAuditPolicy(filterPreset='Low');
Already in Domain Runtime Tree
Audit Policy Information updated successfully
```

```
wls:/IDMDomain/domainRuntime> getAuditPolicy();
Already in Domain Runtime Tree
FilterPreset:Low
Max Log File Size:104857600
Max Log Dir Size:0
```
The following command sets a custom filter to audit the CheckAuthorization event:

```
setAuditPolicy(filterPreset='Custom',
addCustomEvents='JPS:CheckAuthorization');
```

Audit Policy Information updated successfully
```
getAuditPolicy();
```

FilterPreset:Custom
Special Users:user1
Max Log File Size:104857600
Max Log Dir Size:0
Custom Events:JPS:CheckAuthorization

### 4.2.4 getAuditRepository

Online command that displays audit repository settings.

#### 4.2.4.1 Description

This command displays audit repository settings for Java EE components and applications (for other components like Oracle Internet Directory, the repository configuration resides in opmn.xml). Also displays database configuration if the repository is a database type.

#### 4.2.4.2 Syntax

```
getAuditRepository
```

#### 4.2.4.3 Example

The following command displays audit repository configuration:

```
getAuditRepository()
```

Repository Type:File

### 4.2.5 setAuditRepository

Online command that updates audit repository settings.

#### 4.2.5.1 Description

This command sets the audit repository settings for Java EE components and applications (for other components like Oracle Internet Directory, the repository is configured by editing opmn.xml).

#### 4.2.5.2 Syntax

```
setAuditRepository([switchToDB],[dataSourceName],[interval])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>switchToDB</td>
<td>If true, switches the repository from file to database.</td>
</tr>
<tr>
<td>dataSourceName</td>
<td>Specifies the name of the data source.</td>
</tr>
</tbody>
</table>
4.2.5.3 Examples

The following command switches from a file repository to a database repository:

```
wl Belg:/IDMDomain/domainRuntime> setAuditRepository(switchToDB='true');
```

Audit Repository Information updated

```
wl Belg:/IDMDomain/domainRuntime> getAuditRepository();
```

JNDI Name:jdbc/AuditDB
Interval:15
Repository Type:DB

The following interactive command changes audit repository to a specific database
and sets the audit loader interval to 14 seconds:

```
wls:/mydomain/serverConfig>
setAuditRepository(switchToDB='true',dataSourceName='jdbcAuditDB',interval='14')
```

4.2.6 listAuditEvents

Online command that displays a component's audit events.

4.2.6.1 Description

This command displays a component's audit events and attributes. For non-Java EE
components, pass the component mbean name as a parameter. Java EE applications
and services like Oracle Platform Security Services (OPSS) do not need the mbean
parameter. Without a component type, all generic attributes applicable to all
components are displayed.

**Note:** You can obtain a non-Java EE component's MBean name using
the `getNonJavaEEAuditMBeanName` command.

4.2.6.2 Syntax

```
listAuditEvents([mbeanName],[componentType])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>mbeanName</td>
<td>Specifies the name of the component MBean.</td>
</tr>
<tr>
<td>componentType</td>
<td>Specifies the component type to limit the list to all events of the</td>
</tr>
<tr>
<td></td>
<td>component type.</td>
</tr>
</tbody>
</table>

4.2.6.3 Examples

The following command displays audit events for the Oracle Platform Security
Services component:

```
wls:/IDMDomain/domainRuntime> listAuditEvents(componentType='JPS');
```

Already in Domain Runtime Tree
Common Attributes
- ComponentType
  Type of the component. For MAS integrated SystemComponents this is the componentType
- InstanceId
  Name of the MAS Instance, that this component belongs to
- HostId
  DNS hostname of originating host
- HostNwaddr
  IP or other network address of originating host
- ModuleId
  ID of the module that originated the message. Interpretation is unique within Component ID.
- ProcessId
  ID of the process that originated the message

The following command displays audit events for Oracle HTTP Server:

```
wls:/mydomain/serverConfig> listAuditEvents(componentType='ohs')
```

The following command displays all audit events:

```
wls:/IDMDomain/domainRuntime> listAuditEvents();
```

Already in Domain Runtime Tree

Components:
- DIP
- JPS
- OIF
- OWSM-AGENT
- OWSM-PM-EJB
- ReportsServer
- WS-PolicyAttachment
- WebCache
- WebServices

Attributes applicable to all components:
- ComponentType
- InstanceId
- HostId
- HostNwaddr
- ModuleId
- ProcessId
- OracleHome
- HomeInstance
- ECID
- RID

...  

4.2.7 `exportAuditConfig`

Online command that exports a component’s audit configuration.

4.2.7.1 Description

This command exports the audit configuration to a file. For non-Java EE components, pass the component mbean name as a parameter. Java EE applications and services like Oracle Platform Security Services (OPSS) do not need the mbean parameter.
4.2.7.2 Syntax

```
exportAuditConfig([mbeanName],fileName, [componentType])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mbeanName</code></td>
<td>Specifies the name of the non-Java EE component MBean.</td>
</tr>
<tr>
<td><code>fileName</code></td>
<td>Specifies the path and file name to which the audit configuration should be exported.</td>
</tr>
<tr>
<td><code>componentType</code></td>
<td>Specifies that only events of the given component be exported to the file. If not specified, the audit configuration in <code>jps-config.xml</code> is exported.</td>
</tr>
</tbody>
</table>

4.2.7.3 Examples

The following interactive command exports the audit configuration for a component:

```
wls:/mydomain/serverConfig>
exportAuditConfig(one='oracle.security.audit.test:type=CSAuditMBean,
name=CSAuditProxyMBean',fileName='/tmp/auditconfig')
```

The following interactive command exports the audit configuration for a Java EE component; no mBean is specified:

```
wls:/mydomain/serverConfig> exportAuditConfig(fileName='/tmp/auditconfig')
```

4.2.8 importAuditConfig

Online command that imports a component’s audit configuration.

4.2.8.1 Description

This command imports the audit configuration from an external file. For non-Java EE components, pass the component mbean name as a parameter. Java EE applications and services like Oracle Platform Security Services (OPSS) do not need the mbean parameter.

```
importAuditConfig([mbeanName],fileName, [componentType])
```

Note: You can obtain a non-Java EE component's MBean name using the `getNonJavaEEAuditMBeanName` command.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>mbeanName</code></td>
<td>Specifies the name of the non-Java EE component MBean.</td>
</tr>
<tr>
<td><code>fileName</code></td>
<td>Specifies the path and file name from which the audit configuration should be imported.</td>
</tr>
<tr>
<td><code>componentType</code></td>
<td>Specifies that only events of the given component be imported from the file. If not specified, the audit configuration in <code>jps-config.xml</code> is imported.</td>
</tr>
</tbody>
</table>
4.2.8.3 Examples
The following interactive command imports the audit configuration for a component:

```
importAuditConfig(on='oracle.security.audit.test:type=CSAuditMBean,
name='CSAuditProxyMBean',fileName='/tmp/auditconfig')
```

The following interactive command imports the audit configuration from a file; no mBean is specified:

```
importAuditConfig(fileName='/tmp/auditconfig')
```

4.2.9 createAuditDBView

Creates a SQL script that can generate a view for audit in the database.

4.2.9.1 Description

This command generates a SQL script that you can use to create a database view of the audit definitions of a specified component. The script is written to the specified file and also printed out to the console.

Upon execution, the result of the SQL script depends on the audit model at your site:

- If using the 11.1.1.6.0 model, and the component is registered in the audit store, the script creates a view using the system component tables (IAU_COMMON, IAU_USERSESSION, IAU_AUDITSERVICE and IAU_CUSTOM) for the specified component.
- If using the pre-11.1.1.6.0 model, the component is not registered in the audit store but its event definitions reside in the component_events.xml file (in the oracle_common/modules/oracle.iau_11.1.1/components/<componentType> dir), and the view is created using the IAU_BASE and component tables.

4.2.9.2 Syntax

```
createAuditDBView(fileName, componentType)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>fileName</td>
<td>Specifies the path and file name to which the SQL script is written.</td>
</tr>
<tr>
<td>componentType</td>
<td>The component whose definitions are the basis of the view.</td>
</tr>
</tbody>
</table>

4.2.9.3 Example

```
createAuditDBView(fileName="/tmp/JPSAuditView.sql", componentType="JPS")
```

4.2.10 listAuditComponents

Lists components that can be audited.

4.2.10.1 Description

This command creates a list of the components that can be audited. It lists components registered in the audit store using both the 11.1.1.6.0 model and the pre-11.1.1.6.0 model.
4.2.10.2 Syntax

listAuditComponents(fileName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>fileName</td>
<td>Specifies the path and file name to which the output is written.</td>
</tr>
</tbody>
</table>

4.2.10.3 Example

listAuditComponents(fileName = "/tmp/complist.txt")

4.2.11 registerAudit

Registers the specified component in the audit store.

4.2.11.1 Description

Adds the event definition and translation content for a specified component to the audit store. If you try to register using the pre-11.1.1.6.0 audit XML schema definition, it is upgraded to the 11.1.1.6.0 XML schema definition and then registered with the audit store.

4.2.11.2 Syntax

registerAudit(xmlFile, [xlfFile], componentType, [mode=OVERWRITE|UPGRADE])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>xmlFile</td>
<td>Specifies the Component Event definition file.</td>
</tr>
<tr>
<td>xlfFile</td>
<td>Specifies the component xlf jar file. Optional.</td>
</tr>
<tr>
<td>componentType</td>
<td>Specifies the component to be registered.</td>
</tr>
<tr>
<td>mode</td>
<td>OVERWRITE or UPGRADE. Default is UPGRADE.</td>
</tr>
</tbody>
</table>

4.2.11.3 Example

wls:/mydomain/serverConfig>registerAudit(xmlFile="/tmp/comp.xml", xmlFile="/tmp/comp_xlf.jar", componentType="AuditApp", mode="UPGRADE")

4.2.12 deregisterAudit

Removes the event definition and translation content for the specified component from the audit store.

4.2.12.1 Description

Removes an existing event definition and translation content for a specified component or application from the audit store.

4.2.12.2 Syntax

deregisterAudit(componentType)
4.3 SSL Configuration Commands

Use the WLST commands listed in Table 4–3 to view and manage SSL configuration for Oracle Fusion Middleware components.

Table 4–3  WLST Commands for SSL Configuration

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>addCertificateRequest</td>
<td>Generate a certificate signing request in an Oracle wallet.</td>
<td>Online</td>
</tr>
<tr>
<td>addSelfSignedCertificate</td>
<td>Add a self-signed certificate to an Oracle wallet.</td>
<td>Online</td>
</tr>
<tr>
<td>changeKeyStorePassword</td>
<td>Change the password to a JKS keystore.</td>
<td>Online</td>
</tr>
<tr>
<td>changeWalletPassword</td>
<td>Change the password to an Oracle wallet.</td>
<td>Online</td>
</tr>
<tr>
<td>configureSSL</td>
<td>Set the SSL attributes for a component listener.</td>
<td>Online</td>
</tr>
<tr>
<td>createKeyStore</td>
<td>Create a JKS keystore.</td>
<td>Online</td>
</tr>
<tr>
<td>createWallet</td>
<td>Create an Oracle wallet.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteKeyStore</td>
<td>Delete a JKS keystore.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteWallet</td>
<td>Delete an Oracle wallet.</td>
<td>Online</td>
</tr>
<tr>
<td>exportKeyStore</td>
<td>Export a JKS keystore to a file.</td>
<td>Online</td>
</tr>
<tr>
<td>exportKeyStoreObject</td>
<td>Export an object from a JKS keystore to a file.</td>
<td>Online</td>
</tr>
<tr>
<td>exportWallet</td>
<td>Export an Oracle wallet to a file.</td>
<td>Online</td>
</tr>
<tr>
<td>exportWalletObject</td>
<td>Export an object from an Oracle wallet to a file.</td>
<td>Online</td>
</tr>
<tr>
<td>generateKey</td>
<td>Generate a key pair in a JKS keystore.</td>
<td>Online</td>
</tr>
<tr>
<td>getKeyStoreObject</td>
<td>Display a certificate or other object present in a JKS keystore.</td>
<td>Online</td>
</tr>
<tr>
<td>getSSL</td>
<td>Display the SSL attributes for a component listener.</td>
<td>Online</td>
</tr>
<tr>
<td>getWalletObject</td>
<td>Display a certificate or other object present in an Oracle wallet.</td>
<td>Online</td>
</tr>
<tr>
<td>importKeyStore</td>
<td>Import a JKS keystore from a file.</td>
<td>Online</td>
</tr>
<tr>
<td>importKeyStoreObject</td>
<td>Import a certificate or other object from a file to a JKS keystore.</td>
<td>Online</td>
</tr>
<tr>
<td>importWallet</td>
<td>Import an Oracle wallet from a file.</td>
<td>Online</td>
</tr>
<tr>
<td>importWalletObject</td>
<td>Import a certificate or other object from a file to an Oracle wallet.</td>
<td>Online</td>
</tr>
<tr>
<td>listKeyStoreObjects</td>
<td>List all objects present in a JKS keystore.</td>
<td>Online</td>
</tr>
<tr>
<td>listKeyStores</td>
<td>List all JKS keystores configured for a component instance.</td>
<td>Online</td>
</tr>
</tbody>
</table>
4.3.1 addCertificateRequest

Online command that generates a certificate signing request in an Oracle wallet.

**4.3.1.1 Description**

This command generates a certificate signing request in Base64 encoded PKCS#10 format in an Oracle wallet for a component instance (Oracle HTTP Server, Oracle WebCache or Oracle Internet Directory). To get a certificate signed by a certificate authority (CA), send the certificate signing request to your CA.

**4.3.1.2 Syntax**

```plaintext
addCertificateRequest(instName, compName, compType, walletName, password, DN, keySize)
```

**Argument**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are 'ohs', 'oid', and 'webcache'.</td>
</tr>
<tr>
<td>walletName</td>
<td>Specifies the name of the wallet file.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the wallet.</td>
</tr>
<tr>
<td>DN</td>
<td>Specifies the Distinguished Name of the key pair entry.</td>
</tr>
<tr>
<td>keySize</td>
<td>Specifies the key size in bits.</td>
</tr>
</tbody>
</table>

**4.3.1.3 Example**

The following command generates a certificate signing request with DN cn=www.acme.com and key size 1024 in wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1:

```plaintext
wls:/mydomain/serverConfig> addCertificateRequest('inst1', 'oid1', 'oid', 'wallet1', 'password', 'cn=www.acme.com', '1024')
```

4.3.2 addSelfSignedCertificate

Online command that adds a self-signed certificate.
4.3.2.1 Description
This command creates a key pair and wraps it in a self-signed certificate in an Oracle wallet for the specified component instance (Oracle HTTP Server, Oracle WebCache or Oracle Internet Directory). Only keys based on the RSA algorithm are generated.

4.3.2.2 Syntax
addSelfSignedCertificate(instName, compName, compType, walletName, password, DN, keySize)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are 'ohs', 'oid', and 'webcache'.</td>
</tr>
<tr>
<td>walletName</td>
<td>Specifies the name of the wallet file.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the wallet.</td>
</tr>
<tr>
<td>DN</td>
<td>Specifies the Distinguished Name of the key pair entry.</td>
</tr>
<tr>
<td>keySize</td>
<td>Specifies the key size in bits.</td>
</tr>
</tbody>
</table>

4.3.2.3 Example
The following command adds a self-signed certificate with DN `cn=www.acme.com`, key size 1024 to `wallet1`, for Oracle Internet Directory instance `oid1`, in application server instance `inst1`:

```shell
wls:/mydomain/serverConfig> addSelfSignedCertificate('inst1', 'oid1', 'oid', 'wallet1', 'password', 'cn=www.acme.com', '1024')
```

4.3.3 changeKeyStorePassword
Online command that changes the keystore password.

4.3.3.1 Description
This command changes the password of a Java Keystore (JKS) file for an Oracle Virtual Directory instance.

4.3.3.2 Syntax
changeKeyStorePassword(instName, compName, compType, keystoreName, currPassword, newPassword)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid value is 'ovd'.</td>
</tr>
<tr>
<td>keystoreName</td>
<td>Specifies the filename of the keystore.</td>
</tr>
<tr>
<td>currPassword</td>
<td>Specifies the current keystore password.</td>
</tr>
<tr>
<td>newPassword</td>
<td>Specifies the new keystore password.</td>
</tr>
</tbody>
</table>
4.3.3 Example
The following command changes the password of file keys.jks for Oracle Virtual Directory instance ovd1 in application server instance inst1:

```
ws:/mydomain/serverConfig> changeKeyStorePassword('inst1', 'ovd1', 'ovd', 'keys.jks', 'currpassword', 'newpassword')
```

4.3.4 changeWalletPassword
Online command that changes the password of an Oracle wallet.

4.3.4.1 Description
This command changes the password of an Oracle wallet for the specified component instance (Oracle HTTP Server, Oracle WebCache or Oracle Internet Directory). This command is only applicable to password-protected wallets.

4.3.4.2 Syntax
```
changeWalletPassword(instName, compName, compType, walletName, currPassword, newPassword)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are 'oid', 'ohs', and 'webcache'.</td>
</tr>
<tr>
<td>walletName</td>
<td>Specifies the filename of the wallet.</td>
</tr>
<tr>
<td>currPassword</td>
<td>Specifies the current wallet password.</td>
</tr>
<tr>
<td>newPassword</td>
<td>Specifies the new wallet password.</td>
</tr>
</tbody>
</table>

4.3.4.3 Example
The following command changes the password for wallet1 from currpassword to newpassword for Oracle HTTP Server instance ohs1 in application server instance inst1:

```
ws:/mydomain/serverConfig> changeWalletPassword('inst1', 'ohs1', 'ohs', 'wallet1', 'currpassword', 'newpassword')
```

4.3.5 configureSSL
Online command that sets SSL attributes.

4.3.5.1 Description
This command sets the SSL attributes for a component listener. The attributes are specified in a properties file format (name=value). If a properties file is not provided, or it does not contain any SSL attributes, default attribute values are used. For component-specific SSL attribute value defaults, see the chapter “SSL Configuration in Oracle Fusion Middleware” in the Oracle Fusion Middleware Administrator’s Guide.

4.3.5.2 Syntax
```
configureSSL(instName, compName, compType, listener, filePath)
```
4.3.5.3 Examples
The following command configures SSL attributes specified in the properties file /tmp/ssl.properties for Oracle Virtual Directory instance ovd1 in application server instance inst1, for listener listener1:

```
wls:/mydomain/serverConfig> configureSSL('inst1', 'ovd1', 'ovd', 'listener1', '/tmp/ssl.properties')
```

The following command configures SSL attributes without specifying a properties file. Since no file is provided, the default SSL attribute values are used:

```
wls:/mydomain/serverConfig> configureSSL('inst1', 'ovd1', 'ovd', 'listener2')
```

4.3.6 createKeyStore
Online command that creates a JKS keystore.

4.3.6.1 Description
This command creates a Java keystore (JKS) for the specified Oracle Virtual Directory instance. For keystore file location and other information, see the chapter "Managing Keystores, Wallets, and Certificates" in the Oracle Fusion Middleware Administrator’s Guide.

4.3.6.2 Syntax
createKeyStore(instName, compName, compType, keystoreName, password)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are 'oid', 'ovd', 'ohs', and 'webcache'.</td>
</tr>
<tr>
<td>listener</td>
<td>Specifies the name of the component listener to be configured for SSL.</td>
</tr>
<tr>
<td>filePath</td>
<td>Specifies the absolute path of the properties file containing the SSL attributes to set.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid value is 'ovd'.</td>
</tr>
<tr>
<td>keystoreName</td>
<td>Specifies the filename of the keystore file to be created.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the keystore password.</td>
</tr>
</tbody>
</table>

4.3.6.3 Example
The following command creates JKS file keys.jks with password password for Oracle Virtual Directory instance ovd1 in application server instance inst1:

```
wls:/mydomain/serverConfig> createKeyStore('inst1', 'ovd1', 'ovd', 'keys.jks', 'password')
```
4.3.7 createWallet

Online command that creates an Oracle wallet.

4.3.7.1 Description
This command creates an Oracle wallet for the specified component instance (Oracle HTTP Server, Oracle WebCache or Oracle Internet Directory). Wallets can be of password-protected or auto-login type. For wallet details, see the chapter "Managing Keystores, Wallets, and Certificates" in the Oracle Fusion Middleware Administrator’s Guide.

4.3.7.2 Syntax
createWallet(instName, compName, compType, walletName, password)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are 'oid', 'ohs', and 'webcache'.</td>
</tr>
<tr>
<td>walletName</td>
<td>Specifies the name of the wallet file to be created.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the wallet password.</td>
</tr>
</tbody>
</table>

4.3.7.3 Examples
The following command creates a wallet named wallet1 with password password, for Oracle HTTP Server instance ohs1 in application server instance inst1:

```
wls:/mydomain/serverConfig> createWallet('inst1', 'ohs1', 'ohs', 'wallet1', 'password')
```

The following command creates an auto-login wallet named wallet2 for Oracle WebCache instance wc1, in application server instance inst1:

```
wls:/mydomain/serverConfig> createWallet('inst1', 'wc1', 'webcache', 'wallet2', '')
```

4.3.8 deleteKeyStore

Online command that deletes a keystore.

4.3.8.1 Description
This command deletes a keystore for a specified Oracle Virtual Directory instance.

4.3.8.2 Syntax
deleteKeyStore(instName, compName, compType, keystoreName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid value is 'ovd'.</td>
</tr>
<tr>
<td>keystoreName</td>
<td>Specifies the name of the keystore file to delete.</td>
</tr>
</tbody>
</table>
4.3.8.3 Example
The following command deletes JKS file keys.jks for Oracle Virtual Directory instance ovd1 in application server instance inst1:

```
<wls:/mydomain/serverConfig> deleteKeyStore('inst1', 'ovd1', 'ovd', 'keys.jks')
```

4.3.9 deleteWallet
Online command that deletes an Oracle wallet.

4.3.9.1 Description
This command deletes an Oracle wallet for the specified component instance (Oracle HTTP Server, Oracle WebCache or Oracle Internet Directory).

4.3.9.2 Syntax
```
deleteWallet(instName, compName, compType, walletName)
```

### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are 'oid', 'ohs', and 'webcache'.</td>
</tr>
<tr>
<td>walletName</td>
<td>Specifies the name of the wallet file to be deleted.</td>
</tr>
</tbody>
</table>

4.3.9.3 Example
The following command deletes a wallet named wallet1 for Oracle HTTP Server instance ohs1 in application server instance inst1:

```
<wls:/mydomain/serverConfig> deleteWallet('inst1', 'ohs1', 'ohs', 'wallet1')
```

4.3.10 exportKeyStore
Online command that exports the keystore to a file.

4.3.10.1 Description
This command exports a keystore, configured for the specified Oracle Virtual Directory instance, to a file under the given directory. The exported filename is the same as the keystore name.

4.3.10.2 Syntax
```
exportKeyStore(instName, compName, compType, keystoreName, password, path)
```

### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid value is 'ovd'.</td>
</tr>
<tr>
<td>keystoreName</td>
<td>Specifies the name of the keystore file.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the keystore.</td>
</tr>
</tbody>
</table>
4.3.10.3 Example
The following command exports the keystore keys.jks for Oracle Virtual Directory instance ovd1 to file keys.jks under /tmp:

```
wls:/mydomain/serverConfig> exportKeyStore('inst1', 'ovd1', 'ovd', 'keys.jks', 'password', '/tmp')
```

4.3.11 exportKeyStoreObject

Online command that exports an object from a keystore to a file.

4.3.11.1 Description
This command exports a certificate signing request, certificate/certificate chain, or trusted certificate present in a Java keystore (JKS) to a file for the specified Oracle Virtual Directory instance. The certificate signing request is generated before exporting the object. The alias specifies the object to be exported.

4.3.11.2 Syntax

```
exportKeyStoreObject(instName, compName, compType, keystoreName, password, type, path, alias)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid value is 'ovd'.</td>
</tr>
<tr>
<td>keystoreName</td>
<td>Specifies the name of the keystore file.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the keystore.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of the keystore object to be exported. Valid values are 'CertificateRequest', 'Certificate', 'TrustedCertificate' and 'TrustedChain'.</td>
</tr>
<tr>
<td>path</td>
<td>Specifies the absolute path of the directory under which the object is exported as a file named base64.txt.</td>
</tr>
<tr>
<td>alias</td>
<td>Specifies the alias of the keystore object to be exported.</td>
</tr>
</tbody>
</table>

4.3.11.3 Examples
The following command generates and exports a certificate signing request from the key-pair indicated by alias mykey in keys.jks, for Oracle Virtual Directory instance ovd1 in application server instance inst1. The certificate signing request is exported under the directory /tmp:

```
wls:/mydomain/serverConfig> exportKeyStoreObject('inst1', 'ovd1', 'ovd', 'keys.jks', 'password', 'CertificateRequest', '/tmp', 'mykey')
```

The following command exports a certificate or certificate chain indicated by alias mykey in keys.jks, for Oracle Virtual Directory instance ovd1, in application server instance inst1. The certificate or certificate chain is exported under the directory /tmp:
The following command exports a trusted certificate indicated by alias mykey in keys.jks, for Oracle Virtual Directory instance ovd1, in application server instance inst1. The trusted certificate is exported under the directory /tmp:

```wls:/mydomain/serverConfig> exportKeyStoreObject('inst1', 'ovd1', 'ovd', 'keys.jks', 'password', 'Certificate', '/tmp', 'mykey')```

### 4.3.12 exportWallet

Online command that exports an Oracle wallet.

#### 4.3.12.1 Description

This command exports an Oracle wallet, configured for a specified component instance (Oracle HTTP Server, Oracle WebCache or Oracle Internet Directory), to file(s) under the given directory. If the exported file is an auto-login only wallet, the file name is 'cwallet.sso'. If it is password-protected wallet, two files are created: 'ewallet.p12' and 'cwallet.sso'.

#### 4.3.12.2 Syntax

```
exportWallet(instName, compName, compType, walletName, password, path)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are 'oid', 'ohs', and 'webcache'.</td>
</tr>
<tr>
<td>walletName</td>
<td>Specifies the name of the wallet file.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the wallet.</td>
</tr>
<tr>
<td>path</td>
<td>Specifies the absolute path of the directory under which the object is exported.</td>
</tr>
</tbody>
</table>

#### 4.3.12.3 Examples

The following command exports auto-login wallet wallet1 for Oracle Internet Directory instance oid1 to file cwallet.sso under /tmp:

```wls:/mydomain/serverConfig> exportWallet('inst1', 'oid1', 'oid', 'wallet1', '', '/tmp')```

The following command exports password-protected wallet wallet2 for Oracle Internet Directory instance oid1 to two files, ewallet.p12 and cwallet.sso, under /tmp:

```wls:/mydomain/serverConfig> exportWallet('inst1', 'oid1', 'oid', 'wallet2', 'password', '/tmp')```

### 4.3.13 exportWalletObject

Online command that exports a certificate or other wallet object to a file.
4.3.13.1 Description

This command exports a certificate signing request, certificate, certificate chain or trusted certificate present in an Oracle wallet to a file for the specified component instance (Oracle HTTP Server, Oracle WebCache or Oracle Internet Directory). DN is used to indicate the object to be exported.

4.3.13.2 Syntax

```
exportWalletObject(instName, compName, compType, walletName, password, type, path, DN)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are ‘ohs’, ‘oid’, and ‘webcache’.</td>
</tr>
<tr>
<td>walletName</td>
<td>Specifies the name of the wallet file.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the wallet.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of wallet object to be exported. Valid values are ‘CertificateRequest’, ‘Certificate’, ‘TrustedCertificate’ or ‘TrustedChain’.</td>
</tr>
<tr>
<td>path</td>
<td>Specifies the absolute path of the directory under which the object is exported as a file base64.txt.</td>
</tr>
<tr>
<td>DN</td>
<td>Specifies the Distinguished Name of the wallet object being exported.</td>
</tr>
</tbody>
</table>

4.3.13.3 Examples

The following command exports a certificate signing request with DN cn=www.acme.com in wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1. The certificate signing request is exported under the directory /tmp:

```
wls:/mydomain/serverConfig> exportWalletObject('inst1', 'oid1', 'oid', 'wallet1', 'password', 'CertificateRequest', '/tmp', 'cn=www.acme.com')
```

The following command exports a certificate with DN cn=www.acme.com in wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1. The certificate or certificate chain is exported under the directory /tmp:

```
wls:/mydomain/serverConfig> exportWalletObject('inst1', 'oid1', 'oid', 'wallet1', 'password', 'Certificate', '/tmp', 'cn=www.acme.com')
```

The following command exports a trusted certificate with DN cn=www.acme.com in wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1. The trusted certificate is exported under the directory /tmp:

```
wls:/mydomain/serverConfig> exportWalletObject('inst1', 'oid1', 'oid', 'wallet1', 'password', 'TrustedCertificate', '/tmp', 'cn=www.acme.com')
```

The following command exports a certificate chain with DN cn=www.acme.com in wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1. The certificate or certificate chain is exported under the directory /tmp:

```
wls:/mydomain/serverConfig> exportWalletObject('inst1', 'oid1', 'oid', 'wallet1', 'password', 'TrustedChain', '/tmp', 'cn=www.acme.com')
```
4.3.14 generateKey

Online command that generates a key pair in a Java keystore.

4.3.14.1 Description
This command generates a key pair in a Java keystore (JKS) for Oracle Virtual Directory. It also wraps the key pair in a self-signed certificate. Only keys based on the RSA algorithm are generated.

4.3.14.2 Syntax
`generateKey(instName, compName, compType, keystoreName, password, DN, keySize, alias, algorithm)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid value is 'ovd'.</td>
</tr>
<tr>
<td>keystoreName</td>
<td>Specifies the name of the keystore.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the keystore.</td>
</tr>
<tr>
<td>DN</td>
<td>Specifies the Distinguished Name of the key pair entry.</td>
</tr>
<tr>
<td>keySize</td>
<td>Specifies the key size in bits.</td>
</tr>
<tr>
<td>alias</td>
<td>Specifies the alias of the key pair entry in the keystore.</td>
</tr>
<tr>
<td>algorithm</td>
<td>Specifies the key algorithm. Valid value is 'RSA'.</td>
</tr>
</tbody>
</table>

4.3.14.3 Examples
The following command generates a key pair with DN `cn=www.acme.com`, key size 1024, algorithm RSA and alias `mykey` in `keys.jks`, for Oracle Virtual Directory instance `ovd1` in application server instance `inst1`:

```
  wls:/mydomain/serverConfig> generateKey('inst1', 'ovd1', 'ovd','keys.jks',
                                                  'password', 'cn=www.acme.com', '1024', 'mykey', 'RSA')
```

The following command is the same as above, except it does not explicitly specify the key algorithm:

```
  wls:/mydomain/serverConfig> generateKey('inst1', 'ovd1', 'ovd','keys.jks',
                                                  'password', 'cn=www.acme.com', '1024', 'mykey')
```

4.3.15 getKeyStoreObject

Online command that shows details about a keystore object.

4.3.15.1 Description
This command displays a specific certificate or trusted certificate present in a Java keystore (JKS) for Oracle Virtual Directory. The keystore object is indicated by its index number, as given by the `listKeyStoreObjects` command. It shows the certificate details including DN, key size, algorithm, and other information.

4.3.15.2 Syntax
`getKeyStoreObject(instName, compName, compType, keystoreName, password, type,`
4.3.15.3 Examples
The following command shows a trusted certificate with index 1 present in keys.jks, for Oracle Virtual Directory instance ovd1, in application server instance inst1:

```
wls:/mydomain/serverConfig> getKeyStoreObject('inst1', 'ovd1', 'ovd','keys.jks', 'password', 'TrustedCertificate', '1')
```

The following command shows a certificate with index 1 present in keys.jks, for Oracle Virtual Directory instance ovd1, in application server instance inst1:

```
wls:/mydomain/serverConfig> getKeyStoreObject('inst1', 'ovd1', 'ovd','keys.jks', 'password', 'Certificate', '1')
```

4.3.16 getSSL
Online command that lists the configured SSL attributes.

4.3.16.1 Description
This command lists the configured SSL attributes for the specified component listener. For Oracle Internet Directory, the listener name is always sslport1.

4.3.16.2 Syntax
getSSL(instName, compName, compType, listener)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are 'ovd', 'oid', 'ohs', and 'webcache'.</td>
</tr>
<tr>
<td>keystoreName</td>
<td>Specifies the name of the keystore file.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the keystore.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of the keystore object to be listed. Valid values are 'Certificate' and 'TrustedCertificate'.</td>
</tr>
<tr>
<td>index</td>
<td>Specifies the index number of the keystore object as returned by the listKeyStoreObjects command.</td>
</tr>
</tbody>
</table>

4.3.16.3 Example
The following command shows the SSL attributes configured for Oracle Internet Directory instance oid1, in application server instance inst1, for listener sslport1:

```
wls:/mydomain/serverConfig> getSSL('inst1', 'oid1', 'oid', 'sslport1')
```
4.3.17 getWalletObject

Online command that displays information about a certificate or other object in an Oracle wallet.

4.3.17.1 Description

This command displays a specific certificate signing request, certificate or trusted certificate present in an Oracle wallet for the specified component instance (Oracle HTTP Server, Oracle WebCache or Oracle Internet Directory). The wallet object is indicated by its index number, as given by the listWalletObjects command. For certificates or trusted certificates, it shows the certificate details including DN, key size, algorithm and other data. For certificate signing requests, it shows the subject DN, key size and algorithm.

4.3.17.2 Syntax

getWalletObject(instName, compName, compType, walletName, password, type, index)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are 'ohs', 'oid', and 'webcache'.</td>
</tr>
<tr>
<td>walletName</td>
<td>Specifies the name of the wallet file.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the wallet.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of wallet object to be exported. Valid values are 'CertificateRequest', 'Certificate', and 'TrustedCertificate'.</td>
</tr>
<tr>
<td>index</td>
<td>Specifies the index number of the wallet object as returned by the listWalletObjects command.</td>
</tr>
</tbody>
</table>

4.3.17.3 Examples

The following command shows certificate signing request details for the object with index 0 present in wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1:

wls:/mydomain/serverConfig> getKeyStoreObject('inst1', 'oid1', 'oid', 'wallet1', 'password', 'CertificateRequest', '0')

The following command shows certificate details for the object with index 0 present in wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1:

wls:/mydomain/serverConfig> getKeyStoreObject('inst1', 'oid1', 'oid', 'wallet1', 'password', 'Certificate', '0')

The following command shows trusted certificate details for the object with index 0, present in wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1:

wls:/mydomain/serverConfig> getKeyStoreObject('inst1', 'oid1', 'oid', 'wallet1', 'password', 'TrustedCertificate', '0')
4.3.18 importKeyStore

Online command that imports a keystore from a file.

4.3.18.1 Description

This command imports a Java keystore (JKS) from a file to the specified Oracle Virtual Directory instance for manageability. The component instance name must be unique.

4.3.18.2 Syntax

importKeyStore(instName, compName, compType, keystoreName, password, filePath)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid value is 'ovd'.</td>
</tr>
<tr>
<td>keystoreName</td>
<td>Specifies the name of the keystore being imported. This name must be unique for this component instance.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the keystore.</td>
</tr>
<tr>
<td>filePath</td>
<td>Specifies the absolute path of the keystore file to be imported.</td>
</tr>
</tbody>
</table>

4.3.18.3 Example

The following command imports the keystore /tmp/keys.jks as file.jks into Oracle Virtual Directory instance ovd1. Subsequently, the keystore is managed through the name file.jks:

```
<Find admin>
<identity user='admin' password='admin'>
<server name='mydomain'>
<server configuration>
<server configuration> wls:/mydomain/serverConfig>
importKeyStore('inst1', 'ovd1', 'ovd', 'file.jks', 'password', '/tmp/keys.jks')
```

4.3.19 importKeyStoreObject

Online command that imports an object from a file to a keystore.

4.3.19.1 Description

This command imports a certificate, certificate chain, or trusted certificate into a Java keystore (JKS) for Oracle Virtual Directory, assigning it the specified alias which must be unique in the keystore. If a certificate or certificate chain is being imported, the alias must match that of the corresponding key-pair.

4.3.19.2 Syntax

importKeyStoreObject(instName, compName, compType, keystoreName, password, type, filePath, alias)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid value is 'ovd'.</td>
</tr>
<tr>
<td>keystoreName</td>
<td>Specifies the name of the keystore.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the keystore.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of the object being imported.</td>
</tr>
<tr>
<td>filePath</td>
<td>Specifies the absolute path of the file to import.</td>
</tr>
<tr>
<td>alias</td>
<td>Specifies the alias of the object.</td>
</tr>
</tbody>
</table>
4.3.19.3 Examples

The following command imports a certificate or certificate chain from file cert.txt into keys.jks, using alias mykey for Oracle Virtual Directory instance ovd1, in application server instance inst1. The file keys.jks must already have an alias mykey for a key-pair whose public key matches that in the certificate being imported:

```
<wls:/mydomain/serverConfig> > importKeyStoreObject('inst1', 'ovd1',
    'ovd', 'keys.jks', 'password', 'Certificate', '/tmp/cert.txt', 'mykey')
```

The following command imports a trusted certificate from file trust.txt into keys.jks using alias mykey1, for Oracle Virtual Directory instance ovd1 in application server instance inst1:

```
<wls:/mydomain/serverConfig> importKeyStoreObject('inst1', 'ovd1',
    'ovd', 'keys.jks', 'password', 'TrustedCertificate', '/tmp/trust.txt', 'mykey1')
```

### 4.3.20 importWallet

Online command that imports an Oracle wallet from a file.

#### 4.3.20.1 Description

This command imports an Oracle wallet from a file to the specified component instance (Oracle HTTP Server, Oracle WebCache, or Oracle Internet Directory) for manageability. If the wallet being imported is an auto-login wallet, the file path must point to cwallet.sso; if the wallet is password-protected, it must point to ewallet.p12. The wallet name must be unique for the component instance.

#### 4.3.20.2 Syntax

```
importWallet(instName, compName, compType, walletName, password, filePath)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are 'ohs', 'oid', and 'webcache'.</td>
</tr>
<tr>
<td>walletName</td>
<td>Specifies the name of the wallet being imported. The name must be unique for the component instance.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the wallet.</td>
</tr>
<tr>
<td>filePath</td>
<td>Specifies the absolute path of the wallet file being imported.</td>
</tr>
</tbody>
</table>

#### 4.3.20.3 Examples

The following command imports auto-login wallet file /tmp/cwallet.sso as wallet1 into Oracle Internet Directory instance oid1. Subsequently, the wallet is managed with the name wallet1. No password is passed since it is an auto-login wallet:

```
```
SSL Configuration Commands

The following command imports password-protected wallet /tmp/ewallet.p12 as wallet2 into Oracle Internet Directory instance oid1. Subsequently, the wallet is managed with the name wallet2. The wallet password is passed as a parameter:

```bash
wls:/mydomain/serverConfig> importWallet('inst1', 'oid1', 'oid', 'wallet2', 'password', '/tmp/ewallet.p12')
```

### 4.3.21 importWalletObject

Online command that imports a certificate or other object into an Oracle wallet.

#### 4.3.21.1 Description

This command imports a certificate, trusted certificate or certificate chain into an Oracle wallet for the specified component instance (Oracle HTTP Server, Oracle WebCache component or Oracle Internet Directory). When importing a certificate, use the same wallet file from which the certificate signing request was generated.

#### 4.3.21.2 Syntax

```bash
importWalletObject(instName, compName, compType, walletName, password, type, filePath)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are 'ohs', 'oid', and 'webcache'.</td>
</tr>
<tr>
<td>walletName</td>
<td>Specifies the name of the wallet file.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the wallet.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of wallet object to be imported. Valid values are 'Certificate', 'TrustedCertificate' and 'TrustedChain'.</td>
</tr>
<tr>
<td>filePath</td>
<td>Specifies the absolute path of the file containing the wallet object.</td>
</tr>
</tbody>
</table>

#### 4.3.21.3 Examples

The following command imports a certificate chain in PKCS#7 format from file chain.txt into wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1:

```bash
wls:/mydomain/serverConfig> importWalletObject('inst1', 'oid1', 'oid', 'wallet1', 'password', 'TrustedChain', '/tmp/chain.txt')
```

The following command imports a certificate from file cert.txt into wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1:

```bash
wls:/mydomain/serverConfig> importWalletObject('inst1', 'oid1', 'oid', 'wallet1', 'password', 'Certificate', '/tmp/cert.txt')
```

The following command imports a trusted certificate from file trust.txt into wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1:

```bash
wls:/mydomain/serverConfig> importWalletObject('inst1', 'oid1', 'oid', 'wallet1', 'password', 'TrustedCertificate', '/tmp/trust.txt')
```
4.3.22 listKeyStoreObjects

Online command that lists the contents of a keystore.

4.3.22.1 Description
This command lists all the certificates or trusted certificates present in a Java keystore (JKS) for Oracle Virtual Directory.

4.3.22.2 Syntax
listKeyStoreObjects(instName, compName, compType, keystoreName, password, type)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid value is 'ovd'.</td>
</tr>
<tr>
<td>keystoreName</td>
<td>Specifies the name of the keystore file.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the keystore.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of keystore object to be listed. Valid values are 'Certificate' and 'TrustedCertificate'.</td>
</tr>
</tbody>
</table>

4.3.22.3 Examples
The following command lists all trusted certificates present in keys.jks, for Oracle Virtual Directory instance ovdl, in application server instance inst1:

```wls:/mydomain/serverConfig> listKeyStoreObjects('inst1', 'ovdl', 'ovd','keys.jks', 'password', 'TrustedCertificate')```

The following command lists all certificates present in keys.jks, for Oracle Virtual Directory instance ovdl, in application server instance inst1:

```wls:/mydomain/serverConfig> listKeyStoreObjects('inst1', 'ovdl', 'ovd','keys.jks', 'password', 'Certificate')```

4.3.23 listKeyStores

Online command that lists all the keystores for a component.

4.3.23.1 Description
This command lists all the Java keystores (JKS) configured for the specified Oracle Virtual Directory instance.

4.3.23.2 Syntax
listKeyStores(instName, compName, compType)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid value is 'ovd'.</td>
</tr>
</tbody>
</table>
4.3.23.3 Example
The following command lists all keystores for Oracle Virtual Directory instance ovd1 in application server instance inst1:

\[ \text{wls:/mydomain/serverConfig> listKeyStores('inst1', 'ovd1', 'ovd')} \]

4.3.24 listWalletObjects
Online command that lists all objects in an Oracle wallet.

4.3.24.1 Description
This command lists all certificate signing requests, certificates, or trusted certificates present in an Oracle wallet for the specified component instance (Oracle HTTP Server, Oracle WebCache or Oracle Internet Directory).

4.3.24.2 Syntax
listWalletObjects(instName, compName, compType, walletName, password, type)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are 'ohs', 'oid', and 'webcache'.</td>
</tr>
<tr>
<td>walletName</td>
<td>Specifies the name of the wallet file.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the wallet.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of wallet object to be listed. Valid values are 'CertificateRequest', 'Certificate', and 'TrustedCertificate'.</td>
</tr>
</tbody>
</table>

4.3.24.3 Examples
The following command lists all certificate signing requests in wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1:

\[ \text{wls:/mydomain/serverConfig> > listWalletObjects('inst1', 'oid1', 'oid', 'wallet1', 'password', 'CertificateRequest')} \]

The following command lists all certificates in wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1:

\[ \text{wls:/mydomain/serverConfig> listWalletObjects('inst1', 'oid1', 'oid', 'wallet1', 'password', 'Certificate')} \]

The following command lists all trusted certificates in wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1:

\[ \text{wls:/mydomain/serverConfig> listWalletObjects('inst1', 'oid1', 'oid', 'wallet1', 'password', 'TrustedCertificate')} \]

4.3.25 listWallets
Online command that lists all wallets configured for a component instance.
4.3.25.1 Description
This command displays all the wallets configured for the specified component instance (Oracle HTTP Server, Oracle WebCache or Oracle Internet Directory), and identifies the auto-login wallets.

4.3.25.2 Syntax
listWallets(instName, compName, compType)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are 'ohs', 'oid', and 'webcache'.</td>
</tr>
</tbody>
</table>

4.3.25.3 Example
The following command lists all wallets for Oracle Internet Directory instance oid1 in application server instance inst1:

wls:/mydomain/serverConfig> listWallets('inst1', 'oid1', 'oid')

4.3.26 removeKeyStoreObject
Online command that removes an object from a keystore.

4.3.26.1 Description
This command removes a certificate request, certificate, trusted certificate, or all trusted certificates from a Java keystore (JKS) for Oracle Virtual Directory. Use an alias to remove a specific object; no alias is needed if all trusted certificates are being removed.

4.3.26.2 Syntax
removeKeyStoreObject(instName, compName, compType, keystoreName, password, type, alias)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid value is 'ovd'.</td>
</tr>
<tr>
<td>keystoreName</td>
<td>Specifies the name of the keystore file.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the keystore.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of the keystore object to be removed. Valid values are 'Certificate', 'TrustedCertificate' or 'TrustedAll'.</td>
</tr>
<tr>
<td>alias</td>
<td>Specifies the alias of the keystore object to be removed.</td>
</tr>
</tbody>
</table>

4.3.26.3 Examples
The following command removes a certificate or certificate chain denoted by alias mykey in keys.jks, for Oracle Virtual Directory instance ovd1, in application server instance inst1:
**4.3.27 removeWalletObject**

Online command that removes a certificate or other object from an Oracle wallet.

**4.3.27.1 Description**

This command removes a certificate signing request, certificate, trusted certificate or all trusted certificates from an Oracle wallet for the specified component instance (Oracle HTTP Server, Oracle WebCache or Oracle Internet Directory). DN is used to indicate the object to be removed.

**4.3.27.2 Syntax**

```
removeWalletObject(instName, compName, compType, walletName, password, type, DN)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>instName</td>
<td>Specifies the name of the application server instance.</td>
</tr>
<tr>
<td>compName</td>
<td>Specifies the name of the component instance.</td>
</tr>
<tr>
<td>compType</td>
<td>Specifies the type of component. Valid values are 'ohs', 'oid', and 'webcache'.</td>
</tr>
<tr>
<td>walletName</td>
<td>Specifies the name of the wallet file.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the wallet.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of the keystore object to be removed. Valid values are 'CertificateRequest', 'Certificate', 'TrustedCertificate' or 'TrustedAll'.</td>
</tr>
<tr>
<td>DN</td>
<td>Specifies the Distinguished Name of the wallet object to be removed.</td>
</tr>
</tbody>
</table>

**4.3.27.3 Examples**

The following command removes all trusted certificates from wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1. It is not necessary to provide a DN, so we pass null (denoted by None) for the DN parameter:

```
wls:/mydomain/serverConfig> removeWalletObject('inst1', 'oid1', 'oid','wallet1', 'password', 'TrustedAll',None)
```

The following command removes a certificate signing request indicated by DN cn=www.acme.com from wallet1, for Oracle Internet Directory instance oid1, in application server instance inst1:
The following command removes a certificate indicated by DN `cn=www.acme.com` from wallet1, for Oracle Internet Directory instance `oid1`, in application server instance `inst1`:

```
removeWalletObject('inst1', 'oid1', 'oid','wallet1', 'password', 'CertificateRequest','cn=www.acme.com')
```

The following command removes a trusted certificate indicated by DN `cn=www.acme.com` from wallet1, for Oracle Internet Directory instance `oid1`, in application server instance `inst1`:

```
removeWalletObject('inst1', 'oid1', 'oid','wallet1', 'password', 'TrustedCertificate','cn=www.acme.com')
```

### 4.4 Oracle Identity Federation Commands

Use the WLST commands listed in Table 4–4 to view and manage configuration for Oracle Identity Federation.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>addConfigListEntryInMap</td>
<td>Add a configuration list entry to a map.</td>
<td>Online</td>
</tr>
<tr>
<td>addConfigMapEntryInMap</td>
<td>Add a configuration map entry to a map.</td>
<td>Online</td>
</tr>
<tr>
<td>addConfigPropertyListEntry</td>
<td>Add a configuration property list entry.</td>
<td>Online</td>
</tr>
<tr>
<td>addConfigPropertyMapEntry</td>
<td>Add a configuration property map entry to</td>
<td>Online</td>
</tr>
<tr>
<td></td>
<td>the map.</td>
<td></td>
</tr>
<tr>
<td>addCustomAuthnEngine</td>
<td>Add a custom authentication engine.</td>
<td>Online</td>
</tr>
<tr>
<td>addCustomSPEngine</td>
<td>Add a custom SP engine.</td>
<td>Online</td>
</tr>
<tr>
<td>addFederationListEntryInMap</td>
<td>Add a federations list entry to the map.</td>
<td>Online</td>
</tr>
<tr>
<td>addFederationMapEntryInMap</td>
<td>Add a federation map entry to the map.</td>
<td>Online</td>
</tr>
<tr>
<td>addFederationPropertyListEntry</td>
<td>Add a federation property list entry.</td>
<td>Online</td>
</tr>
<tr>
<td>addFederationPropertyMapEntry</td>
<td>Add a federation property map entry.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteCustomAuthnEngine</td>
<td>Delete a custom authentication engine.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteCustomSPEngine</td>
<td>Delete a custom SP engine.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteProviderFederation</td>
<td>Delete a provider from the federation.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteUserFederation</td>
<td>Delete a user from the federation.</td>
<td>Online</td>
</tr>
<tr>
<td>changeMessageStore</td>
<td>Change the message store to memory or RDBMS.</td>
<td>Online</td>
</tr>
<tr>
<td>changePeerProviderDescription</td>
<td>Change a peer provider's description.</td>
<td>Online</td>
</tr>
<tr>
<td>changeSessionStore</td>
<td>Change the session store to memory or RDBMS.</td>
<td>Online</td>
</tr>
<tr>
<td>createConfigPropertyList</td>
<td>Create a configuration property list.</td>
<td>Online</td>
</tr>
<tr>
<td>createConfigPropertyListInMap</td>
<td>Create a configuration property list in the</td>
<td>Online</td>
</tr>
<tr>
<td></td>
<td>map.</td>
<td></td>
</tr>
<tr>
<td>Use this command...</td>
<td>To...</td>
<td>Use with WLST...</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>--------------------------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>createConfigPropertyMap</td>
<td>Create a configuration property map.</td>
<td>Online</td>
</tr>
<tr>
<td>createConfigPropertyMapInMap</td>
<td>Create a nested configuration property map in a map.</td>
<td>Online</td>
</tr>
<tr>
<td>createFederationPropertyList</td>
<td>Create a federation property list.</td>
<td>Online</td>
</tr>
<tr>
<td>createFederationPropertyListInMap</td>
<td>Create a federation property list in the map.</td>
<td>Online</td>
</tr>
<tr>
<td>createFederationPropertyMap</td>
<td>Create a federation property map.</td>
<td>Online</td>
</tr>
<tr>
<td>createFederationPropertyMapInMap</td>
<td>Create a nested federation property map in a map.</td>
<td>Online</td>
</tr>
<tr>
<td>createPeerProviderEntry</td>
<td>Create a peer provider entry.</td>
<td>Online</td>
</tr>
<tr>
<td>getConfigListValueInMap</td>
<td>Retrieve a configuration list value from the map.</td>
<td>Online</td>
</tr>
<tr>
<td>getConfigMapEntryInMap</td>
<td>Retrieve a configuration map value from the map.</td>
<td>Online</td>
</tr>
<tr>
<td>getConfigProperty</td>
<td>Retrieve a configuration property entry.</td>
<td>Online</td>
</tr>
<tr>
<td>getConfigPropertyList</td>
<td>Retrieve a configuration property list.</td>
<td>Online</td>
</tr>
<tr>
<td>getConfigPropertyMapEntry</td>
<td>Retrieve a configuration property map entry.</td>
<td>Online</td>
</tr>
<tr>
<td>getFederationListValueInMap</td>
<td>Retrieve a federation list value from the map.</td>
<td>Online</td>
</tr>
<tr>
<td>getFederationMapEntryInMap</td>
<td>Retrieve a federation map entry from a nested map.</td>
<td>Online</td>
</tr>
<tr>
<td>getFederationProperty</td>
<td>Retrieve a federation property.</td>
<td>Online</td>
</tr>
<tr>
<td>getFederationPropertyList</td>
<td>Retrieve the federation property list.</td>
<td>Online</td>
</tr>
<tr>
<td>extractproviderprops</td>
<td>Export all provider configuration properties to a text file.</td>
<td>Script</td>
</tr>
<tr>
<td>setproviderprops</td>
<td>Set a provider’s properties based on an input text file.</td>
<td>Script</td>
</tr>
<tr>
<td>getFederationPropertyMapEntry</td>
<td>Retrieve a federation property map entry.</td>
<td>Online</td>
</tr>
<tr>
<td>listCustomAuthnEngines</td>
<td>Display the list of custom authentication engines.</td>
<td>Online</td>
</tr>
<tr>
<td>listCustomSPEngines</td>
<td>Display the list of custom SP engines.</td>
<td>Online</td>
</tr>
<tr>
<td>loadMetadata</td>
<td>Load metadata from a file.</td>
<td>Online</td>
</tr>
<tr>
<td>oifStatus</td>
<td>Display the current status of Oracle Identity Federation on the managed server.</td>
<td>Online</td>
</tr>
<tr>
<td>removeConfigListInMap</td>
<td>Delete a configuration list in the map.</td>
<td>Online</td>
</tr>
<tr>
<td>removeConfigMapEntryInMap</td>
<td>Delete a configuration map entry in the map.</td>
<td>Online</td>
</tr>
<tr>
<td>removeConfigMapInMap</td>
<td>Delete a nested configuration map.</td>
<td>Online</td>
</tr>
<tr>
<td>removeConfigProperty</td>
<td>Delete a configuration property.</td>
<td>Online</td>
</tr>
<tr>
<td>removeConfigPropertyList</td>
<td>Delete a property list.</td>
<td>Online</td>
</tr>
<tr>
<td>removeConfigPropertyMap</td>
<td>Delete a property map.</td>
<td>Online</td>
</tr>
</tbody>
</table>
Oracle Identity Federation Commands

4.4.1 addConfigListEntryInMap

Online command that adds a property value to a map.

4.4.1.1 Description

This command adds a property value to a nested list inside a map in config.xml.

4.4.1.2 Syntax

addConfigListEntryInMap(configName, mapname, listName, value, type)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be updated.</td>
</tr>
<tr>
<td>mapname</td>
<td>Specifies the name of the property to map to be changed in config.xml.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the name of the list.</td>
</tr>
<tr>
<td>value</td>
<td>Specifies the property value.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of property, BOOLEAN or STRING or LONG.</td>
</tr>
</tbody>
</table>

4.4.1.3 Example

The following command adds valueA to a map list in server configuration:

wls:/mydomain/serverConfig>
addConfigListEntryInMap('serverconfig','mymap','mylistA','valueA','string')
4.4.2 addConfigMapEntryInMap
Online command that adds a nested map property entry in a map.

4.4.2.1 Description
This command that adds a property name/value pair to a map nested inside a map in config.xml.

4.4.2.2 Syntax
addConfigMapEntryInMap(configName, mapname, nestedMapName, propName, value, type)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be updated.</td>
</tr>
<tr>
<td>mapname</td>
<td>Specifies the name of the property map to be changed in config.xml.</td>
</tr>
<tr>
<td>nestedMapName</td>
<td>name of the nested property map to be changed.</td>
</tr>
<tr>
<td>propName</td>
<td>Specifies the name of the list.</td>
</tr>
<tr>
<td>value</td>
<td>Specifies the property value.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of property, BOOLEAN or STRING or LONG.</td>
</tr>
</tbody>
</table>

4.4.2.3 Example
The following command adds a boolean name/value pair to nestedmapB inside the map mymap.

```
wls:/mydomain/serverConfig>
addConfigMapEntryInMap('serverconfig','mymap','nestedmapB','myvarB','true','boolean')
```

4.4.3 addConfigPropertyListEntry
Online command that adds a list property entry to config.xml.

4.4.3.1 Description
This command adds a property value to a list in config.xml.

4.4.3.2 Syntax
addConfigPropertyListEntry(configName, listName, value, type)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be updated.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the name of the property list to be added in config.xml.</td>
</tr>
<tr>
<td>value</td>
<td>Specifies the new property list value. The entered value is appended to the list.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of property, BOOLEAN or STRING or LONG.</td>
</tr>
</tbody>
</table>

4.4.3.3 Example
The following command adds a string value to mylistA.

```
addConfigPropertyListEntry(configName, 'mylistA', 'myVarA', 'myValueA', 'string')
```
4.4.4 addConfigPropertyMapEntry

Online command that adds a property name/value entry in a map in config.xml.

4.4.4.1 Description
This command adds a property name/value entry in a map in config.xml.

4.4.4.2 Syntax
addConfigPropertyMapEntry(configName, mapName, propName, value, type)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be updated.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map in config.xml.</td>
</tr>
<tr>
<td>propName</td>
<td>Specifies the name of the property map.</td>
</tr>
<tr>
<td>value</td>
<td>Specifies the property map value to be added.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of property, BOOLEAN or STRING or LONG.</td>
</tr>
</tbody>
</table>

4.4.4.3 Example
The following command adds valueA of string type to a map.

wls:/mydomain/serverConfig>
addConfigPropertyMapEntry('serverconfig','mymapA','myvarA','valueA','string')

4.4.5 addCustomAuthnEngine

Online command that adds a custom authentication integration engine.

4.4.5.1 Description
This command adds a custom authentication integration engine to config.xml.

4.4.5.2 Syntax
addCustomAuthnEngine(name, [enabled], [webContext], [authnRelativePath], [logoutRelativePath], [logoutEnabled])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Specifies the name of the custom engine.</td>
</tr>
<tr>
<td>enabled</td>
<td>This flag specifies whether the engine is enabled (true) or not (false, default).</td>
</tr>
<tr>
<td>webContext</td>
<td>Specifies the web context for the engine.</td>
</tr>
<tr>
<td>authnRelativePath</td>
<td>Specifies the authentication relative path URL for the engine.</td>
</tr>
<tr>
<td>logoutRelativePath</td>
<td>Specifies the logout relative path URL for the engine.</td>
</tr>
<tr>
<td>logoutEnabled</td>
<td>This flag is set true to enable logout for the engine, else false.</td>
</tr>
</tbody>
</table>
4.4.5.3 Example
The following command defines an engine named test and enables it.

```
wls:/mydomain/serverConfig> addCustomAuthnEngine('test','true')
```

4.4.6 addCustomSPEngine
Online command that adds a custom service provider (SP) engine.

4.4.6.1 Description
This command adds a custom SP integration engine to config.xml.

4.4.6.2 Syntax
```
addCustomSPEngine(name, [enabled, [authnMech], [webContext], [authnRelativePath],
[logoutRelativePath], [logoutEnabled])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Specifies the name of the custom engine.</td>
</tr>
<tr>
<td>enabled</td>
<td>This flag specifies whether the engine is enabled (true) or not (false).</td>
</tr>
<tr>
<td>authnMech</td>
<td>Specifies the authentication mechanism for the engine.</td>
</tr>
<tr>
<td>webContext</td>
<td>Specifies the web context for the engine.</td>
</tr>
<tr>
<td>authnRelativePath</td>
<td>Specifies the authentication relative path URL for the engine.</td>
</tr>
<tr>
<td>logoutRelativePath</td>
<td>Specifies the logout relative path URL for the engine.</td>
</tr>
<tr>
<td>logoutEnabled</td>
<td>This flag is set true to enable logout for the engine, else false.</td>
</tr>
</tbody>
</table>

4.4.6.3 Example
The following command adds an engine and gives it a disabled status.

```
addCustomSPEngine('new engine','false','oracle:fed:authentication:unspecified','webcontext')
```

4.4.7 addFederationListEntryInMap
Online command that adds a list property entry in a map.

4.4.7.1 Description
This command adds a property value to a nested list inside a map in cot.xml.

4.4.7.2 Syntax
```
addFederationListEntryInMap(providerID, mapname, listName, value, type)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the provider ID.</td>
</tr>
<tr>
<td>mapname</td>
<td>Specifies the name of the property map to be changed in cot.xml.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the name of the property list to be added to the map.</td>
</tr>
<tr>
<td>value</td>
<td>Specifies the property list value to be added. The entered value is appended to the list.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of property, BOOLEAN or STRING or LONG.</td>
</tr>
</tbody>
</table>
4.4.7.3 Example
The following command adds a boolean property list to mymap.

```
>addFederationListEntryInMap('providerB','mymap','mylistB','true','boolean')
```

4.4.8 addFederationMapEntryInMap

Online command that adds a nested map property entry in a map.

4.4.8.1 Description
This command adds a property name/value pair to a map nested inside a map in cot.xml.

4.4.8.2 Syntax

```
addFederationMapEntryInMap(providerID, mapname, nestedMapName, propName, value, type)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the provider ID.</td>
</tr>
<tr>
<td>mapname</td>
<td>Specifies the name of the property map to be changed in cot.xml.</td>
</tr>
<tr>
<td>nestedMapName</td>
<td>Specifies the name of the nested property map to be changed.</td>
</tr>
<tr>
<td>propName</td>
<td>Specifies the name of the property to be updated in the map.</td>
</tr>
<tr>
<td>value</td>
<td>Specifies the property value to be added. The entered value is appended to the list.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of property, BOOLEAN or STRING or LONG.</td>
</tr>
</tbody>
</table>

4.4.8.3 Example
The following command adds a value of type string to the myvarA property in a nested map.

```
>addFederationMapEntryInMap('providerA','mymap','nestedmapA','myvarA','valueA', 'string')
```

4.4.9 addFederationPropertyListEntry

Online command that adds a list property entry.

4.4.9.1 Description
This command adds a property value to a list in cot.xml.

4.4.9.2 Syntax

```
addFederationPropertyListEntry(providerID, listName, value, type)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the provider ID.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the name of the property list to be updated.</td>
</tr>
</tbody>
</table>
### 4.4.9.3 Example
The following command adds a value in string format to a specified property list.

```
$ wls:/mydomain/serverConfig>
addFederationPropertyListEntry('providerA','mylistA','valueA','string')
```

### 4.4.10 addFederationPropertyMapEntry
Online command that a property name/value entry in a map.

#### 4.4.10.1 Description
This command adds a property name/value pair to a map in cot.xml.

#### 4.4.10.2 Syntax
```
addFederationPropertyMapEntry(providerID, mapName, propName, value, type)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the provider ID.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map to be changed in cot.xml.</td>
</tr>
<tr>
<td>propName</td>
<td>Specifies the name of the property to be added in the map.</td>
</tr>
<tr>
<td>value</td>
<td>Specifies the property value to be added. The entered value is appended to the list.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of property, BOOLEAN or STRING or LONG.</td>
</tr>
</tbody>
</table>

#### 4.4.10.3 Example
The following command adds boolean property `myvarB` to a map.

```
$ wls:/mydomain/serverConfig>
addFederationPropertyMapEntry('providerA', 'mymapB', 'myvarB', 'true', 'boolean')
```

### 4.4.11 deleteCustomAuthnEngine
Online command that deletes a custom authentication integration engine from the configuration.

#### 4.4.11.1 Description
This command deletes a custom authentication integration engine in config.xml. You must provide the engine ID for an existing custom authentication engine in config.xml.

#### 4.4.11.2 Syntax
```
deleteCustomAuthnEngine(engineID)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>engineID</td>
<td>Specifies the engine ID of an existing engine to be deleted.</td>
</tr>
</tbody>
</table>
4.4.11.3 Example
The following command deletes the authentication engine with ID id1234.

```
wls:/mydomain/serverConfig> deleteCustomAuthnEngine('id1234')
```

4.4.12 deleteCustomSPEngine
Online command that deletes a custom service provider (SP) integration engine from the configuration.

4.4.12.1 Description
This command deletes a custom SP integration engine in config.xml. The EngineID for an existing custom SP engine in config.xml must be provided.

4.4.12.2 Syntax
```
ddeleteCustomSPEngine(engineID)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>engineID</td>
<td>Specifies the engine ID of an existing engine to be deleted.</td>
</tr>
</tbody>
</table>

4.4.12.3 Example
The following command deletes the engine with ID id1234.

```
wls:/mydomain/serverConfig> deleteCustomSPEngine('id1234')
```

4.4.13 deleteProviderFederation
Online command that deletes federations for given provider.

4.4.13.1 Description
This command deletes federations for given provider ID.

4.4.13.2 Syntax
```
ddeleteProviderFederation(providerID)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the ProviderID for the peer provider for which federation is to be deleted.</td>
</tr>
</tbody>
</table>

4.4.13.3 Example
The following command deletes providerA:

```
wls:/mydomain/serverConfig> deleteProviderFederation(providerA)
```

4.4.14 deleteUserFederation
Online command that deletes federations for given users.

4.4.14.1 Description
This command deletes federations for the given list of users.
4.4.14.2 Syntax

deleteUserFederation([user1, ...])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>user1</td>
<td>Specifies a comma-separated list of users whose federations are to be deleted. At least one user must be specified.</td>
</tr>
</tbody>
</table>

4.4.14.3 Example

The following command deletes federations for three users:

```
 wls:/mydomain/serverConfig> deleteUserFederation(['userA','userB','userC'])
```

4.4.15 changeMessageStore

Online command that changes the message store between memory and RDBMS.

4.4.15.1 Description

This command changes the message store to memory or RDBMS.

4.4.15.2 Syntax

changeMessageStore(type, [jndiname])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Specifies the type of store, RDBMS or Memory. Default is Memory.</td>
</tr>
<tr>
<td>jndiname</td>
<td>Specifies the jndi name to set for the store. Required if type is RDBMS.</td>
</tr>
</tbody>
</table>

4.4.15.3 Example

The following command changes the message store to RDBMS:

```
 wls:/mydomain/serverConfig> changeMessageStore('RDBMS','jdbc/mydb')
```

4.4.16 changePeerProviderDescription

Online command that changes the peer provider description.

4.4.16.1 Description

This command updates a peer provider description in cot.xml.

4.4.16.2 Syntax

changePeerProviderDescription(providerID, description)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the provider ID.</td>
</tr>
<tr>
<td>description</td>
<td>Specifies the provider description.</td>
</tr>
</tbody>
</table>

4.4.16.3 Example

The following command updates the description of a provider:

```
 wls:/mydomain/serverConfig> changePeerProviderDescription('providerA','new
### 4.4.17 changeSessionStore

Online command that changes the session store between memory and RDBMS.

#### 4.4.17.1 Description

This command changes the session store to memory or RDBMS.

#### 4.4.17.2 Syntax

```plaintext
changeSessionStore(type, [jndiname])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Specifies the type of store, RDBMS or Memory. Default is Memory.</td>
</tr>
<tr>
<td>jndiname</td>
<td>Specifies the jndi name to set for the store. Required if type is RDBMS.</td>
</tr>
</tbody>
</table>

#### 4.4.17.3 Example

The following command changes the session store to RDBMS.

```
wls:/mydomain/serverConfig> changeSessionStore('RDBMS','jdbc/mydb')
```

### 4.4.18 createConfigPropertyList

Online command that creates a property list.

#### 4.4.18.1 Description

This command creates a property list in config.xml.

#### 4.4.18.2 Syntax

```plaintext
createConfigPropertyList(configName, listName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be updated.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the property list name.</td>
</tr>
</tbody>
</table>

#### 4.4.18.3 Example

The following command creates property list mylistA.

```
wls:/mydomain/serverConfig> createConfigPropertyList('serverconfig','mylistA')
```

### 4.4.19 createConfigPropertyListInMap

Online command that creates a property list nested in the property map.

#### 4.4.19.1 Description

This command creates a property list, nested in the property map, in config.xml.

#### 4.4.19.2 Syntax

```plaintext
createConfigPropertyListInMap(configName, mapName, listName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be updated.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the property map name.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the property list name.</td>
</tr>
</tbody>
</table>
4.4.19.3 Example
The following command creates property list mylistA nested in a property map.

```
wls:/mydomain/serverConfig>
cREATECONFIGPROPERTYLISTINMAP('serverconfig','mymapA','mylistA')
```

4.4.20 createConfigPropertyMap
Online command that creates a property map.

4.4.20.1 Description
This command that creates a property map in config.xml.

4.4.20.2 Syntax
```
createConfigPropertyMap(configName, mapName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be updated.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies an existing property map to contain the nested list.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the property list name.</td>
</tr>
</tbody>
</table>

4.4.20.3 Example
The following command creates property map mymapA:

```
wls:/mydomain/serverConfig> createConfigPropertyMap('serverconfig','mymapA')
```

4.4.21 createConfigPropertyMapInMap
Online command that creates a property map.

4.4.21.1 Description
This command that creates a property map in config.xml.

4.4.21.2 Syntax
```
createConfigPropertyMapInMap(configName, mapName, nestedMapName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be updated.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of an existing property map.</td>
</tr>
<tr>
<td>nestedMapName</td>
<td>Specifies the name of the property map to create nested inside mapName.</td>
</tr>
</tbody>
</table>
4.4.21.3 Example
The following command creates nested property map `nestedmymapA`:

```wls:/mydomain/serverConfig>
createConfigPropertyMapInMap('serverconfig','mymapA','nestedmapA')
```

4.4.22 `createFederationPropertyList`
Online command that creates a property list.

4.4.22.1 Description
This command creates a property list in cot.xml.

4.4.22.2 Syntax

```python
createFederationPropertyList(providerID, listName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the provider ID.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the name of the property list.</td>
</tr>
</tbody>
</table>

4.4.22.3 Example
The following command creates property list `mylistA`:

```wls:/mydomain/serverConfig>
createFederationPropertyList('providerA','mylistA')
```

4.4.23 `createFederationPropertyListInMap`
Online command that creates a property list nested in a property map.

4.4.23.1 Description
This command creates a property list, nested in a property map, in cot.xml.

4.4.23.2 Syntax

```python
createFederationPropertyListInMap(providerID, mapName, listName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the provider ID.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies an existing property map to contain the nested list.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the name of the property list.</td>
</tr>
</tbody>
</table>

4.4.23.3 Example
The following command creates nested property list `mylistA`:

```wls:/mydomain/serverConfig>
createFederationPropertyListInMap('providerA','mymapA','mylistA')
```

4.4.24 `createFederationPropertyMap`
Online command that creates a property map.
4.4.24.1 Description
This command that creates a property map in cot.xml.

4.4.24.2 Syntax
createFederationPropertyMap(providerID, mapName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the provider ID.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map to be added to cot.xml.</td>
</tr>
</tbody>
</table>

4.4.24.3 Example
The following command creates property map mymapA:

```
wls:/mydomain/serverConfig> createFederationPropertyMap('providerA','mymapA')
```

4.4.25 createFederationPropertyMapInMap
Online command that creates a nested property map.

4.4.25.1 Description
This command that creates a property map, nested in another property map, in cot.xml.

4.4.25.2 Syntax
createFederationPropertyMapInMap(providerID, mapName, nestedMapName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the provider ID.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of an existing property map.</td>
</tr>
<tr>
<td>nestedMapName</td>
<td>Specifies the name of the property map to be nested inside mapName in cot.xml.</td>
</tr>
</tbody>
</table>

4.4.25.3 Example
The following command creates nested property map nestedmapA:

```
wls:/mydomain/serverConfig> createFederationPropertyMapInMap('providerA','mymapA','nestedmapA')
```

4.4.26 createPeerProviderEntry
Online command that creates a peer provider property map entry.

4.4.26.1 Description
This command creates a peer provider as a Map property entry to cot.xml.

4.4.26.2 Syntax
createPeerProviderEntry(providerID, description, providerType, version)
4.4.26.3 Example
The following command creates a SAML 2.0 service provider:

```wls:/mydomain/serverConfig> createPeerProviderEntry('providerA','idptest','SP','SAML2.0')```

4.4.27 `getConfigListValueInMap`

Online command that returns a list nested in a map.

4.4.27.1 Description
This command returns a list, nested in a map, from config.xml.

4.4.27.2 Syntax
```
getConfigListValueInMap(configName, mapName, listName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>configName</code></td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be accessed.</td>
</tr>
<tr>
<td><code>mapName</code></td>
<td>Specifies the name of the property map.</td>
</tr>
<tr>
<td><code>listName</code></td>
<td>Specifies the name of the list to be fetched from the map.</td>
</tr>
</tbody>
</table>

4.4.27.3 Example
The following command returns `mylistA`:

```wls:/mydomain/serverConfig> getConfigListValueInMap('serverConfig','mymapA','mylistA')```

4.4.28 `getConfigMapEntryInMap`

Online command that returns a map property entry nested in a map.

4.4.28.1 Description
This command returns a map property entry, nested in a map, from config.xml.

4.4.28.2 Syntax
```
getConfigMapEntryInMap(configName, mapname, nestedMapName, propName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>configName</code></td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be accessed.</td>
</tr>
<tr>
<td><code>mapname</code></td>
<td>Specifies the name of the property map.</td>
</tr>
<tr>
<td><code>nestedMapName</code></td>
<td>Specifies the name of the nested map.</td>
</tr>
<tr>
<td><code>propName</code></td>
<td>Specifies the name of the property map.</td>
</tr>
</tbody>
</table>
4.4.28.3 Example
The following command returns property entry myvarA:
```
<wls:/mydomain/serverConfig>
getConfigMapEntryInMap('serverconfig','mymap','nestedmapA','myvarA')
```

4.4.29 getConfigProperty
Online command that returns a property value.

4.4.29.1 Description
This command returns a property value from config.xml.

4.4.29.2 Syntax
```
getConfigProperty(configName, propName)
```

4.4.29.3 Example
The following command returns property myvarA:
```
wls:/mydomain/serverConfig> getConfigProperty('serverconfig','myvarA')
```

4.4.30 getConfigPropertyList
Online command that returns a property list.

4.4.30.1 Description
This command returns a property list from config.xml.

4.4.30.2 Syntax
```
getConfigPropertyList(configName, listName)
```

4.4.30.3 Example
The following command returns mylistA:
```
wls:/mydomain/serverConfig> getConfigPropertyList('serverconfig','mylistA')
```
4.4.31 getConfigPropertyMapEntry

Online command that returns a property value from a map.

4.4.31.1 Description
This command returns a property value from a map in config.xml.

4.4.31.2 Syntax
getConfigPropertyMapEntry(configName, mapName, propName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the configuration name (for example, idpsaml20, serverconfig, spsaml20,...).</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map.</td>
</tr>
<tr>
<td>propName</td>
<td>Specifies the name of the property to be fetched from the map in config.xml.</td>
</tr>
</tbody>
</table>

4.4.31.3 Example
The following command returns property propA:

```
wls:/mydomain/serverConfig> getConfigPropertyMapEntry('serverconfig','mapA', 'propA')
```

4.4.32 getFederationListValueInMap

Online command that returns a list value nested in a map.

4.4.32.1 Description
This command returns a list value nested in a map from cot.xml.

4.4.32.2 Syntax
getFederationListValueInMap(providerID, mapName, listName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the name of the peer provider to be accessed.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the name of the list to be fetched from the map.</td>
</tr>
</tbody>
</table>

4.4.32.3 Example
The following command returns nested list mylistA:

```
wls:/mydomain/serverConfig>
getFederationListValueInMap('providerA','mymapA','mylistA')
```

4.4.33 getFederationMapEntryInMap

Online command that returns a map property entry nested in a map.

4.4.33.1 Description
This command returns a map property entry, nested in a map, from cot.xml.
4.4.33.2 Syntax

getFederationMapEntryInMap(providerID, mapname, nestedMapName, propName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the name of the peer provider to be accessed.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map.</td>
</tr>
<tr>
<td>nestedMapName</td>
<td>Specifies the name of the nested property map.</td>
</tr>
<tr>
<td>propName</td>
<td>Specifies the name of the property to be fetched from the nested map.</td>
</tr>
</tbody>
</table>

4.4.33.3 Example

The following command returns property entry myvarA:

```java
wls:/mydomain/serverConfig> getFederationMapEntryInMap('providerA','mymap','nestedmapA','myvarA')
```

4.4.34 getFederationProperty

Online command that returns a property value.

4.4.34.1 Description

This command returns a property value from cot.xml.

4.4.34.2 Syntax

getFederationProperty(providerID, propName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the name of the peer provider to be accessed.</td>
</tr>
<tr>
<td>propName</td>
<td>Specifies the name of the property to be fetched from cot.xml.</td>
</tr>
</tbody>
</table>

4.4.34.3 Example

The following command returns property myvarA:

```java
wls:/mydomain/serverConfig> getFederationProperty('providerA','myvarA')
```

4.4.35 getFederationPropertyList

Online command that returns a property list.

4.4.35.1 Description

This command returns a property list from cot.xml.

4.4.35.2 Syntax

getFederationPropertyList(providerID, listName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the name of the peer provider to be accessed.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the name of the list to be fetched from the map.</td>
</tr>
</tbody>
</table>
4.4.35.3 Example
The following command returns list mylistA:

```
wls:/mydomain/serverConfig> getFederationPropertyList('providerA','mylistA')
```

4.4.36 extractproviderprops
A WLST script that exports the properties of a provider.

4.4.36.1 Description
A WLST script that extracts all the configuration properties of the specified provider and exports them to a text file. You can later use this file to set the same properties on another provider. Execute this command from a UNIX or Windows command shell prompt and not from the WLST command shell. This script is stored in `ORACLE_HOME/fed/scripts`.

4.4.36.2 Syntax
```
extractproviderprops.py providerID filename
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the name of the provider whose properties are to be extracted.</td>
</tr>
<tr>
<td>filename</td>
<td>Specifies the name of the text file to which the provider properties are extracted.</td>
</tr>
</tbody>
</table>

When you execute the script, you are prompted for the WebLogic administrator credentials and the connection URL; for the latter, specify the Managed Server port, not the Administration Server port.

**File Format**
The format of the extract file is:

```
TYPE:NAME:PROPNAME:PROPVALUE:PROPTYPE
```

For example:

```
X:X:sendattribute:false:boolean
MAP:attributelist/mailemail:datastore-attr:mail:string
LIST:sendattributeformenamoid:unspecifed::string
```

4.4.37 setproviderprops
A WLST script that sets the properties of a provider using values from a text file.

4.4.37.1 Description
A WLST script that sets the properties of a provider using values from a text file. Execute this command from a UNIX or Windows command shell prompt and not from the WLST command shell. This script is stored in `ORACLE_HOME/fed/scripts`.

The text file is generated by the `extractproviderprops` command.

4.4.37.2 Syntax
```
setproviderprops.py providerID filename
```
When you execute the script, you are prompted for the WebLogic administrator credentials and the connection URL; for the latter, specify the Managed Server port, not the Administration Server port.

4.4.38 getFederationPropertyMapEntry

Online command that returns a property value from a map.

4.4.38.1 Description

This command returns a property value from a map in cot.xml.

4.4.38.2 Syntax

getFederationPropertyMapEntry(providerID, mapName, propName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the name of the provider whose properties are to be updated.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map.</td>
</tr>
<tr>
<td>propName</td>
<td>Specifies the name of the property to be fetched from the nested map.</td>
</tr>
</tbody>
</table>

4.4.38.3 Example

The following command returns property propA from a map:

wls:/mydomain/serverConfig> getFederationPropertyMapEntry('providerA','mapA', 'propA')

4.4.39 listCustomAuthnEngines

Online command that returns a list of custom authentication integration engines.

4.4.39.1 Description

This command returns a list of custom authentication integration engines from config.xml.

4.4.39.2 Syntax

listCustomAuthnEngines()

4.4.39.3 Example

The following command returns the list of all SP engines:

wls:/mydomain/serverConfig> listCustomAuthnEngines()

4.4.40 listCustomSPEngines

Online command that returns a list of custom SP integration engines.
4.4.40.1 Description
This command returns a list of custom service provider (SP) integration engines from config.xml.

4.4.40.2 Syntax
listCustomSPEngines()

4.4.40.3 Example
The following command returns the list of all SP integration engines:

```
wlst:/mydomain/serverConfig> listCustomSPEngines()
```

4.4.41 loadMetadata
Online command that loads metadata from an input file.

4.4.41.1 Description
This command loads metadata from an input file into cot.xml.

4.4.41.2 Syntax
loadMetadata(metadatafile,description)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>metadatafile</td>
<td>Specifies the metadata file of the peer provider to be added or updated.</td>
</tr>
<tr>
<td>description</td>
<td>This is a brief description of the peer provider to be loaded.</td>
</tr>
</tbody>
</table>

4.4.41.3 Example
The following command loads metadata from the file metadatafile.xml:

```
wls:/mydomain/serverConfig> loadMetadata('/home/metadatafile.xml','some description')
```

4.4.42 oifStatus
Online command that reports the current status of the Oracle Identity Federation application in the managed server to which WLST is connected.

4.4.42.1 Description
This command displays the current status of Oracle Identity Federation on the managed server.

4.4.42.2 Syntax
loifStatus('serverurl','configfile','keyfile')

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverurl</td>
<td>Specifies the URL of the managed server.</td>
</tr>
<tr>
<td>configfile</td>
<td>This is a pre-defined user configuration file created with the WLST storeUserConfig command.</td>
</tr>
</tbody>
</table>
### 4.4.42.3 Example

The following command provides no arguments; WLST prompts you for the Oracle WebLogic Server username, password, and the managed server URL, then displays the federation server status:

```
wls:/mydomain/serverConfig> oifStatus()
```

The following command provides only the managed server URL; WLST prompts you for the Oracle WebLogic Server username and password:

```
wls:/mydomain/serverConfig> oifStatus('', '', 't3://localhost:7499')
```

The following command provides all arguments needed for WLST to display the federation server status:

```
wls:/mydomain/serverConfig> oifStatus('configfileA', 'keyfileB',
                          't3://localhost:7499')
```

### 4.4.43 removeConfigListInMap

Online command that removes a list property nested in a map.

#### 4.4.43.1 Description

This command removes a list property nested in a map from config.xml.

#### 4.4.43.2 Syntax

```
removeConfigListInMap(configName, mapName, listName)
```

#### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be accessed.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the name of the list to be removed from the map.</td>
</tr>
</tbody>
</table>

#### 4.4.43.3 Example

The following command removes the list property mylistA:

```
wls:/mydomain/serverConfig>
removeConfigListInMap('serverConfig','mymapA','mylistA')
```

### 4.4.44 removeConfigMapEntryInMap

Online command that removes a map property nested in a map.

#### 4.4.44.1 Description

This command removes a map property entry nested in a map from config.xml.
4.4.44.2 Syntax
removeConfigMapEntryInMap(configName, mapname, nestedMapName, propName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be accessed.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map.</td>
</tr>
<tr>
<td>nestedMapName</td>
<td>Specifies the name of the nested property map.</td>
</tr>
<tr>
<td>propName</td>
<td>Specifies the name of the property to be removed from the nested map.</td>
</tr>
</tbody>
</table>

4.4.44.3 Example
The following command removes the nested property myvarA:

```bash
wls:/mydomain/serverConfig>
removeConfigMapEntryInMap('serverconfig','mymap','nestedmapA','myvarA')
```

4.4.45 removeConfigMapInMap
Online command that removes a map property nested in a map.

4.4.45.1 Description
This command removes a map property entry nested in a map from config.xml.

4.4.45.2 Syntax
removeConfigMapEntryInMap(configName, mapName, nestedMapName, propName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be updated.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map.</td>
</tr>
<tr>
<td>nestedMapName</td>
<td>Specifies the name of the nested property map.</td>
</tr>
<tr>
<td>propName</td>
<td>Specifies the name of the property to be removed from the nested map.</td>
</tr>
</tbody>
</table>

4.4.45.3 Example
The following command removes the nested property myvarA:

```bash
wls:/mydomain/serverConfig>
removeConfigMapEntryInMap('serverconfig','mymap','nestedmapA','myvarA')
```

4.4.46 removeConfigProperty
Online command that removes a configuration property.

4.4.46.1 Description
This command removes a property from config.xml.

4.4.46.2 Syntax
removeConfigProperty(configName, propName)
4.4.46.3 Example
The following command removes the property myvarA:
```
wlsc:/mydomain/serverConfig> removeConfigProperty('serverconfig','myvarA')
```

4.4.47 removeConfigPropertyList
Online command that removes a configuration property list.

4.4.47.1 Description
This command removes a property list from config.xml.

4.4.47.2 Syntax
```
removeConfigPropertyList(configName, listName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be updated.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the name of the property list to be removed.</td>
</tr>
</tbody>
</table>

4.4.47.3 Example
The following command removes the property list mylistA:
```
wlsc:/mydomain/serverConfig> removeConfigPropertyList('serverconfig','mylistA')
```

4.4.48 removeConfigPropertyMap
Online command that removes a property map.

4.4.48.1 Description
This command removes a property map in config.xml.

4.4.48.2 Syntax
```
removeConfigPropertyMap(configName, mapName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be updated.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map to be removed.</td>
</tr>
</tbody>
</table>

4.4.48.3 Example
The following command removes mapA:
```
wls:/mydomain/serverConfig> removeConfigPropertyMap('serverconfig','mapA')
```
4.4.49  removeConfigPropertyMapEntry

Online command that removes a property value from a map.

4.4.49.1  Description
This command removes a property value from a map in config.xml.

4.4.49.2  Syntax
removeConfigPropertyMapEntry(configName, mapName, propName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configName</td>
<td>Specifies the name of the configuration (for example, idpsaml20, serverconfig, spsaml20, ...) to be updated.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map to be updated.</td>
</tr>
<tr>
<td>propName</td>
<td>Specifies the name of the property to be removed from the map.</td>
</tr>
</tbody>
</table>

4.4.49.3  Example
The following command removes property propA:

wls:/mydomain/serverConfig> removeConfigPropertyMapEntry('serverconfig','mapA','propA')

4.4.50  removeFederationListInMap

Online command that removes a property list in a map.

4.4.50.1  Description
This command removes a property list in a map, in cot.xml.

4.4.50.2  Syntax
removeFederationListInMap(providerID, mapName, listName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the name of the peer provider to be accessed.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the name of the property list to be removed.</td>
</tr>
</tbody>
</table>

4.4.50.3  Example
The following command removes mylistA in mymapA:

wls:/mydomain/serverConfig> removeFederationListInMap('providerA','mymapA','mylistA')

4.4.51  removeFederationMapInMap

Online command that removes a nested map in a map.

4.4.51.1  Description
This command removes a property map nested inside a map in cot.xml.
4.4.51.2 Syntax

`removeFederationMapInMap(providerID, mapname, nestedMapName)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>providerID</code></td>
<td>Specifies the name of the peer provider to be accessed.</td>
</tr>
<tr>
<td><code>mapName</code></td>
<td>Specifies the name of the property map containing the nested map.</td>
</tr>
<tr>
<td><code>nestedMapName</code></td>
<td>Specifies the name of the nested property map to be removed.</td>
</tr>
</tbody>
</table>

4.4.51.3 Example

The following command removes `nestedmapA` in `mymap`:

```
wls:/mydomain/serverConfig>
removeFederationMapInMap('providerA','mymap','nestedmapA')
```

4.4.52 `removeFederationMapEntryInMap`

Online command that removes a nested map property entry in a map.

4.4.52.1 Description

This command removes a property name/value pair to a map nested inside a map in `cot.xml`.

4.4.52.2 Syntax

`removeFederationMapEntryInMap(providerID, mapname, nestedMapName, propName)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>providerID</code></td>
<td>Specifies the name of the peer provider to be accessed.</td>
</tr>
<tr>
<td><code>mapName</code></td>
<td>Specifies the name of the property map containing the nested map.</td>
</tr>
<tr>
<td><code>nestedMapName</code></td>
<td>Specifies the name of the nested property map.</td>
</tr>
<tr>
<td><code>propName</code></td>
<td>Specifies the name of the property to be removed from the nested map.</td>
</tr>
</tbody>
</table>

4.4.52.3 Example

The following command removes map property entry `myvarA`:

```
wls:/mydomain/serverConfig>
removeFederationMapEntryInMap('providerA','mymap','nestedmapA','myvarA')
```

4.4.53 `removeFederationProperty`

Online command that removes a property value.

4.4.53.1 Description

This command removes a property entry in `cot.xml`.

4.4.53.2 Syntax

`removeFederationProperty(providerID, propName)`
4.4.53  Example
The following command removes the provider property myvarA:

```
wls:/mydomain/serverConfig> removeFederationProperty('providerA','myvarA')
```

4.4.54  removeFederationPropertyList
Online command that removes a property list entry.

4.4.54.1  Description
This command removes a property list entry in cot.xml.

4.4.54.2  Syntax
```
removeFederationPropertyList(providerID, listName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the name of the peer provider to be accessed.</td>
</tr>
<tr>
<td>listName</td>
<td>Specifies the name of the property list to be removed.</td>
</tr>
</tbody>
</table>

4.4.54.3  Example
The following command removes mylistA:

```
wls:/mydomain/serverConfig> removeFederationPropertyList('providerA','mylistA')
```

4.4.55  removeFederationPropertyMap
Online command that removes a property map.

4.4.55.1  Description
This command removes a property map in cot.xml.

4.4.55.2  Syntax
```
removeFederationPropertyMap(providerID, mapName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the name of the peer provider to be accessed.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map to be removed.</td>
</tr>
</tbody>
</table>

4.4.55.3  Example
The following command removes a map:

```
wls:/mydomain/serverConfig> removeFederationPropertyMap('providerA','mapA')
```
4.4.56 removeFederationPropertyMapEntry

Online command that removes a property value from a map.

4.4.56.1 Description
This command removes a property value from a map in cot.xml.

4.4.56.2 Syntax
removeFederationPropertyMapEntry(providerID, mapName, propName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the name of the peer provider to be accessed.</td>
</tr>
<tr>
<td>mapName</td>
<td>Specifies the name of the property map to be updated.</td>
</tr>
<tr>
<td>propName</td>
<td>Specifies the name of the property to be removed from the map.</td>
</tr>
</tbody>
</table>

4.4.56.3 Example
The following command removes property propA from a map:

```
wls:/mydomain/serverConfig> removeFederationPropertyMapEntry('providerA','mapA','propA')
```

4.4.57 removePeerProviderEntry

Online command that removes a peer provider entry.

4.4.57.1 Description
This command removes a peer provider entry from cot.xml.

4.4.57.2 Syntax
removePeerProviderEntry(providerID)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the name of the peer provider to be removed.</td>
</tr>
</tbody>
</table>

4.4.57.3 Example
The following command removes providerA:

```
wls:/mydomain/serverConfig> removePeerProviderEntry('providerA')
```

4.4.58 setConfigProperty

Online command that sets a property value in config.xml.

4.4.58.1 Description
This command adds or updates a property value in config.xml.

4.4.58.2 Syntax
setConfigProperty(configname, propName, value, type)
Example
The following command sets the property `myvarA` and its value in the server configuration:

```
setConfigProperty('serverconfig','myvarA','myvalA','string')
```

**setCustomAuthnEngine**

Online command that updates a custom authentication integration engine.

### 4.4.59.1 Description
This command updates a custom authentication integration engine in config.xml.

### 4.4.59.2 Syntax

```
setCustomAuthnEngine(engineID, name, [enabled], [webContext], [authnRelativePath], [logoutRelativePath], [logoutEnabled])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>engineID</td>
<td>Specifies the engine ID of an existing engine.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the custom engine.</td>
</tr>
<tr>
<td>enabled</td>
<td>This flag specifies whether the engine is enabled (true) or not (false).</td>
</tr>
<tr>
<td>webContext</td>
<td>Specifies the web context for the engine.</td>
</tr>
<tr>
<td>authnRelativePath</td>
<td>Specifies the authentication relative path URL for the engine.</td>
</tr>
<tr>
<td>logoutRelativePath</td>
<td>Specifies the logout relative path URL for the engine.</td>
</tr>
<tr>
<td>logoutEnabled</td>
<td>This flag is set true to enable logout for the engine, else false.</td>
</tr>
</tbody>
</table>

**Example**
The following command updates the configuration of custom authentication engine `abcdef`:

```
wls:/mydomain/serverConfig> setCustomAuthnEngine('abcdef', 'custom one', 'false', 'oracle:fed:authentication:unspecified', 'webcontext')
```

**setCustomSPEngine**

Online command that updates a custom SP integration engine.

### 4.4.60.1 Description
This command updates an existing custom SP integration engine in config.xml.
4.4.60.2 Syntax

```
setCustomSPEngine(engineID, name, [enabled, [authnMech], [webContext],
[authnRelativePath], [logoutRelativePath], [logoutEnabled])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>engineID</td>
<td>Specifies the engine ID of an existing custom engine.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the custom engine.</td>
</tr>
<tr>
<td>enabled</td>
<td>This flag specifies whether the engine is enabled (true) or not (false).</td>
</tr>
<tr>
<td>authnMech</td>
<td>Specifies the authentication mechanism for the engine.</td>
</tr>
<tr>
<td>webContext</td>
<td>Specifies the web context for the engine.</td>
</tr>
<tr>
<td>authnRelativePath</td>
<td>Specifies the authentication relative path URL for the engine.</td>
</tr>
<tr>
<td>logoutRelativePath</td>
<td>Specifies the logout relative path URL for the engine.</td>
</tr>
<tr>
<td>logoutEnabled</td>
<td>This flag is set true to enable logout for the engine, else false.</td>
</tr>
</tbody>
</table>

4.4.60.3 Example

The following command sets the name and the enabled flag for the engine with ID engineID2:

```
wls:/mydomain/serverConfig> setCustomSPEngine('engineid2','test','true')
```

4.4.61 setFederationProperty

Online command that adds or updates a property value.

4.4.61.1 Description

This command adds a property entry or updates an existing entry in cot.xml.

4.4.61.2 Syntax

```
setFederationProperty(providerID, propName, value, type)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerID</td>
<td>Specifies the name of the peer provider to be updated.</td>
</tr>
<tr>
<td>propName</td>
<td>Specifies the name of the property to be added/updated in cot.xml.</td>
</tr>
<tr>
<td>value</td>
<td>Specifies the property value.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of property, BOOLEAN or STRING or LONG.</td>
</tr>
</tbody>
</table>

4.4.61.3 Example

The following command creates the property myvarA and sets its value:

```
wls:/mydomain/serverConfig>
setFederationProperty('providerA','myvarA','myvalA','string')
```

4.5 Directory Integration Platform Commands

Some of the Directory Integration Platform (DIP) tools use WLST internally, and therefore, there are no custom WLST commands available to run from the WLST command prompt or to use within scripts. For information on DIP tools, see "Directory
Integration Platform Tools” in the Oracle Fusion Middleware Reference for Oracle Identity Management.

4.6 Security Commands

Use the WLST security commands listed in Table 4–5 to operate on a domain policy or credential store, to migrate policies and credentials from a source repository to a target repository, and to import and export (credential) encryption keys.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>addBootStrapCredential</td>
<td>Add a credential to the bootstrap credential store.</td>
<td>Offline</td>
</tr>
<tr>
<td>addResourceToEntitlement</td>
<td>Add a resource to an entitlement.</td>
<td>Online</td>
</tr>
<tr>
<td>createAppRole</td>
<td>Create a new application role.</td>
<td>Online</td>
</tr>
<tr>
<td>createCred</td>
<td>Create a new credential.</td>
<td>Online</td>
</tr>
<tr>
<td>createEntitlement</td>
<td>Create an entitlement.</td>
<td>Online</td>
</tr>
<tr>
<td>createResource</td>
<td>Create a resource.</td>
<td>Online</td>
</tr>
<tr>
<td>createResourceType</td>
<td>Create a new resource type.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAppPolicies</td>
<td>Remove all policies in an application.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAppRole</td>
<td>Remove an application role.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteCred</td>
<td>Remove a credential.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteEntitlement</td>
<td>Remove an entitlement.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteResource</td>
<td>Remove a resource.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteResourceType</td>
<td>Remove an existing resource type.</td>
<td>Online</td>
</tr>
<tr>
<td>exportEncryptionKey</td>
<td>Export the domain encryption key to the file ewallet.p12.</td>
<td>Offline</td>
</tr>
<tr>
<td>getEntitlement</td>
<td>List an entitlement.</td>
<td>Online</td>
</tr>
<tr>
<td>getResourceType</td>
<td>Fetch an existing resource type.</td>
<td>Online</td>
</tr>
<tr>
<td>grantAppRole</td>
<td>Add a principal to a role.</td>
<td>Online</td>
</tr>
<tr>
<td>grantEntitlement</td>
<td>Create an entitlement.</td>
<td>Online</td>
</tr>
<tr>
<td>grantPermission</td>
<td>Create a new permission.</td>
<td>Online</td>
</tr>
<tr>
<td>importEncryptionKey</td>
<td>Import the encryption key in file ewallet.p12 to the domain.</td>
<td>Offline</td>
</tr>
<tr>
<td>listAppRoles</td>
<td>List all roles in an application.</td>
<td>Online</td>
</tr>
<tr>
<td>listAppRolesMembers</td>
<td>List all members in an application role.</td>
<td>Online</td>
</tr>
<tr>
<td>listAppStripes</td>
<td>List application stripes in policy store.</td>
<td>Online</td>
</tr>
<tr>
<td>listCodeSourcePermissions</td>
<td>List permissions assigned to a source code in global policies.</td>
<td>Online</td>
</tr>
<tr>
<td>listEntitlements</td>
<td>List entitlements in an application stripe.</td>
<td>Online</td>
</tr>
<tr>
<td>listPermissions</td>
<td>List all permissions granted to a principal.</td>
<td>Online</td>
</tr>
<tr>
<td>listResources</td>
<td>List resources in an application stripe.</td>
<td>Online</td>
</tr>
</tbody>
</table>
4.6.1 createAppRole

Online command that creates a new application role.

4.6.1.1 Description

Creates a new application role in the domain policy store with a given application and role name. In the event of an error, the command returns a WLSTException.

4.6.1.2 Syntax

createAppRole(appStripe, appRoleName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies an application stripe.</td>
</tr>
<tr>
<td>appRoleName</td>
<td>Specifies a role name.</td>
</tr>
</tbody>
</table>

4.6.1.3 Example

The following invocation creates a new application role with application stripe myApp and role name myRole:

wls:/mydomain/serverConfig> createAppRole(appStripe="myApp", appRoleName="myRole")
4.6.2 deleteAppRole

Online command that removes an application role.

4.6.2.1 Description
Removes an application role in the domain policy store with a given application and role name. In the event of an error, the command returns a WLSTException.

4.6.2.2 Syntax
createAppRole(appStripe, appRoleName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies an application stripe.</td>
</tr>
<tr>
<td>appRoleName</td>
<td>Specifies a role name.</td>
</tr>
</tbody>
</table>

4.6.2.3 Example
The following invocation removes the role with application stripe myApp and role name myRole:

```
wls:/mydomain/serverConfig> createAppRole(appStripe="myApp", appRoleName="myRole")
```

4.6.3 grantAppRole

Online command that adds a principal to a role.

4.6.3.1 Description
Adds a principal (class or name) to a role with a given application stripe and name. In the event of an error, the command returns a WLSTException.

4.6.3.2 Syntax
grantAppRole(appStripe, appRoleName,principalClass, principalName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies an application stripe.</td>
</tr>
<tr>
<td>appRoleName</td>
<td>Specifies a role name.</td>
</tr>
<tr>
<td>principalClass</td>
<td>Specifies the fully qualified name of a class.</td>
</tr>
<tr>
<td>principalName</td>
<td>Specifies the principal name.</td>
</tr>
</tbody>
</table>

4.6.3.3 Example
The following invocation adds a principal to the role with application stripe myApp and role name myRole:

```
wls:/mydomain/serverConfig> grantAppRole(appStripe="myApp", appRoleName="myRole",principalClass="com.example.xyzPrincipal", principalName="myPrincipal")
```

4.6.4 revokeAppRole

Online command that removes a principal from a role.
4.6.4.1 Description
Removes a principal (class or name) from a role with a given application stripe and
name. In the event of an error, the command returns a WLSTException.

4.6.4.2 Syntax
revokeAppRole(appStripe, appRoleName, principalClass, principalName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies an application stripe.</td>
</tr>
<tr>
<td>appRoleName</td>
<td>Specifies a role name.</td>
</tr>
<tr>
<td>principalClass</td>
<td>Specifies the fully qualified name of a class.</td>
</tr>
<tr>
<td>principalName</td>
<td>Specifies the principal name.</td>
</tr>
</tbody>
</table>

4.6.4.3 Example
The following invocation removes a principal to the role with application stripe myApp
and role name myRole:

```shell
wls:/mydomain/serverConfig> revokeAppRole(appStripe="myApp",
appRoleName="myRole",principalClass="com.example.xyzPrincipal",
principalName="myPrincipal")
```

4.6.5 listAppRoles
Online command that lists all roles in an application.

4.6.5.1 Description
Lists all roles within a given application stripe. In the event of an error, the command
returns a WLSTException.

4.6.5.2 Syntax
listAppRoles(appStripe)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies an application stripe.</td>
</tr>
</tbody>
</table>

4.6.5.3 Example
The following invocation returns all roles with application stripe myApp:

```shell
wls:/mydomain/serverConfig> listAppRoles(appStripe="myApp")
```

4.6.6 listAppRoleMembers
Online command that lists all members in a role.

4.6.6.1 Description
Lists all members in a role with a given application stripe and role name. In the event
of an error, the command returns a WLSTException.

4.6.6.2 Syntax
listAppRoleMembers(appStripe, appRoleName)
4.6.6.3 Example
The following invocation returns all members in the role with application stripe myApp and role name myRole:

```
wlst:/mydomain/serverConfig> listAppRoleMembers(appStripe="myApp",
    appRoleName="myRole")
```

4.6.7 grantPermission
Online command that creates a new permission.

4.6.7.1 Description
Creates a new permission for a given code base or URL. In the event of an error, the command returns a WLSTException.

4.6.7.2 Syntax
Optional arguments are enclosed in between square brackets.

```
grantPermission([appStripe,] [codeBaseURL,] [principalClass,] [principalName,]
    permClass, [permTarget,] [permActions])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies an application stripe. If not specified, the command works on system policies.</td>
</tr>
<tr>
<td>codeBaseURL</td>
<td>Specifies the URL of the code granted the permission.</td>
</tr>
<tr>
<td>principalClass</td>
<td>Specifies the fully qualified name of a class (grantee).</td>
</tr>
<tr>
<td>principalName</td>
<td>Specifies the name of the grantee principal.</td>
</tr>
<tr>
<td>permClass</td>
<td>Specifies the fully qualified name of the permission class.</td>
</tr>
<tr>
<td>permTarget</td>
<td>Specifies, when available, the name of the permission target. Some permissions may not include this attribute.</td>
</tr>
<tr>
<td>permActions</td>
<td>Specifies a comma-separated list of actions granted. Some permissions may not include this attribute and the actions available depend on the permission class.</td>
</tr>
</tbody>
</table>

4.6.7.3 Examples
The following invocation creates a new application permission (for the application with application stripe myApp) with the specified data:

```
wlst:/mydomain/serverConfig> grantPermission(appStripe="myApp",
    principalClass="my.custom.Principal", principalName="manager",
    permClass="java.security.AllPermission")
```

The following invocation creates a new system permission with the specified data:

```
wlst:/mydomain/serverConfig> grantPermission(principalClass="my.custom.Principal",
    principalName="manager",
    permClass="java.io.FilePermission", permTarget="/tmp/fileName.ext",
    permTarget="/tmp/fileName.ext", permActions="read,write")
```
4.6.8 revokePermission

Online command that removes a permission.

4.6.8.1 Description
Removes a permission for a given code base or URL. In the event of an error, the command returns a WLSTException.

4.6.8.2 Syntax
Optional arguments are enclosed in between square brackets.

revokePermission([appStripe,] [codeBaseURL,] [principalClass,] [principalName,] permClass, [permTarget,] [permActions])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies an application stripe. If not specified, the command works on system policies.</td>
</tr>
<tr>
<td>codeBaseURL</td>
<td>Specifies the URL of the code granted the permission.</td>
</tr>
<tr>
<td>principalClass</td>
<td>Specifies the fully qualified name of a class (grantee).</td>
</tr>
<tr>
<td>principalName</td>
<td>Specifies the name of the grantee principal.</td>
</tr>
<tr>
<td>permClass</td>
<td>Specifies the fully qualified name of the permission class.</td>
</tr>
<tr>
<td>permTarget</td>
<td>Specifies, when available, the name of the permission target. Some permissions may not include this attribute.</td>
</tr>
<tr>
<td>permActions</td>
<td>Specifies a comma-separated list of actions granted. Some permissions may not include this attribute and the actions available depend on the permission class.</td>
</tr>
</tbody>
</table>

4.6.8.3 Examples
The following invocation removes the application permission (for the application with application stripe myApp) with the specified data:

wls:/mydomain/serverConfig> revokePermission(appStripe="myApp", principalClass="my.custom.Principal", principalName="manager", permClass="java.security.AllPermission")

The following invocation removes the system permission with the specified data:

wls:/mydomain/serverConfig> revokePermission(principalClass="my.custom.Principal", principalName="manager", permClass="java.io.FilePermission", permTarget="/tmp/fileName.ext", permActions="read,write")

4.6.9 listPermissions

Online command that lists all permissions granted to a given principal.

4.6.9.1 Description
Lists all permissions granted to a given principal. In the event of an error, the command returns a WLSTException.

4.6.9.2 Syntax
Optional arguments are enclosed in between square brackets.
listPermissions([appStripe,] principalClass, principalName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies an application stripe. If not specified, the command works on system policies.</td>
</tr>
<tr>
<td>principalClass</td>
<td>Specifies the fully qualified name of a class (grantee).</td>
</tr>
<tr>
<td>principalName</td>
<td>Specifies the name of the grantee principal.</td>
</tr>
</tbody>
</table>

### 4.6.9.3 Examples

The following invocation lists all permissions granted to a principal by the policies of application `myApp`:

```
wlst:/mydomain/serverConfig> listPermissions(appStripe="myApp", principalClass="my.custom.Principal", principalName="manager")
```

The following invocation lists all permissions granted to a principal by system policies:

```
wlst:/mydomain/serverConfig> listPermissions(principalClass="my.custom.Principal", principalName="manager")
```

### 4.6.10 deleteAppPolicies

Online command that removes all policies with a given application stripe.

### 4.6.10.1 Description

Removes all policies with a given application stripe. In the event of an error, the command returns a `WLSTException`.

### 4.6.10.2 Syntax

```
deleteAppPolicies(appStripe)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies an application stripe. If not specified, the command works on system policies.</td>
</tr>
</tbody>
</table>

### 4.6.10.3 Example

The following invocation removes all policies of application `myApp`:

```
wls:/mydomain/serverConfig> deleteAppPolicies(appStripe="myApp")
```

### 4.6.11 migrateSecurityStore

Offline command that migrates identities, policies (application-specific and system), credentials, audit, and keystore service artifacts from one repository to another.

For details, see the following sections in Oracle Fusion Middleware Application Security Guide:

- Migrating Identities
- Migrating Policies and Credentials
- Migrating Audit Information
### Migrating Keystore Service Artifacts

#### 4.6.11.1 Description

Migrates identities, application-specific, or system policies from a source repository to a target repository. Migrates a specific credential folder or all credentials.

The kinds of the repositories where the source and target data is stored is transparent to the command, and any combination of file-based and LDAP-based repositories is allowed (LDAP-repositories must use an OVD or an OID LDAP server only). In the event of an error, the command returns a WLSTException.

#### 4.6.11.2 Syntax

The command syntax varies depending on the scope (system or application-specific or both) of the policies being migrated.

Optional arguments are enclosed in square brackets.

To migrate identities, use the following syntax:

```
 migratingSecurityStore(type="idStore", configFile, src, dst, [dstLdifFile])
```

To migrate all policies (system and application-specific, for all applications) use the following syntax:

```
migrateSecurityStore(type="policyStore", configFile, src, dst, [overwrite,][preserveAppRoleGuid])
```

To migrate just system policies, use the following syntax:

```
migrateSecurityStore(type="globalPolicies", configFile, src, dst, [overwrite])
```

To migrate just application-specific policies, for one application, use the following syntax:

```
```

To migrate all credentials, use the following syntax:

```
migrateSecurityStore(type="credStore", configFile, src, dst, [overwrite])
```

To migrate just one credential folder, use the following syntax:

```
migrateSecurityStore(type="folderCred", configFile, src, dst, [srcFolder,] [dstFolder,] [srcConfigFile,] [overwrite])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Specifies the type of policies migrates.</td>
</tr>
<tr>
<td></td>
<td>To migrate identities, set it to <code>idStore</code>.</td>
</tr>
<tr>
<td></td>
<td>To migrate all policies (system and application-specific, for all applications), set to <code>policyStore</code>.</td>
</tr>
<tr>
<td></td>
<td>To migrate just system policies, set to <code>globalPolicies</code>.</td>
</tr>
<tr>
<td></td>
<td>To migrate just application-specific policies, set to <code>appPolicies</code>.</td>
</tr>
<tr>
<td></td>
<td>To migrate all credentials, set to <code>credStore</code>.</td>
</tr>
<tr>
<td></td>
<td>To migrate just one credential folder, set to <code>folderCred</code>.</td>
</tr>
</tbody>
</table>
Note the following requirements about the passed arguments:

- The file jps-config.xml is found in the passed location.
The file jps-config.xml includes the passed jps-contexts.

The source and the destination context names are distinct. From these two contexts, the command determines the locations of the source and the target repositories involved in the migration.

4.6.11.3 Example

The following invocation illustrates the migration of the file-based policies of application PolicyServlet1 to file-based policies of application PolicyServlet2, that does not stop on encountering duplicate principals or permissions, that migrates just one of duplicate items, and that logs a warning when duplicates are found:

```bash
wls:/mydomain/serverConfig> migrateSecurityStore(type="appPolicies", configFile="jps-config.xml", src="default1", dst="context2", srcApp="PolicyServlet1", dstApp="PolicyServlet2", overWrite="true", mode="lax")
```

The above invocation assumes that:

- The file jps-config.xml is located in the directory where the command is run (current directory).
- That file includes the following elements:

  ```xml
  <serviceInstance name="policystore1.xml" provider="some.provider">
   <property name="location" value="jazn-data1.xml"/>
  </serviceInstance>
  <serviceInstance name="policystore2.xml" provider="some.provider">
   <property name="location" value="jazn-data2.xml"/>
  </serviceInstance>
  ...
  <jpsContext name="default1">
   <serviceInstanceRef ref="policystore1.xml"/>
  ...
  </jpsContext>
  <jpsContext name="context2">
   <serviceInstanceRef ref="policystore2.xml"/>
  ...
  </jpsContext>
  ...
  
  The file-based policies for the two applications involved in the migration are defined in the files jazn-data1.xml and jazn-data2.xml, which are not shown but assumed located in the current directory.

The following invocation illustrates the migration of file-based credentials from one location to another:

```bash
wls:/mydomain/serverConfig> migrateSecurityStore(type="credStore", configFile="jps-config.xml", src="default1", dst="context2")
```

The above invocation assumes that:

- The file jps-config.xml is located in the directory where the command is run (current directory).
- That file includes the following elements:

  ```xml
  <serviceInstance name="credstore1" provider="some.provider">
   <property name="location" value="/credstore1/cwallet.sso"/>
  </serviceInstance>
  <serviceInstance name="credstore2" provider="some.provider">
   <property name="location" value="/credstore2/cwallet.sso"/>
  </serviceInstance>
  ```
For detailed configuration examples to use with this command, see Oracle Fusion Middleware Application Security Guide.

### 4.6.12 updateCred

Online command that modifies the type, user name, and password of a credential.

#### 4.6.12.1 Description

Modifies the type, user name, password, URL, and port number of a credential in the domain credential store with given map name and key name. This command can update the data encapsulated in credentials of type password only. In the event of an error, the command returns a WLSTException. This command runs in interactive mode only. When the command executes successfully no error messages are displayed.

#### 4.6.12.2 Syntax

Optional arguments are enclosed in square brackets.

`updateCred(map, key, user, password, [desc])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>map</code></td>
<td>Specifies a map name (folder).</td>
</tr>
<tr>
<td><code>key</code></td>
<td>Specifies a key name.</td>
</tr>
<tr>
<td><code>user</code></td>
<td>Specifies the credential user name.</td>
</tr>
<tr>
<td><code>password</code></td>
<td>Specifies the credential password.</td>
</tr>
<tr>
<td><code>desc</code></td>
<td>Specifies a string describing the credential.</td>
</tr>
</tbody>
</table>

#### 4.6.12.3 Example

The following invocation updates a password credential with the specified data:

`wls:/mydomain/serverConfig> updateCred(map="myMap", key="myKey", user="myUser", password="myPassw", desc="updated passw cred to connect to app xyz")`

### 4.6.13 createCred

Online command that creates a new credential in the domain credential store.

#### 4.6.13.1 Description

Creates a new credential in the domain credential store with a given map name, key name, type, user name and password, URL and port number. In the event of an error, the command returns a WLSTException. This command runs in interactive mode only. When the command executes successfully no error messages are displayed.
4.6.13.2 Syntax
Optional arguments are enclosed in square brackets.
createCred(map, key, user, password, [desc])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>map</td>
<td>Specifies a map name (folder).</td>
</tr>
<tr>
<td>key</td>
<td>Specifies a key name.</td>
</tr>
<tr>
<td>user</td>
<td>Specifies the credential user name.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the credential password.</td>
</tr>
<tr>
<td>desc</td>
<td>Specifies a string describing the credential.</td>
</tr>
</tbody>
</table>

4.6.13.3 Example
The following invocation creates a new password credential with the specified data:

```
wlst:/mydomain/serverConfig> createCred(map="myMap, key="myKey", user="myUsr", password="myPassw", desc="updated usr name and passw to connect to app xyz")
```

4.6.14 deleteCred
Online command that removes a credential in the domain credential store.

4.6.14.1 Description
Removes a credential with given map name and key name from the domain credential store. In the event of an error, the command returns a WLSTException. When the command executes successfully no error messages are displayed.

4.6.14.2 Syntax
deleteCred(map,key)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>map</td>
<td>Specifies a map name (folder).</td>
</tr>
<tr>
<td>key</td>
<td>Specifies a key name.</td>
</tr>
</tbody>
</table>

4.6.14.3 Example
The following invocation removes the credential with map name myMap and key name myKey:

```
wls:/mydomain/serverConfig> deleteCred(map="myApp",key="myKey")
```

4.6.15 modifyBootStrapCredential
Offline command that updates a bootstrap credential store.

4.6.15.1 Description
Updates a bootstrap credential store with given user name and password. In the event of an error, the command returns a WLSTException.

Typically used in the following scenario: suppose that the domain policy and credential stores are LDAP-based, and the credentials to access the LDAP store (stored
in the LDAP server) are changed. Then this command can be used to seed those changes into the bootstrap credential store.

### 4.6.15.2 Syntax

```plaintext
modifyBootStrapCredential(jpsConfigFile, username, password)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>jpsConfigFile</td>
<td>Specifies the location of the file <code>jps-config.xml</code> relative to the location where the command is run.</td>
</tr>
<tr>
<td>username</td>
<td>Specifies the distinguished name of the user in the LDAP store.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the user.</td>
</tr>
</tbody>
</table>

### 4.6.15.3 Example

Suppose that in the LDAP store, the password of the user with distinguished name `cn=orcladmin` has been changed to `welcome1`, and that the configuration file `jps-config.xml` is located in the current directory.

Then the following invocation changes the password in the bootstrap credential store to `welcome1`:

```plaintext
wls:/mydomain/serverConfig>
modifyBootStrapCredential(jpsConfigFile='./jps-config.xml', username='cn=orcladmin', password='welcome1')
```

Any output regarding the audit service can be disregarded.

### 4.6.16 addBootStrapCredential

Offline command that adds a credential to the bootstrap credential store.

### 4.6.16.1 Description

Adds a password credential with the given map, key, user name, and user password to the bootstrap credentials configured in the default JPS context of a JPS configuration file. In the event of an error, the command returns a `WLSTException`.

### 4.6.16.2 Syntax

```plaintext
addBootStrapCredential(jpsConfigFile, map, key, username, password)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>jpsConfigFile</td>
<td>Specifies the location of the file <code>jps-config.xml</code> relative to the location where the command is run.</td>
</tr>
<tr>
<td>map</td>
<td>Specifies the map of the credential to add.</td>
</tr>
<tr>
<td>key</td>
<td>Specifies the key of the credential to add.</td>
</tr>
<tr>
<td>username</td>
<td>Specifies the name of the user in the credential to add.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the password of the user in the credential to add.</td>
</tr>
</tbody>
</table>

### 4.6.16.3 Example

The following invocation adds a credential to the bootstrap credential store:

```plaintext
wls:/mydomain/serverConfig>
addBootStrapCredential(jpsConfigFile='./jps-config.xml', map='myMapName',
```
4.6.17 exportEncryptionKey

Offline command that extracts the encryption key from a domain's bootstrap wallet to the file ewallet.p12.

4.6.17.1 Description

Writes the domain's credential encryption key to the file ewallet.p12. The password passed must be used to import data from that file with the command importEncryptionKey.

4.6.17.2 Syntax

exportEncryptionKey(jpsConfigFile, keyFilePath, keyFilePassword)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>jpsConfigFile</td>
<td>Specifies the location of the file jps-config.xml relative to the location where the command is run.</td>
</tr>
<tr>
<td>keyFilePath</td>
<td>Specifies the directory where the file ewallet.p12 is created; note that the content of this file is encrypted and secured by the value passed to keyFilePassword.</td>
</tr>
<tr>
<td>keyFilePassword</td>
<td>Specifies the password to secure the file ewallet.p12; note that this same password must be used when importing that file.</td>
</tr>
</tbody>
</table>

4.6.17.3 Example

The following invocation writes the file ewallet.p12 in the directory myDir:

```ruby
eexportEncryptionKey(jpsConfigFile="pathName", keyFilePath="myDir", keyFilePassword="password")
```

4.6.18 importEncryptionKey

Offline command that imports keys from the specified ewallet.p12 file into the domain.

4.6.18.1 Description

Imports encryption keys from the file ewallet.p12 into the domain. The password passed must be the same as that used to create the file with the command exportEncryptionKey.

4.6.18.2 Syntax

importEncryptionKey(jpsConfigFile, keyFilePath, keyFilePassword)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>jpsConfigFile</td>
<td>Specifies the location of the file jps-config.xml relative to the location where the command is run.</td>
</tr>
<tr>
<td>keyFilePath</td>
<td>Specifies the directory where the ewallet.p12 is located.</td>
</tr>
<tr>
<td>keyFilePassword</td>
<td>Specifies the password used when the file ewallet.p12 was generated.</td>
</tr>
</tbody>
</table>
4.6.18.3 Example

importEncryptionKey(jpsConfigFile="pathName", keyFilePath="dirloc", keyFilePassword="password")

4.6.19 restoreEncryptionKey

Offline command to restore the domain credential encryption key.

4.6.19.1 Description

Restores the state of the domain bootstrap keys as it was before running importEncryptionKey.

4.6.19.2 Syntax

restoreEncryptionKey(jpsConfigFile)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>jpsConfigFile</td>
<td>Specifies the location of the file jps-config.xml relative to the location where the command is run.</td>
</tr>
</tbody>
</table>

4.6.19.3 Example

restoreEncryptionKey(jpsConfigFile="pathName")

4.6.20 rollOverEncryptionKey

Offline command to create a new domain credential encryption key.

4.6.20.1 Description

Creates a new credential encryption key in the domain wallet. Note the following important points:

- It should be executed from the administration server in the domain. No server restart is needed after invoking this script.
- If the domain is the only domain accessing the security store, nothing else is required.
- However, if two or more domains share the security store, the newly generated key should be exported from the domain where the script was run and imported into each of the other domains sharing the security store, using the scripts exportEncryptionKey and importEncryptionKey.
- On the WebSphere platform, the script name is Opss.rollOverEncryptionKey.

4.6.20.2 Syntax

rollOverEncryptionKey(jpsConfigFile)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>jpsConfigFile</td>
<td>Specifies the location of the file jps-config.xml relative to the location where the command is run.</td>
</tr>
</tbody>
</table>
4.6.20.3 Example
rollOverEncryptionKey(jpsConfigFile="pathName")

4.6.21 reassociateSecurityStore
Online command that migrates the security store from one repository to another.
For complete details, see section reassociateSecurityStore in Oracle Fusion Middleware Application Security Guide.

4.6.22 upgradeSecurityStore
Offline command that migrates release 10.1.x security data to release 11 security data.
For complete details, see section Upgrading with upgradeSecurityStore in Oracle Fusion Middleware Application Security Guide.

4.6.23 createResourceType
Online command that creates a new resource type in the domain policy store within a given application stripe.

4.6.23.1 Description
Creates a new resource type element in the domain policy store within a given application stripe and with specified name, display name, description, and actions. Optional arguments are enclosed in between square brackets; all other arguments are required. In the event of an error, the command returns a WLSTException.

4.6.23.2 Syntax
Optional arguments are enclosed in square brackets.
createResourceType(appStripe, resourceTypeName, displayName, description [, provider] [, matcher], actions [, delimeter])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe where to insert the resource type.</td>
</tr>
<tr>
<td>resourceTypeName</td>
<td>Specifies the name of the resource type to insert.</td>
</tr>
<tr>
<td>displayName</td>
<td>Specifies the name for the resource type used in UI gadgets.</td>
</tr>
<tr>
<td>description</td>
<td>Specifies a brief description of the resource type.</td>
</tr>
<tr>
<td>provider</td>
<td>Specifies the provider for the resource type.</td>
</tr>
<tr>
<td>matcher</td>
<td>Specifies the class of the resource type. If unspecified, it defaults to oracle.security.jps.ResourcePermission.</td>
</tr>
<tr>
<td>actions</td>
<td>Specifies the actions allowed on instances of the resource type.</td>
</tr>
<tr>
<td>delimeter</td>
<td>Specifies the character used to delimit the list of actions. If unspecified, it defaults to comma ','.</td>
</tr>
</tbody>
</table>

4.6.23.3 Example
The following invocation creates a resource type in the stripe myApplication with actions BWPrint and ColorPrint delimited by a semicolon:

wls:/mydomain/serverConfig> createResourceType(appStripe="myApplication",
4.6.24 getResourceType

Online command that fetches a resource type from the domain policy store within a given application stripe.

4.6.24.1 Description

Gets the relevant parameters of a <resource-type> entry in the domain policy store within a given application stripe and with specified name. In the event of an error, the command returns a WLSTException.

4.6.24.2 Syntax

generateResourceType(appStripe, resourceTypeName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe from where to fetch the resource type.</td>
</tr>
<tr>
<td>resourceTypeName</td>
<td>Specifies the name of the resource type to fetch.</td>
</tr>
</tbody>
</table>

4.6.24.3 Example

The following invocation fetches the resource type myResType from the stripe myApplication:

```
wlst:/mydomain/serverConfig> generateResourceType(appStripe="myApplication", resourceTypeName="myResType")
```

4.6.25 deleteResourceType

Online command that removes a resource type from the domain policy store within a given application stripe.

4.6.25.1 Description

Removes a <resource-type> entry in the domain policy store within a given application stripe and with specified name. In the event of an error, the command returns a WLSTException.

4.6.25.2 Syntax

deleteResourceType(appStripe, resourceTypeName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe from where to remove the resource type.</td>
</tr>
<tr>
<td>resourceTypeName</td>
<td>Specifies the name of the resource type to remove.</td>
</tr>
</tbody>
</table>

4.6.25.3 Example

The following invocation removes the resource type myResType from the stripe myApplication:

```
wlst:/mydomain/serverConfig> deleteResourceType(appStripe="myApplication", resourceTypeName="myResType")
```
4.6.26 listSecurityStoreInfo

Offline command that lists the type, the location, and the administrative user of the domain security store.

4.6.26.1 Description

The script runs in offline mode and outputs the type of the OPSS security store (file, OID, or DB), its location, and the user allowed to access it (typically a security administrator).

4.6.26.2 Syntax

listSecurityStoreInfo(domainConfig="configFilePath")

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>domainConfig</td>
<td>Specifies the full absolute path to the OPSS configuration file jps-config.xml; the file jps-config-jse.xml is also expected to be in the passed directory.</td>
</tr>
</tbody>
</table>

4.6.26.3 Example

The following invocation returns the type, location, and administrative user of the OPSS policy store:

```bash
wls:/mydomain/serverConfig>
listSecurityStoreInfo(domainConfig="/home/myConfigPathDirectory/config/fmwconfig")
```

4.6.27 listAppStripes

Online or offline command that lists the application stripes in the policy store.

4.6.27.1 Description

This script can be run in offline or online mode. When run in offline mode, a configuration file must be passed, and it lists the application stripes in the policy store referred to by the configuration in the default context of the passed configuration file; the default configuration must not have a service instance reference to an identity store. When run in online mode, a configuration file must not be passed, and it lists stripes in the policy store of the domain to which you connect. In any mode, if a regular expression is passed, it lists the application stripes with names that match the regular expression; otherwise, it lists all application stripes.

If this command is used in offline mode after reassociating to a DB-based store, the configuration file produced by the reassociation must be manually edited.

4.6.27.2 Syntax

listAppStripes([configFile="configFileName"] [, regularExpression="aRegExp"])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configFile</td>
<td>Specifies the path to the OPSS configuration file. Optional. If specified, the script runs offline; the default context in the specified configuration file must not have a service instance reference to an identity store. If unspecified, the script runs online and it lists application stripes in the policy store.</td>
</tr>
</tbody>
</table>
4.6.27.3 Examples
The following (online) invocation returns the list of application stripes in the policy store:

```
wlst:/mydomain/serverConfig> listAppStripes
```

The following (offline) invocation returns the list of application stripes in the policy store referenced in the default context of the specified configuration file:

```
wlst:/mydomain/serverConfig> listAppStripes(configFile="/home/myFile/jps-config.xml")
```

The following (online) invocation returns the list of application stripes that contain the prefix App:

```
wlst:/mydomain/serverConfig> listAppStripes(regularExpression="App*")
```

4.6.28 listCodeSourcePermissions

Online command that lists permissions assigned to a source code in global policies.

4.6.28.1 Description
This command allows listing codebase permissions in global policies.

4.6.28.2 Syntax

```
listCodeSourcePermissions([codeBase="codeUrl"])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>codeBaseURL</td>
<td>Specifies the name of the grantee codebase URL</td>
</tr>
</tbody>
</table>

4.6.28.3 Examples

The following invocation returns the list permissions assigned to a code source in all global policies:

```
wlst:/mydomain/serverConfig>
listCodeSourcePermissions(codeBaseURL="file:/tmp/lib/myJars.jar")
```

4.6.29 createResource

Online command that creates a new resource.

4.6.29.1 Description

Creates a resource of a specified type in a specified application stripe. The passed resource type must exist in the passed application stripe.

4.6.29.2 Syntax

```
createResource(appStripe="appStripeName", name="resName", type="resTypeName" [-,displayName="dispName"] [,,-description="descript"])
```
4.6.29.3 Example
The following invocation creates the resource myResource in the stripe myApplication:

```
wlst:/mydomain/serverConfig> createResource(appStripe="myApplication",
    names="myResource", type="myResType", displayName="myNewResource")
```

4.6.30 deleteResource
Online command that deletes a resource.

4.6.30.1 Description
Deletes a resource and all its references from entitlements in an application stripe. It performs a cascading deletion: if the entitlement refers to one resource only, it removes the entitlement; otherwise, it removes from the entitlement the resource actions for the passed type.

4.6.30.2 Syntax
deleteResource(appStripe="appStripeName", name="resName", type="resType Name")

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe where the resource is deleted.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the resource deleted.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of resource deleted. The passed resource type must be present in the application stripe at the time this script is invoked.</td>
</tr>
</tbody>
</table>

4.6.30.3 Example
The following invocation deletes the resource myResource in the stripe myApplication:

```
wlst:/mydomain/serverConfig> deleteResource(appStripe="myApplication",
    names="myResource", type="myResType")
```

4.6.31 listResources
Online command that lists resources in a specified application stripe.

4.6.31.1 Description
If a resource type is specified, it lists all the resources of the specified resource type; otherwise, it lists all the resources of all types.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe where the resource is created.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the resource created.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of resource created. The passed resource type must be present in the application stripe at the time this script is invoked.</td>
</tr>
<tr>
<td>displayName</td>
<td>Specifies the display name of the resource created. Optional.</td>
</tr>
<tr>
<td>description</td>
<td>Specifies the description of the resource created. Optional.</td>
</tr>
</tbody>
</table>
4.6.31.2 Syntax
listResources(appStripe='appStripeName' [,type='resTypeName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe where the resources are listed.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of resource listed. The passed resource type must be present in the application stripe at the time this script is invoked.</td>
</tr>
</tbody>
</table>

4.6.31.3 Example
The following invocation lists all resources of type myResType in the stripe myApplication:

```
wls:/mydomain/serverConfig> listResources(appStripe="myApplication", type="myResType")
```

4.6.32 listResourceActions
Online command that lists the resources and actions in an entitlement.

4.6.32.1 Description
Lists the resources and actions in an entitlement within an application stripe.

4.6.32.2 Syntax
listResourceActions(appStripe="appStripeName", permSetName="entitlementName")

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe where the entitlement resides.</td>
</tr>
<tr>
<td>permSetName</td>
<td>Specifies the name of the entitlement whose resources and actions to list.</td>
</tr>
</tbody>
</table>

4.6.32.3 Example
The following invocation lists the resources and actions of the entitlement myEntitlement in the stripe myApplication:

```
wls:/mydomain/serverConfig> listResourceActions(appStripe="myApplication", permSetName="myEntitlement")
```

4.6.33 createEntitlement
Online command that creates a new entitlement.

4.6.33.1 Description
Creates a new entitlement with just one resource and a list of actions in a specified application stripe. Use addResourceToEntitlement to add additional resources to an existing entitlement; use revokeResourceFromEntitlement to delete resources from an existing entitlement.

4.6.33.2 Syntax
createEntitlement{appStripe='appStripeName', name='entitlementName', resourceName='resName', actions='actionList' [,,-displayName='dispName'] [,,-description='descript']}

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe where the resources are listed.</td>
</tr>
<tr>
<td>resourceName</td>
<td>Specifies the resource for the new entitlement.</td>
</tr>
<tr>
<td>actions</td>
<td>Specifies the list of actions for the new entitlement.</td>
</tr>
<tr>
<td>displayName</td>
<td>Specifies the display name for the new entitlement.</td>
</tr>
<tr>
<td>description</td>
<td>Specifies the description for the new entitlement.</td>
</tr>
</tbody>
</table>
### 4.6.33 createEntitlement

Example

The following invocation creates the entitlement myEntitlement with just the resource myResource in the stripe myApplication:

```
wls:/mydomain/serverConfig> createEntitlement(appStripe="myApplication", name="myEntitlement", resourceName="myResource", actions="read,write")
```

### 4.6.34 getEntitlement

Online command that gets an entitlement.

#### 4.6.34.1 Description

Returns the name, display name, and all the resources (with their actions) of an entitlement in an application stripe.

#### 4.6.34.2 Syntax

```
getEntitlement(appStripe="appStripeName", name="entitlementName")
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe where the entitlement is located.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the entitlement to access.</td>
</tr>
</tbody>
</table>

#### 4.6.34.3 Example

The following invocation returns the information of the entitlement myEntitlement in the stripe myApplication:

```
wls:/mydomain/serverConfig> getEntitlement(appStripe="myApplication", name="myEntitlement")
```

### 4.6.35 deleteEntitlement

Online command that deletes an entitlement.

#### 4.6.35.1 Description

Deletes an entitlement in a specified application stripe. It performs a cascading deletion by removing all references to the specified entitlement in the application stripe.
4.6.35.2 Syntax
deleteEntitlement(appStripe="appStripeName", name="entitlementName")

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe where the entitlement is deleted.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the entitlement to delete.</td>
</tr>
</tbody>
</table>

4.6.35.3 Example
The following invocation deletes the entitlement myEntitlement in the stripe myApplication:

wls:/mydomain/serverConfig> deleteEntitlement(appStripe="myApplication", name="myEntitlement")

4.6.36 addResourceToEntitlement
Online command that adds a resource with specified actions to an entitlement.

4.6.36.1 Description
Adds a resource with specified actions to an entitlement in a specified application stripe. The passed resource type must exist in the passed application stripe.

4.6.36.2 Syntax
addResourceToEntitlement(appStripe="appStripeName", name="entName", resourceName="resName", actions="actionList")

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe where the entitlement is located.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the entitlement to modify.</td>
</tr>
<tr>
<td>resourceName</td>
<td>Specifies the name of the resource to add.</td>
</tr>
<tr>
<td>resourceType</td>
<td>Specifies the type of the resource to add. The passed resource type must be present in the application stripe at the time this script is invoked.</td>
</tr>
<tr>
<td>actions</td>
<td>Specifies the comma-separated list of actions for the added resource.</td>
</tr>
</tbody>
</table>

4.6.36.3 Example
The following invocation adds the resource myResource to the entitlement myEntitlement in the application stripe myApplication:

wls:/mydomain/serverConfig> addResourceToEntitlement(appStripe="myApplication", name="myEntitlement", resourceName="myResource", resourceType="myResType", actions="view,edit")

4.6.37 revokeResourceFromEntitlement
Online command that removes a resource from an entitlement.

4.6.37.1 Description
Removes a resource from an entitlement in a specified application stripe.
4.6.37.2 Syntax

revokeResourceFromEntitlement(appStripe="appStripeName", name="entName", 
resourceName="resName", resourceType="resTypeName", actions="actionList")

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe where the entitlement is located.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the entitlement to modify.</td>
</tr>
<tr>
<td>resourceName</td>
<td>Specifies the name of the resource to remove.</td>
</tr>
<tr>
<td>resourceType</td>
<td>Specifies the type of the resource to remove.</td>
</tr>
<tr>
<td>actions</td>
<td>Specifies the comma-separated list of actions to remove.</td>
</tr>
</tbody>
</table>

4.6.37.3 Example

The following invocation removes the resource myResource from the entitlement myEntitlement in the stripe myApplication:

```bash
wls:/mydomain/serverConfig> 
revokeResourceFromEntitlement(appStripe="myApplication", name="myEntitlement", 
resourceName="myResource", resourceType="myResType", actions="view,edit")
```

4.6.38 listEntitlements

Online command that lists the entitlements in an application stripe.

4.6.38.1 Description

Lists all the entitlements in an application stripe. If a resource name and a resource type are specified, it lists the entitlements that have a resource of the specified type matching the specified resource name; otherwise, it lists all the entitlements in the application stripe.

4.6.38.2 Syntax

```bash
listEntitlements(appStripe="appStripeName" [,resourceTypeName="resTypeName", 
resourceName="resName"])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe from where to list entitlements.</td>
</tr>
<tr>
<td>resourceTypeName</td>
<td>Specifies the name of the type of the resources to list. Optional.</td>
</tr>
<tr>
<td>resourceName</td>
<td>Specifies the name of resource to match. Optional.</td>
</tr>
</tbody>
</table>

4.6.38.3 Examples

The following invocation lists all the entitlements in the stripe myApplication:

```bash
wls:/mydomain/serverConfig> listEntitlements(appStripe="myApplication")
```

The following invocation lists all the entitlements in the stripe myApplication that contain a resource type myResType and a resource whose name match the resource name myResName:

```bash
wls:/mydomain/serverConfig> listEntitlements(appStripe="myApplication", 
resourceTypeName="myResType", resourceName="myResName")
```
4.6.39  
grantEntitlement

Online command that creates a new entitlement.

4.6.39.1  Description

Creates a new entitlement with a specified principal in a specified application stripe.

4.6.39.2  Syntax

grantEntitlement(appStripe="appStripeName", principalClass="principalClass", principalName="principalName", permSetName="entName")

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe where the entitlement is created.</td>
</tr>
<tr>
<td>principalClass</td>
<td>Specifies the class associated with the principal.</td>
</tr>
<tr>
<td>principalName</td>
<td>Specifies the name of the principal to which the entitlement is granted.</td>
</tr>
<tr>
<td>permSetName</td>
<td>Specifies the name of the entitlement created.</td>
</tr>
</tbody>
</table>

4.6.39.3  Example

The following invocation creates the entitlement myEntitlement in the stripe myApplication:

```
wlst:/mydomain/serverConfig> grantEntitlement(appStripe="myApplication", principalClass="oracle.security.jps.service.policystore.ApplicationRole", principalName="myPrincipalName", permSetName="myEntitlement")
```

4.6.40  
revokeEntitlement

Online command that deletes an entitlement.

4.6.40.1  Description

Deletes an entitlement and revokes the entitlement from the principal in a specified application stripe.

4.6.40.2  Syntax

revokeEntitlement(appStripe="appStripeName", principalClass="principalClass", principalName="principalName", permSetName="entName")

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe where the entitlement is deleted.</td>
</tr>
<tr>
<td>principalClass</td>
<td>Specifies the class associated with the principal.</td>
</tr>
<tr>
<td>principalName</td>
<td>Specifies the name of the principal to which the entitlement is revoked.</td>
</tr>
<tr>
<td>permSetName</td>
<td>Specifies the name of the entitlement deleted.</td>
</tr>
</tbody>
</table>

4.6.40.3  Example

The following invocation deleted the entitlement myEntitlement in the stripe myApplication:

```
wlst:/mydomain/serverConfig> revokeEntitlement(appStripe="myApplication", principalClass="oracle.security.jps.service.policystore.ApplicationRole", principalName="myPrincipalName", permSetName="myEntitlement")
```
4.6.41 listResourceTypes
Online command that lists resource types.

4.6.41.1 Description
Lists all the resource types in a specified application stripe.

4.6.41.2 Syntax
listResourceTypes(appStripe="appStripeName")

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appStripe</td>
<td>Specifies the application stripe where the resource types are located.</td>
</tr>
</tbody>
</table>

4.6.41.3 Example
The following invocation lists all resource types in the stripe myApplication:

wls:/mydomain/serverConfig> listResourceTypes(appStripe="myApplication")

4.6.42 updateTrustServiceConfig
Online command that updates the domain trust service configuration values.

4.6.42.1 Description
Updates the trust service configuration with the values passed in a file.

4.6.42.2 Syntax
updateTrustServiceConfig([providerName="the provider name",] propsFile="path to the property file")

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>providerName</td>
<td>Specifies the trust service provider name. Optional. If unspecified, it defaults to trust.provider.embedded.</td>
</tr>
<tr>
<td>propsFile</td>
<td>Specifies the path to the property file. Required.</td>
</tr>
</tbody>
</table>

4.6.42.3 Example
The following invocation updates the trust service store with the values specified in the property file myProps:

wls:/mydomain/serverConfig> updateTrustServiceConfig(propsFile="myProps")

The following is a sample property file:

trust.keystoreType=KSS
trust.keyStoreName=kss://<stripeName>/<keystoreName>
trust.trustStoreName=kss://<stripeName>/<truststoreName>
trust.aliasName=<aliasName>
trust.issuerName=<aliasName>

The type can be KSS or JKS; if a property is set to the empty string, then that property is removed from the trust service configuration.
4.7 Oracle Access Manager Commands

Use the WLST commands listed in Table 4–6 to manage Oracle Access Manager (OAM)-related components, such as authorization providers, identity asserters, and SSO providers, as well as to display metrics and deployment topology, manage Oracle Access Manager server and agent configuration and logger settings.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>listOAMAuthnProviderParams</td>
<td>List the parameters set for an Oracle Access Manager authentication or identity assertion provider.</td>
<td>Online</td>
</tr>
<tr>
<td>createOAMIdentityAsserter</td>
<td>Create a new identity asserter.</td>
<td>Online</td>
</tr>
<tr>
<td>updateOAMIdentityAsserter</td>
<td>Update an existing identity asserter.</td>
<td>Online</td>
</tr>
<tr>
<td>createOAMAuthenticator</td>
<td>Create a new authenticator.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteOAMAuthnProvider</td>
<td>Delete an existing authentication provider.</td>
<td>Online</td>
</tr>
<tr>
<td>updateOAMAuthenticator</td>
<td>Update an existing authenticator.</td>
<td>Online</td>
</tr>
<tr>
<td>addOAMSSOProvider</td>
<td>Add a new SSO provider.</td>
<td>Online</td>
</tr>
<tr>
<td>displayTopology</td>
<td>List the details of deployed Oracle Access Manager Servers.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>displayMetrics</td>
<td>Display the performance metrics of an Oracle Access Manager Server and domain.</td>
<td>Online</td>
</tr>
<tr>
<td>displayOamServer</td>
<td>Display Oracle Access Manager Server configuration details.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>createOamServer</td>
<td>Create an entry for an Oracle Access Manager Server configuration.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>editOamServer</td>
<td>Edit the entry for an Oracle Access Manager Server configuration.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>deleteOamServer</td>
<td>Delete the named Oracle Access Manager Server configuration.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>displayOssoAgent</td>
<td>Display OSSO Agent configuration details.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>editOssoAgent</td>
<td>Edit OSSO Agent configuration details.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>deleteOssoAgent</td>
<td>Delete the named OSSO Agent configuration.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>displayWebgateAgent</td>
<td>Display WebGate Agent configuration details.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>editWebgateAgent</td>
<td>Edit 10g WebGate Agent registration details.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>deleteWebgateAgent</td>
<td>Delete the named 10g WebGate Agent configuration.</td>
<td>Online, Offline</td>
</tr>
</tbody>
</table>
### Table 4–6 (Cont.) WLST Oracle Access Manager Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>changeLoggerSetting</td>
<td>Change Logger Settings.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>changeConfigDataEncryptionKey</td>
<td>Regenerate the configuration data encryption key and re-encrypt data.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>displayUserIdentityStore</td>
<td>Display a user identity store registration.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>editUserIdentityStore</td>
<td>Edit a user identity store registration.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>createUserIdentityStore</td>
<td>Create a user identity store registration.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>deleteUserIdentityStore</td>
<td>Delete a user identity store registration.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>configRequestCacheType</td>
<td>Configure the SSO server request cache type.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>displayRequestCacheType</td>
<td>Display the SSO server request cache type entry.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>exportPolicy</td>
<td>Export Oracle Access Manager policy data from a test (source) to an intermediate Oracle Access Manager file.</td>
<td>Online</td>
</tr>
<tr>
<td>importPolicy</td>
<td>Import Oracle Access Manager policy data from the Oracle Access Manager file specified.</td>
<td>Online</td>
</tr>
<tr>
<td>importPolicyDelta</td>
<td>Import Oracle Access Manager policy changes from the Oracle Access Manager file specified.</td>
<td>Online</td>
</tr>
<tr>
<td>migratePartnersToProd</td>
<td>Migrate partners from the source Oracle Access Manager Server to the specified target Oracle Access Manager Server.</td>
<td>Online</td>
</tr>
<tr>
<td>exportPartners</td>
<td>Export the Oracle Access Manager partners from the source to the intermediate Oracle Access Manager file specified.</td>
<td>Online</td>
</tr>
<tr>
<td>importPartners</td>
<td>Import the Oracle Access Manager partners from the intermediate Oracle Access Manager file specified.</td>
<td>Online</td>
</tr>
<tr>
<td>configureOAAM</td>
<td>Configure the Oracle Access Manager-Oracle Adaptive Access Manager basic integration.</td>
<td>Online</td>
</tr>
<tr>
<td>registerOIFDAPPartner</td>
<td>Register Oracle Identity Federation as Delegated Authentication Protocol (DAP) Partner.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>enableCoexistMode</td>
<td>Enable the Coexist Mode.</td>
<td>Online</td>
</tr>
<tr>
<td>disableCoexistMode</td>
<td>Disable the Coexist Mode.</td>
<td>Online</td>
</tr>
<tr>
<td>editGITOValues</td>
<td>Edit GITO configuration parameters.</td>
<td>Online, Offline</td>
</tr>
<tr>
<td>editWebgate11gAgent</td>
<td>Edit an 11g WebGate registration.</td>
<td>Online, Offline</td>
</tr>
</tbody>
</table>
### 4.7.1 listOAMAuthnProviderParams

Online command that lists the values of the parameters in effect in a domain authenticator or identity asserter.

#### 4.7.1.1 Description

Lists the values of the parameters set for a given Oracle Access Manager authenticator or identity asserter. In the event of an error, the command returns a `WLSTException`.

#### 4.7.1.2 Syntax

```
listOAMAuthnProviderParams(name)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Specifies the name of the authenticator or identity asserter.</td>
</tr>
</tbody>
</table>

#### 4.7.1.3 Example

The following invocation lists the parameters and values set for the asserter named `myIdAsserter`:

```
listOAMAuthnProviderParams(name="myIdAsserter")
```
4.7.2 createOAMIdentityAsserter

Online command that creates an Oracle Access Manager identity asserter in the current domain.

4.7.2.1 Description

Creates an identity asserter with a given name in the current domain. Before executing this command, make sure that no Oracle Access Manager identity asserter is already configured in the current domain. In the event of an error, the command returns a WLSTException.

4.7.2.2 Syntax

createOAMIdentityAsserter(name)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Specifies the name of the new identity asserter. If no name is specified, it defaults to &quot;OAMIdentityAsserter&quot;.</td>
</tr>
</tbody>
</table>

4.7.2.3 Example

The following invocation creates a new identity asserter named OAMIdentityAsserter:

createOAMIdentityAsserter(name="OAMIdentityAsserter")

4.7.3 updateOAMIdentityAsserter

Online command that updates the values of parameters of the Oracle Access Manager identity asserter in the current domain.

4.7.3.1 Description

Updates the value of given parameters of the domain Oracle Access Manager identity asserter. In the event of an error, the command returns a WLSTException.

4.7.3.2 Syntax

updateOAMIdentityAsserter(name, paramNameValueList)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Specifies the name of the Oracle Access Manager identity asserter whose parameter values to update.</td>
</tr>
</tbody>
</table>
4.7.3.3 Example
The following invocation updates the parameters accessGateName, accessGatePwd, pAccessServer, and ssoHeaderName in the Oracle Access Manager identity asseterter named myIdAsserter:

updateOAMIdentityAsserter(name="myIdAsserter",
accessGateName="OAM IAP AccessGate", accessGatePwd="welcome1",
pAccessServer="myhost.domain.com:5543", ssoHeaderName="OAM_SSO_HEADER")

4.7.4 createOAMAuthenticator
Online command that creates an Oracle Access Manager authenticator in the current domain.

4.7.4.1 Description
Creates an Oracle Access Manager authenticator with a given name in the current domain. Before executing this command, make sure that no Oracle Access Manager
authenticator is already configured in the default security domain. In the event of an error, the command returns a \texttt{WLSTException}.

### 4.7.4.2 Syntax
\texttt{createOAMAuthenticator(name)}

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Specifies the name of the new authentication provider in the default domain. If no name is specified, it defaults to &quot;OAMAuthenticator&quot;.</td>
</tr>
</tbody>
</table>

### 4.7.4.3 Example
The following invocation creates a new authentication provider named \texttt{OAMAuthenticator}:

\texttt{createOAMAuthenticator(name="OAMAuthenticator")}

### 4.7.5 deleteOAMAuthnProvider
Online command that deletes the OAM authenticator from the current domain.

#### 4.7.5.1 Description
Deletes the OAM authenticator with a given name from the current domain. In the event of an error, the command returns a \texttt{WLSTException}.

#### 4.7.5.2 Syntax
\texttt{deleteOAMAuthnProvider(name)}

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Specifies the name of the authentication provider to delete.</td>
</tr>
</tbody>
</table>

#### 4.7.5.3 Example
The following invocation deletes the authenticator \texttt{myAuthenticator}:

\texttt{deleteOAMAuthnProvider(name="myAuthenticator")}

### 4.7.6 updateOAMAuthenticator
Online command that updates the values of parameters of the Oracle Access Manager authenticator in the current domain.

#### 4.7.6.1 Description
Updates the value of given parameters of the domain Oracle Access Manager authenticator. In the event of an error, the command returns a \texttt{WLSTException}.

#### 4.7.6.2 Syntax
\texttt{updateOAMAuthenticator(name, paramNameValueList)}

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Specifies the name of the Oracle Access Manager authenticator whose parameter values to update.</td>
</tr>
</tbody>
</table>
Example

The following invocation updates the parameters `accessGateName`, `accessGatePwd`, and `pAccessServer` in the Oracle Access Manager authenticator named `myAuthenticator`:

```java
updateOAMAuthenticator(name="myAuthenticator",
  accessGateName="OAM AP AccessGate", accessGatePwd="welcome1",
  pAccessServer="myhost.domain.com:5543")
```

4.7.7 addOAMSSOPrivate

Online command that adds an Oracle Access Manager SSO provider with the given login URI, logout URI, and auto-login URI.

4.7.7.1 Description

Adds an SSO provider with the given login URI, logout URI, and auto-login URI. This command modifies the domain `jps-config.xml` by adding an Oracle Access Manager...

---

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>paramNameValueList</code></td>
<td>Specifies the comma-separated list of pairs of parameter name-value to be updated. The format of each pair is <code>paramName='paramValue'</code></td>
</tr>
</tbody>
</table>

The only parameter names that can be updated are the following:

- `accessGateName` — The name of the AccessGate used by the authenticator.
- `accessGatePwd` — The password to the AccessGate used by the authenticator.
- `pAccessServer` — The name of the primary access server. Values must have the format `hostName:portNumber`.
- `sAccessServer` — The name of the secondary access server. Values must have the format `hostName:portNumber`.
- `transportSecurity` — The mode of communication between AccessGate and OAM Access Server: open, simple, or cert.
- `keystorePwd` — The password to access the domain key store.
- `keystorePath` — The absolute path of the JKS key store used for SSL communication between the authenticator and OAM Access Server.
- `simpleModePassphrase` — The password shared by AccessGate and OAM Access Server in simple communication mode.
- `truststorePath` — The absolute path of the JKS trust store used for SSL communication between the authenticator and OAM Access Server.
- `poolMaxConnections` — The maximum number of connections in the OAM Server connection pool.
- `poolMinConnections` — The minimum number of connections in the OAM Server connection pool.
- `useRetNameAsPrincipal` — Specifies whether the user name retrieved from the OAM authenticator should be used as the name of the Principal in the Subject.
- `controlFlag` — The JAAS control flag that sets up dependencies among all authenticators in the domain. Values can be only REQUIRED, SUFFICIENT, REQUISITE, or OPTIONAL.
- `appDomain` — The name of the application domain.
SSO service instance with the required properties. In the event of an error, the command returns a WLSTException.

4.7.7.2 Syntax
```
addOAMSSOProvider(loginuri, logouturi, autologinuri, beginimpuri, endimpuri)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>loginuri</td>
<td>Required. Specifies the URI of the login page and triggers SSO authentication.</td>
</tr>
<tr>
<td>logouturi</td>
<td>Optional. Specifies the URI of the logout page and logs the signed-on user out. If unspecified, defaults to logouturi=NONE. Set to &quot;&quot; to ensure that ADF security calls the OPSS logout service, which uses the implementation of the class OAMSOSServiceImpl to clear the cookie ObSSOCookie. More generally, an ADF-secured web application that would like to clear cookies without logging out the user should use this setting.</td>
</tr>
<tr>
<td>autologinuri</td>
<td>Required. Specifies the URI of the autologin page. Optional. If unspecified, it defaults to autologin=NONE.</td>
</tr>
<tr>
<td>beginimpuri</td>
<td>Optional. Specifies the URI that triggers the impersonation SSO session.</td>
</tr>
<tr>
<td>endimpuri</td>
<td>Optional. Specifies the URI that terminates the impersonation SSO session.</td>
</tr>
</tbody>
</table>

4.7.7.3 Example
The following invocation adds an SSO provider with the passed URIs; note the special behavior implied by the setting logouturi="" and the impersonation parameters, as explained in the above table:
```
addOAMSSOProvider(loginuri="/${app.context}/adfAuthentication",
    logouturi="/oamsso/logout.html",
    beginimpuri="https://login.acme.com/impersonationInit.html",
    endimpuri="https://login.acme.com/impersonationTerm.html",
    autologinuri="/fooBar.cgi")
```

4.7.8 displayTopology
Online and offline command that displays the information about all the OAM Servers in a deployment.

4.7.8.1 Description
Lists the topology of deployed OAM Servers. There are no arguments for this command.

4.7.8.2 Syntax
```
displayTopology
```

4.7.8.3 Example
The following invocation lists the details of all deployed OAM Servers, as described above:
```
displayTopology
```
4.7.9 displayMetrics

Online command that displays the performance metrics of an OAM Server and domain.

4.7.9.1 Description
Displays the performance metrics of an OAM Server and domain specific to collectors, including host, process, and server names. There are no arguments for this command.

If none of the arguments are specified all the details of all the servers and collectors are displayed.

4.7.9.2 Syntax
displayMetrics()

4.7.9.3 Example
The following invocation lists all metrics specific to named collectors, as described above:

displayMetrics()

4.7.10 displayOamServer

Online and offline command that displays OAM Server registration details.

4.7.10.1 Description
Displays OAM Server registration details, including the host, port, registration name, OAM Proxy port and server ID, and, optionally, the OAM Proxy shared secret.

The scope of this command is an instance, only. The scope is not an argument.

4.7.10.2 Syntax
displayOamServer(host,port)

### Argument | Definition
--- | ---
host | Mandatory. Specifies the name of the OAM Server host.
port | Mandatory. Specifies the listening port of the OAM Server host.
domainhome | Offline mode: Mandatory
| Online mode: Optional

4.7.10.3 Example
The following invocation lists all metrics specific to named collectors, as described above:

displayOamServer(host="my_host", port="15000", domainHome="domainHome1")

4.7.11 createOamServer

Online and offline command that creates an OAM Server entry in the system configuration.
### 4.7.11.1 Description
Creates an OAM Server registration, including the host, port, registration name, OAM Proxy port and server ID, and, optionally, the OAM Proxy shared secret.

The scope of this command is an instance, only. The scope is not an argument.

### 4.7.11.2 Syntax
```
createOamServer(host, port, paramNameValueList)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>Mandatory. Specifies the name of the OAM Server host.</td>
</tr>
<tr>
<td>port</td>
<td>Mandatory. Specifies the listening port of the OAM Server host.</td>
</tr>
<tr>
<td>domainHome</td>
<td>Offline mode: Mandatory. Online mode: Optional</td>
</tr>
</tbody>
</table>
| paramNameValueList| Specifies the comma-separated list of parameter name-value pairs. The format of each pair is: paramName='paramValue' Mandatory:  
  - configurationProfile—The name of this instance registration, which appears under Server Instances on the System Configuration tab in the OAM Administration Console.  
  - oamProxyPort—The listening port of this instance.  
  - oamProxyServerID—The name of the OAM Proxy for this server instance, which will appear under the OAM Proxy sub tab of the server instance in the OAM Administration Console.  
  - siteName—siteName/serverName for the instance. |

### 4.7.11.3 Example
The following invocation creates a configuration for `your_host` with listening port 15000. The configuration entry in the Administration Console will be `oam_server1`. The OAM Proxy port is 3004 and the OAM Proxy Server ID is `AccessServerConfigProxy`:

```
createOamServer(host="my_host", port="15000", configurationProfile='oam_server1', oamProxyPort="3004", oamProxyServerID="ProxyID", siteName="siteName1", domainHome="domainHome1")
```

### 4.7.12 editOamServer
Online and offline command that enables you to edit OAM Server registration details.

#### 4.7.12.1 Description
Edits the registration for an OAM Server, which can include the host, port, registration name, OAM Proxy port and server ID, and, optionally, the OAM Proxy shared secret.

The scope of this command is an instance, only. The scope is not an argument.

#### 4.7.12.2 Syntax
```
editOamServer(name, port, paramNameValueList)
```
### 4.7.12.3 Example
You can use any of the optional attributes to change current settings. The following invocation enables you to add the OAM Proxy shared secret to the configuration entry `oam_server1`.

```bash
editOamServer(name="oam_server1", port="15000", configurationProfile="oam_server1", oamProxyPort="3004", oamProxyServerID="Proxy1", siteName="siteName1", domainHome="domainHome1")
```

### 4.7.13 deleteOamServer
Online and offline command that enables you to delete the named OAM Server registration.

#### 4.7.13.1 Description
Deletes an entire OAM Server configuration.

The scope of this command is an instance, only. The scope is not an argument.

#### 4.7.13.2 Syntax
`deleteOamServer(host, port)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>Mandatory. Specifies the name of the OAM Server host.</td>
</tr>
<tr>
<td>port</td>
<td>Mandatory. Specifies the listening port of the OAM Server host.</td>
</tr>
</tbody>
</table>
| domainhome | Offline mode: Mandatory  
Online mode: Optional |

#### 4.7.13.3 Example
The following invocation enables you to delete the OAM Server registration for `oam_server1` with listening port 15000.
deleteOamServer(host="oam_server1", port="15000", domainHome="domainHome1")

### 4.7.14 displayOssoAgent

Online and offline command that displays OSSO Agent configuration details.

#### 4.7.14.1 Description

Displays OSSO Agent registration details, which also appear in the OAM Administration Console.

The scope of this command is an instance, only. The scope is not an argument

#### 4.7.14.2 Syntax

displayOssoAgent(agentName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentName</td>
<td>Mandatory. Specifies the name of the OSSO Agent.</td>
</tr>
<tr>
<td>domainhome</td>
<td>Offline mode: Mandatory. Online mode: Optional</td>
</tr>
</tbody>
</table>

#### 4.7.14.3 Example

The following invocation displays the OSSO Agent’s registration information:

displayOssoAgent(agentName="OSSOAgent1", domainHome="domainHome1")

### 4.7.15 editOssoAgent

Online and offline command that enables you to edit an OSSO Agent registration.

#### 4.7.15.1 Description

Changes OSSO Agent configuration details, including the Site Token, Success URL, Failure URL, Home URL, Logout URL, Start Date, End Date, Administrator ID, and Administrator Info.

The scope of this command is an instance, only. The scope is not an argument

#### 4.7.15.2 Syntax

editOssoAgent(agentName, paramNameValueList)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentName</td>
<td>Mandatory. Specifies the name of the OSSO Agent.</td>
</tr>
<tr>
<td>domainhome</td>
<td>Offline mode: Mandatory. Online mode: Optional</td>
</tr>
</tbody>
</table>
### 4.7.15.3 Example

The following invocation changes the Administrator ID and information in the registration entry for `OSSOAgent1`:

```bash
editOssoAgent(agentName="OSSOAgent1", siteToken="siteToken", successUrl="successUrl", failureUrl="failureUrl", homeUrl="homeUrl", logoutUrl="logoutUrl", startDate="2009-12-10", endDate="2012-12-30", adminId=345", adminInfo="Agent11", domainHome="domainHome1")
```

### 4.7.16 deleteOssoAgent

Online and offline command that enables you to delete an OSSO Agent registration.

#### 4.7.16.1 Description

Removes an OSSO Agent configuration.

The scope of this command is an instance, only. The scope is not an argument.

#### 4.7.16.2 Syntax

```bash
deleteOssoAgent(agentName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>agentName</code></td>
<td>Mandatory. Specifies the name of the OSSO Agent.</td>
</tr>
<tr>
<td><code>domainHome</code></td>
<td>Offline mode: Mandatory</td>
</tr>
<tr>
<td></td>
<td>Online mode: Optional</td>
</tr>
</tbody>
</table>

#### 4.7.16.3 Example

The following invocation removes the OSSO Agent named `OSSOAgent1`:

```bash
deleteOssoAgent(agentName="OSSOAgent1", domainHome="domainHome1")
```
4.7.17 displayWebgateAgent

Online and offline command that displays a 10g WebGate registration.

4.7.17.1 Description
Displays all 10g WebGate registration details, which can also be seen in the OAM Administration Console.

The scope of this command is an instance, only. The scope is not an argument.

4.7.17.2 Syntax

displayWebgateAgent(agentName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentName</td>
<td>Mandatory. Specifies the name of the WebGate Agent.</td>
</tr>
<tr>
<td>domainhome</td>
<td>Offline mode: Mandatory</td>
</tr>
<tr>
<td></td>
<td>Online mode: Optional</td>
</tr>
</tbody>
</table>

4.7.17.3 Example

The following invocation displays registration information for my_WebGate:

displayWebgateAgent(agentName="my_Webgate", domainHome="domainHome1")

4.7.18 editWebgateAgent

Online and offline command that enables you to edit a 10g WebGate registration.

4.7.18.1 Description
Enables you to change 10g WebGate Agent registration details.

The scope of this command is an instance, only. The scope is not an argument.

4.7.18.2 Syntax

editWebgateAgent(agentName, paramNameValueList)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentName</td>
<td>Mandatory. Specifies the name of the WebGate Agent.</td>
</tr>
<tr>
<td>domainhome</td>
<td>Offline mode: Mandatory</td>
</tr>
<tr>
<td></td>
<td>Online mode: Optional</td>
</tr>
</tbody>
</table>
Example
You can alter any or all of the settings. Use the following invocation to change specific information in the WebGate Agent registration, including the Agent ID, state, maximum connections, OAM Server timeout, primary cookie domain, cache timeout, cookie session timeout, maximum session timeout, idle session timeout, and failover threshold, as follows:

```
agentId=value
accessClientPassword=value
state={enabled|disabled}
preferredHost=value
aaaTimeOutThreshold=value
security={open|simple|cert}
primaryCookieDomain=value
maxConnections=value
maxCacheElements=value
cacheTimeOut=value
cookieSessionTime=value
maxSessionTime=value
idleSessionTimeout=value
failoverThreshold=value
```

### 4.7.18.3 Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>paramNameValueList</code></td>
<td>Specifies the comma-separated list of parameter name-value pairs to be updated. The format of each pair is: <code>paramName='paramValue'</code></td>
</tr>
<tr>
<td>Mandatory:</td>
<td></td>
</tr>
<tr>
<td>- <code>agentId</code></td>
<td>Name of the OAM Agent (WebGate).</td>
</tr>
<tr>
<td>Optional:</td>
<td></td>
</tr>
<tr>
<td>- <code>accessClientPassword</code></td>
<td>An optional password for this WebGate Agent.</td>
</tr>
<tr>
<td>- <code>state</code></td>
<td>Whether the OAM Agent is enabled or disabled.</td>
</tr>
<tr>
<td>- <code>preferredHost</code></td>
<td>Prevents security holes that can be created if a host's identifier is not included in the Host Identifiers list. For virtual hosting, you must use the Host Identifiers feature.</td>
</tr>
<tr>
<td>- <code>aaaTimeOutThreshold</code></td>
<td>Number (in seconds) to wait for a response from the OAM Run-time Server. If this parameter is set, it is used as an application TCP/IP timeout instead of the default TCP/IP timeout. Default = -1 (default network TCP/IP timeout is used).</td>
</tr>
<tr>
<td>- <code>security</code></td>
<td>Level of transport security to and from the OAM Run-time Server: open, simple, or cert.</td>
</tr>
<tr>
<td>- <code>primaryCookieDomain</code></td>
<td>The Web server domain on which the OAM Agent is deployed, for instance, <code>company.com</code>.</td>
</tr>
<tr>
<td>- <code>maxConnections</code></td>
<td>The maximum number of connections that this OAM Agent can establish with the OAM Server. This number must be the same as (or greater than) the number of connections that are actually associated with this agent. Default = 1.</td>
</tr>
<tr>
<td>- <code>maxCacheElements</code></td>
<td>Number of elements maintained in the cache. Cache elements are URLs or Authentication Schemes. The value of this setting refers to the maximum consolidated count for elements in both of these caches. Default = 10000.</td>
</tr>
<tr>
<td>- <code>cacheTimeOut</code></td>
<td>Amount of time cached information remains in the OAM Agent cache when the information is neither used nor referenced. Default = 1800 (seconds).</td>
</tr>
<tr>
<td>- <code>cookieSessionTime</code></td>
<td>Amount of time that the ObSSOCookie persists. Default = 3600 (seconds)*.</td>
</tr>
<tr>
<td>- <code>maxSessionTime</code></td>
<td>Maximum amount of time, in seconds, that a user's authentication session is valid regardless of their activity. At the expiration of this time, the user is re-challenged for authentication. This is a forced logout. Default = 3600 (seconds). A value of 0 disables this timeout setting.</td>
</tr>
<tr>
<td>- <code>idleSessionTimeout</code></td>
<td>Amount of time in seconds that a user's authentication session remains valid without accessing any OAM Agent protected resources. Default = 3600 (seconds). A value of 0 disables this timeout setting.</td>
</tr>
<tr>
<td>- <code>failoverThreshold</code></td>
<td>Number representing the point when this OAM Agent opens connections to a Secondary OAM Server. Default = 1.</td>
</tr>
</tbody>
</table>
4.7.19 deleteWebgateAgent

Online and offline command that enables you to delete a 10g WebGate Agent registration.

4.7.19.1 Description

Removes an 10g WebGate Agent registration.

The scope of this command is an instance, only. The scope is not an argument

4.7.19.2 Syntax

deleteWebgateAgent(agentName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentName</td>
<td>Mandatory. Specifies the name of the WebGate Agent.</td>
</tr>
<tr>
<td>domainHome</td>
<td>Offline mode: Mandatory</td>
</tr>
<tr>
<td></td>
<td>Online mode: Optional</td>
</tr>
</tbody>
</table>

4.7.19.3 Example

The following invocation removes the WebGate Agent named my_WebGate:

deleteWebgateAgent(agentName="my_WebGate", domainHome="domainHome1")

4.7.20 changeLoggerSetting

Online and offline command that changes the logger level.

4.7.20.1 Description

Changes the level of one or more, or all, loggers.

The scope of this command is an instance, only. The scope is not an argument

4.7.20.2 Syntax

changeLoggerSetting (loggerName='', loggerLevel=''):  

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>loggerName</td>
<td>Optional. Specifies the OAM logger name. Multiple OAM logger names can be specified, separated by commas, or you can use the wildcard (*) character to specify all OAM collectors, which is the default.</td>
</tr>
<tr>
<td>loggerLevel</td>
<td>SEVERE, WARNING, INFO, CONFIG, FINE.</td>
</tr>
</tbody>
</table>

4.7.20.3 Example

The following invocation changes the logger level to SEVERE:

changeLoggerSetting(loggerName= "", loggerLevel=SEVERE)
4.7.21 changeConfigDataEncryptionKey

Offline command that regenerates the configuration data encryption key.

4.7.21.1 Description
Regenerates the configuration data encryption key, re-encrypts the configuration data using the new key, and outputs attribute information of the identity store.

The scope of this command is an instance, only. The scope is not an argument.

4.7.21.2 Syntax
changePasswordEncKey (oldpassword='', newPassword='')

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>oldPassword</td>
<td>Mandatory. Specifies the password that retrieves the current encryption key.</td>
</tr>
<tr>
<td>newPassword</td>
<td>Mandatory. Defines a new password that protects the newly generated encryption key.</td>
</tr>
</tbody>
</table>

4.7.21.3 Example
The following invocation changes the old and new password, regenerates the key, and re-encrypts the configuration data:

changePasswordEncKey(oldpassword="oldpassword", newPassword="newpassword")

4.7.22 displayUserIdentityStore

Online and offline command that displays user identity store registration information.

4.7.22.1 Description
Displays information of the user identity store registered with Oracle Access Manager.

The scope of this command is an instance, only. The scope is not an argument.

4.7.22.2 Syntax
displayUserIdentityStore(name)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Mandatory. Specifies the name of the LDAP user identity store.</td>
</tr>
</tbody>
</table>
| domainHome | Offline mode: Mandatory  
              Online mode: Optional |

4.7.22.3 Example
The following invocation displays registration details of the user identity store:

displayUserIdentityStore(name="ID_store1", domainHome="domainHome1")

4.7.23 editUserIdentityStore

Online and offline command that changes attributes of the user identity store for Oracle Access Manager.
4.7.23.1 Description
Changes one or more attributes of the user identity store registered with Oracle Access Manager.

The scope of this command is an instance, only. The scope is not an argument.

4.7.23.2 Syntax
editUserIdentityStore(name, paramNameValueList)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Mandatory. Specifies the unique name of the LDAP user identity store (only upper and lower case alpha characters and numbers).</td>
</tr>
<tr>
<td>domainhome</td>
<td>Offline mode: Mandatory&lt;br&gt;Online mode: Optional</td>
</tr>
</tbody>
</table>
| paramNameValueList| Specifies the comma-separated list of parameter name-value pairs. The format of each pair is:<br>paramName='paramValue'
Include one or more of the following parameter name-value pairs, in addition to those in createUserIdentityStore, to change the OAM user identity store configuration:
- userFilterObjectClasses—List of user filter object classes (separated by semi-colon).
- groupFilterObjectClasses—List of group filter object classes (separated by semi-colon).
- referralPolicy—LDAP referral policy (either "follow", "ignore" or "throw").
- searchTimeLimit—Time limit in seconds for LDAP Search operation.
- minConnections—Minimum number of connections in the connection pool.
- maxConnections—Maximum number of connections in the connection pool.
- connectionWaitTimeout—Number of seconds to wait for obtaining a connection from the pool.
- connectionRetryCount—Number of attempts to establish a connection to identity store.
- groupNameAttr—Name of the attribute to look up the user groups. For example:<br>ou=people,ou=myrealm,dc=base_domain
- groupCacheEnabled—Toggle (true/false) to enable LDAP group cache.
- groupCacheSize—Number of entries in LDAP group cache.
- groupCacheTTL—Total time to live for each entry of LDAP group cache.

4.7.23.3 Example
The following invocation changes the LDAP URL of the user identity store for OAM:

```
editUserIdentityStore(name="identity_store_name",
LDAP_url="ldap://localhost:7003", domainHome="domaonHome1")
```
4.7.24 createUserIdentityStore

Online and offline command that creates a user identity store registration for Oracle Access Manager.

4.7.24.1 Description

Creates an entry for a new user identity store to be registered with Oracle Access Manager.

The scope of this command is an instance, only. The scope is not an argument.

4.7.24.2 Syntax

createUserIdentityStore(name=, paramNameValueList)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Mandatory. Specifies the unique name of the LDAP user identity store (only upper and lower case alpha characters and numbers).</td>
</tr>
<tr>
<td>domainhome</td>
<td>Offline mode: Mandatory&lt;br&gt;Online mode: Optional</td>
</tr>
</tbody>
</table>
| paramNameValueList| Specifies the comma-separated list of parameter name-value pairs. The format of each pair is: paramName='paramValue'<br>Mandatory:  
  ■ name—The name for this user ID store.  
  ■ principal—The login ID of the LDAP administrator. For example, cn=Admin.  
  ■ credential—The password of the Principal, which is encrypted for security.  
  ■ type—The type of the LDAP ID store to be created.  
  ■ userAttr—User attributes of the store.  
  ■ usersearchbase—The node under which user data is stored in the LDAP ID store to be created. For example: cn=users.  
  ■ groupSearchBase—The node under which group data is stored in the LDAP ID store to be created. Mandatory Attribute. For example: cn=groups.  
  ■ ldapUrl—The URL for the LDAP host, including port number of the LDAP ID store to be created. For example, ldap://localhost:7001.  
  ■ roleSecAdmin—Name of the Admin group with all privileges for LDAP ID store.  
  ■ roleSysMonitor—Name of the Admin group with read-only privileges for LDAP ID store to be created.  
  ■ roleSysManager—Name of the Admin group with day-to-day operational privileges for LDAP ID store to be created.  
  ■ ldapProvider—A supported LDAP provider. For example, OVD.  
  ■ isPrimary—The designation of the primary User Identity Store. Boolean field.  
  ■ userIDProvider—User Identity Provider of the store to be created.  
  ■ domainHome—Domain Home location. |
4.7.24.3 Example
The following invocation creates a new Oracle Internet Directory user identity store definition for use with Oracle Access Manager:

```java
createUserIdentityStore(name="Name1",principal="Principal1", credential="Credential1", type="OID", userAttr="userAttr1", ldapProvider="ldapProvider", roleSecAdmin="roleSecAdmin1", roleSysMonitor="roleSysMonitor", roleSysManager="roleSysManager", roleAppAdmin="roleAppAdmin", userSearchBase="cn=users, ldapUrl="ldapUrl", isPrimary="isPrimary", userIDProvider="userIDProvider", groupSearchBase="cn=groups",domainHome="domainHome1")
```

4.7.25 deleteUserIdentityStore
Online and offline command that removes a Oracle Access Manager user identity store registration.

4.7.25.1 Description
Deletes the user identity store registered with Oracle Access Manager.

The scope of this command is an instance, only. The scope is not an argument.

4.7.25.2 Syntax
`deleteUserIdentityStore(name)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Mandatory. Specifies the name of the LDAP user identity store to be removed.</td>
</tr>
<tr>
<td>domainhome</td>
<td>Offline mode: Mandatory</td>
</tr>
<tr>
<td></td>
<td>Online mode: Optional</td>
</tr>
</tbody>
</table>

4.7.25.3 Example
The following invocation deletes the registration of the user identity store:

```java
deleteUserIdentityStore(name="identity_store", domainHome="domainHome1")
```

4.7.26 configRequestCacheType
Online and offline command that configures the SSO server request cache type.

4.7.26.1 Description
Configures the SSO server request cache type.

The scope of this command is an instance, only. The scope is not an argument.

4.7.26.2 Syntax
`configRequestCacheType(type)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Mandatory. Specifies requestCacheType.</td>
</tr>
<tr>
<td>requestCacheType</td>
<td>The value of request cache type: BASIC or COOKIE.</td>
</tr>
</tbody>
</table>
4.7.26.3 Example
The following invocation identifies the request cache type as Cookie:

`configRequestCacheType(type="COOKIE")`

4.7.27 displayRequestCacheType
Online and offline command that displays the SSO server request cache type.

4.7.27.1 Description
Displays the SSO server request cache type entry.
The scope of this command is an instance, only. The scope is not an argument.

4.7.27.2 Syntax
`displayRequestCacheType(domainHome)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
</table>
| type       | Mandatory. Specifies `requestCacheType`.
|            | `requestCacheType`—The value of request cache type: BASIC or COOKIE. |
| domainHome | Offline mode: Mandatory        |
|            | Online mode: Optional          |

4.7.27.3 Example
The following invocation displays the request cache type.

`displayRequestCacheType(domainHome="domainHome")`

4.7.28 exportPolicy
Online only command that exports OAM policy data from a test (source) environment to the intermediate Oracle Access Manager file specified.

4.7.28.1 Description
Exports OAM policy data from a test (source) environment to the intermediate Oracle Access Manager file.
The scope of this command is an instance, only. The scope is not an argument.

4.7.28.2 Syntax
`exportPolicy(pathTempOAMPolicyFile)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>pathTempOAMPolicyFile</code></td>
<td>Mandatory. Specifies the path to the temporary Oracle Access Manager file.</td>
</tr>
</tbody>
</table>

4.7.28.3 Example
The following invocation specifies the path to the temporary file used when exporting policy data from a test (source) environment.

`exportPolicy(pathTempOAMPolicyFile="oam_policy.xml")`
4.7.29 importPolicy

Online only command that imports the OAM policy data from the intermediate Oracle Access Manager file specified.

4.7.29.1 Description

Imports the OAM policy data from the intermediate Oracle Access Manager file specified.

The scope of this command is an instance, only. The scope is not an argument.

4.7.29.2 Syntax

\[ \text{importPolicy(pathTempOAMPolicyFile)} \]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pathTempOAMPolicyFile</td>
<td>Mandatory. Specifies the path to the temporary OAM file.</td>
</tr>
</tbody>
</table>

4.7.29.3 Example

The following invocation specifies the path to the temporary file used when importing policy data to a production (target).

\[ \text{importPolicy(pathTempOAMPolicyFile="oam_policy.xml")} \]

4.7.30 importPolicyDelta

Online only command that imports the OAM policy changes from the intermediate Oracle Access Manager file specified.

4.7.30.1 Description

Imports the OAM policy changes from the intermediate Oracle Access Manager file specified.

The scope of this command is an instance, only. The scope is not an argument.

4.7.30.2 Syntax

\[ \text{importPolicyDelta(pathTempOAMPolicyFile)} \]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pathTempOAMPolicyFile</td>
<td>Mandatory. Specifies the path to the temporary Oracle Access Manager file.</td>
</tr>
</tbody>
</table>

4.7.30.3 Example

The following invocation specifies the path to the temporary file used when importing only changed policy data to a production (target).

\[ \text{importPolicyDelta(pathTempOAMPolicyFile="oam_policy_delta.xml")} \]

4.7.31 migratePartnersToProd

Online only command that migrates partners from the current (source) OAM Server to the specified (target) OAM Server.
4.7.31.1 Description
Migrates partners from the current (source) OAM Server to the specified (target) OAM Server.

The scope of this command is an instance, only. The scope is not an argument.

4.7.31.2 Syntax
migratePartnersToProd(prodServerHost, prodServerPort, prodServerAdminUser, prodServerAdminPwd)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>prodServerHost</td>
<td>Host name of the target OAM Server to which partners are to be migrated.</td>
</tr>
<tr>
<td>prodServerPort</td>
<td>Port of the target OAM Server to which partners are to be migrated.</td>
</tr>
<tr>
<td>prodServerAdminUser</td>
<td>Administrator of the target OAM Server to which partners are to be migrated.</td>
</tr>
<tr>
<td>prodServerAdminPwd</td>
<td>Target OAM Server administrator's password.</td>
</tr>
</tbody>
</table>

4.7.31.3 Example
The following invocation specifies the required information.

migratePartnersToProd(prodServerHost="host", prodServerPort="port", prodServerAdminUser="weblogic", prodServerAdminPwd="welcome")

4.7.32 exportPartners
Online only command that exports Oracle Access Manager partners from the source to the intermediate Oracle Access Manager file specified.

4.7.32.1 Description
Exports the Oracle Access Manager partners from the source to the intermediate Oracle Access Manager file specified.

The scope of this command is an instance, only. The scope is not an argument.

4.7.32.2 Syntax
exportPartners(pathTempOAMPartnerFile)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pathTempOAMPartnerFile</td>
<td>Mandatory. Specifies the path to the temporary Oracle Access Manager partner file.</td>
</tr>
</tbody>
</table>

4.7.32.3 Example
The following invocation specifies the path to the intermediate OAM partners file.

exportPartners(pathTempOAMPartnerFile='oam_partners.xml')

4.7.33 importPartners
Online only command that imports Oracle Access Manager partners from the intermediate Oracle Access Manager file specified.
4.7.33.1 Description
Imports the OAM partners from the intermediate Oracle Access Manager file specified.
The scope of this command is an instance, only. The scope is not an argument.

4.7.33.2 Syntax

importPartners(pathTempOAMPartnerFile)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pathTempOAMPartnerFile</td>
<td>Mandatory. Specifies the path to the temporary OAM partner file.</td>
</tr>
</tbody>
</table>

4.7.33.3 Example

The following invocation specifies the path to the intermediate OAM partners file.

importPartners(pathTempOAMPartnerFile="oam_partners.xml")

4.7.34 configureOAAM

Online only command that configures the Oracle Access Manager-Oracle Adaptive Access Manager basic integration.

4.7.34.1 Description

Configures the OAM-OAAM basic integration.
The scope of this command is an instance, only. The scope is not an argument.

4.7.34.2 Syntax

configureOAAM(dataSourceName,paramNameValueList)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>dataSourceName</td>
<td>Name of the data source to be created</td>
</tr>
<tr>
<td>paramNameValueList</td>
<td>Specifies the comma-separated list of parameter name-value pairs. The format of each pair is: paramName='paramValue' Mandatory: ■ hostName—The name of the database host. ■ port—Database port. ■ sid—The database sid (database identifier). ■ userName—OAAM schema name. ■ password—OAAM schema password. Optional: ■ maxConnectionSize—Max connection reserve time out size. ■ maxPoolSize—Maximum size for connection pool. ■ serverName—Target server for the data source.</td>
</tr>
</tbody>
</table>

4.7.34.3 Example

The following invocation configures the Oracle Access Manager-Oracle Adaptive Access Manager basic integration.
configureOAAM(dataSourceName = "MyOAAMDS", hostName = "host.us.co.com", port = "1521", sid = "sid", userName = "username", passWord = "password", maxConnectionSize = None, maxPoolSize = None, serverName = "oam_server1")

4.7.35 registerOIFDAPPartner

Online and offline command that registers Oracle Identity Federation as a Delegated Authentication Protocol (DAP) Partner.

4.7.35.1 Description

Registers Oracle Identity Federation as Delegated Authentication Protocol (DAP) Partner.

The scope of this command is an instance only. The scope is not an argument.

4.7.35.2 Syntax

registerOIFDAPPartner()

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramNameValueList</td>
<td>Specifies the comma-separated list of parameter name-value pairs. The format of each pair is: paramName='paramValue' Mandatory: Include the following parameter name-value pairs to create a new OAM user identity store configuration: ■ keystoreLocation—Location of the Keystore file (generated at the OIF Server.). ■ logoutURL—The OIF Server's logout URL. Optional: ■ rolloverInterval—The Rollover Interval for the keys used to encrypt/decrypt SASSO Tokens.</td>
</tr>
</tbody>
</table>

4.7.35.3 Example

The following invocation illustrates use of all parameters.


4.7.36 enableCoexistMode

Online command that enables the Coexist Mode.

4.7.36.1 Description

Enables the Coexist Mode.

The scope of this command is an instance, only. The scope is not an argument.

4.7.36.2 Syntax

enableCoexistMode()
4.7.36.3 Example
The following invocation enables the Coexist Mode.

`enableCoexistMode`

4.7.37 disableCoexistMode
Online command that disables the Coexist Mode.

4.7.37.1 Description
Disables the Coexist Mode.
The scope of this command is an instance, only. The scope is not an argument.

4.7.37.2 Syntax
disableCoexistMode()

4.7.37.3 Example
The following invocation enables the Coexist Mode.
disableCoexistMode

4.7.38 editGITOValues
Online and offline command that edits GITO configuration parameters.

4.7.38.1 Description
Edits GITO configuration parameters.
The scope of this command is an instance, only. The scope is not an argument.

4.7.38.2 Syntax
eeditGITOValues{gitoEnabled, paramNameValueList}

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>gitoEnabled</td>
<td>True (or false). Allows (or denies) user to set GITO enabled property.</td>
</tr>
<tr>
<td>domainhome</td>
<td>Offline mode: Mandatory&lt;br&gt;Online mode: Optional</td>
</tr>
</tbody>
</table>
Example

The following invocation edits GITO configuration parameters.

```
editGITOValues(gitoEnabled="true",gitoCookieDomain=".abc.com",gitoCookieName="ABC",gitoVersion="v1.0",gitoTimeout="20",gitoSecureCookieEnabled="false",domainHome="/abc/def/ijk")
```

4.7.39 editWebgate11gAgent

Online and offline command that edits an 11g WebGate registration.

4.7.39.1 Description

Edits an 11g WebGate registration.

The scope of this command is an instance, only. The scope is not an argument.

4.7.39.2 Syntax

```
editWebgate11gAgent(agentname, paramNameValueList)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramNameValueList</td>
<td>Specifies the comma-separated list of parameter name-value pairs.</td>
</tr>
<tr>
<td></td>
<td>The format of each pair is:</td>
</tr>
<tr>
<td></td>
<td>paramName='paramValue'</td>
</tr>
<tr>
<td></td>
<td>Mandatory:</td>
</tr>
<tr>
<td></td>
<td>Include the following parameter name-value pairs to create a new</td>
</tr>
<tr>
<td></td>
<td>OAM user identity store configuration:</td>
</tr>
<tr>
<td></td>
<td>■ gitoCookieDomain—Allows user to set the GITO cookie domain entry.</td>
</tr>
<tr>
<td></td>
<td>Optional:</td>
</tr>
<tr>
<td></td>
<td>■ gitoCookieName—Allows user to set the GITO cookie name.</td>
</tr>
<tr>
<td></td>
<td>■ gitoVersion—Allows user to set the GITO version. Can be ONLY v1.0 or v3.0.</td>
</tr>
<tr>
<td></td>
<td>■ gitoTimeout—Allows user to set the GITO timeout value.</td>
</tr>
<tr>
<td></td>
<td>■ gitoSecureCookieEnabled—True (or false). Allows (or denies) user to set the GITO cookie enabled property.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentname</td>
<td>Name of the registered OAM 11g WebGate agent to be edited.</td>
</tr>
<tr>
<td>domainHome</td>
<td>Offline mode: Mandatory  Online mode: Optional</td>
</tr>
</tbody>
</table>
4.7.39.3 Example

The following invocation lists all mandatory and optional parameters.

```bash
```

4.7.40 deleteWebgate11gAgent

Online and offline command that enables you to delete an 11g WebGate Agent registration.
4.7.40.1 Description
Removes an 11g WebGate Agent registration.

The scope of this command is an instance, only. The scope is not an argument.

4.7.40.2 Syntax
deleteWebgate11gAgent(agentName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentName</td>
<td>Mandatory. Specifies the name of the 11g WebGate Agent.</td>
</tr>
<tr>
<td>domainhome</td>
<td>Offline mode: Mandatory Online mode: Optional</td>
</tr>
</tbody>
</table>

4.7.40.3 Example
The following invocation removes the 11g WebGate Agent named my_11gWebGate:
deleWebgate11gAgent(agentName="my_11gWebGate", domainHome="domainHome1")

4.7.41 displayWebgate11gAgent
Online and offline command that enables you to display an 11g WebGate Agent registration.

4.7.41.1 Description
Displays an 11g WebGate Agent registration.

The scope of this command is an instance, only. The scope is not an argument.

4.7.41.2 Syntax
displayWebgate11gAgent(agentName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>agentName</td>
<td>Mandatory. Specifies the name of the WebGate Agent.</td>
</tr>
<tr>
<td>domainhome</td>
<td>Offline mode: Mandatory Online mode: Optional</td>
</tr>
</tbody>
</table>

4.7.41.3 Example
The following invocation displays the WebGate Agent named my_11gWebGate:
displayWebgate11gAgent(agentName="my_11gWebGate", domainHome="domainHome1")

4.7.42 displayOAMMetrics
Online and offline command that enables the display of metrics of OAM Servers.

4.7.42.1 Description
Enables the display of metrics of OAM Servers.

The scope of this command is an instance, only. The scope is not an argument.
4.7.42 Syntax

displayOAMMetrics(domainHome)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>domainhome</td>
<td>Offline mode: Mandatory</td>
</tr>
<tr>
<td></td>
<td>Online mode: Optional</td>
</tr>
</tbody>
</table>

4.7.42.3 Example

The following invocation enables the display of metrics of OAM Servers.

displayOAMMetrics (domainHome=(domainHome1))

4.7.43 updateOIMHostPort

Online only command that updates the Oracle Identity Manager configuration when integrated with Oracle Access Manager.

4.7.43.1 Description

Updates the Oracle Identity manager configuration in system configuration.

The scope of this command is an instance, only. The scope is not an argument.

4.7.43.2 Syntax

updateOIMHostPort(hostname, port, secureProtocol)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>hostname</td>
<td>Name of the Oracle Identity Manager host.</td>
</tr>
<tr>
<td>port</td>
<td>Port of the Oracle Identity Manager host.</td>
</tr>
<tr>
<td>secureProtocol</td>
<td>True or false.</td>
</tr>
</tbody>
</table>

4.7.43.3 Example

The following invocation illustrates this command.

updateOIMHostPort(hostName="OIM host", port="7777", secureProtocol="true")

4.7.44 configureOIM

Online only command that creates an agent registration specific to Oracle Identity Manager when integrated with Oracle Access Manager.

4.7.44.1 Description

Creates an Agent registration specific to Oracle Identity Manager when integrated with Oracle Access Manager.

The scope of this command is an instance, only. The scope is not an argument.

4.7.44.2 Syntax

updateOIMHostPort(hostname, port, secureProtocol)
### Example

The following invocation illustrates this command.

```sql
updateOIMHostPort(hostName="OIM host", port="7777", secureProtocol="true")
configureOIM(oimHost="OIM host", oimPort="7777", oimSecureProtocolEnabled="true",
oimAccessGatePwd = "Access Gate Password", oimCookieDomain = "OIM Cookie Domain",
oimWgId="OIM Webgate ID", oimWgVersion="OIM Webgate Version")
```

#### 4.7.45 updateOSSOResponseCookieConfig

Online and offline command that updates OSSO Proxy response cookie settings.

### Description

Updates OSSO Proxy response cookie settings.

The scope of this command is an instance, only. The scope is not an argument.

### Syntax

`updateOSSOResponseCookieConfig()`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>domainhome</code></td>
<td>Offline mode: Mandatory Online mode: Optional</td>
</tr>
<tr>
<td><code>cookieName</code></td>
<td>Optional. Name of the cookie for which settings are updated. If not specified, the global setting is updated.</td>
</tr>
<tr>
<td><code>cookieMaxAge</code></td>
<td>Maximum age of a cookie in minutes. A negative value sets a session cookie.</td>
</tr>
<tr>
<td><code>isSecureCookie</code></td>
<td>Boolean flag specifies if cookie should be secure (sent only over SSL channel).</td>
</tr>
<tr>
<td><code>cookieDomain</code></td>
<td>The domain of the cookie.</td>
</tr>
</tbody>
</table>

### Example

The following invocation illustrates this command.

```sql
updateOSSOResponseCookieConfig(cookieName = "<cookieName>",
cookieMaxAge = "<cookie age in minutes>", isSecureCookie = "true | false",
cookieDomain="<domain of the cookie>", domainHome = "<wls_domain_home_path>")
```
4.7.46  deleteOSSOResponseCookieConfig

Online and offline command that deletes OSSO Proxy response cookie settings.

4.7.46.1 Description
Deletes OSSO Proxy response cookie settings.

The scope of this command is an instance, only. The scope is not an argument.

4.7.46.2 Syntax
deleteOSSOResponseCookieConfig()

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>domainhome</td>
<td>Offline mode: Mandatory</td>
</tr>
<tr>
<td></td>
<td>Online mode: Optional</td>
</tr>
<tr>
<td>cookieName</td>
<td>Mandatory. Name of the cookie for which settings are deleted.</td>
</tr>
</tbody>
</table>

4.7.46.3 Example
The following invocation illustrates this command.

deleteOSSOResponseCookieConfig(cookieName = "<cookieName>",
cookieDomain="<domain of the cookie>", domainHome = "<wls_domain_home_path>")

4.7.47  displaySimpleModeGlobalPassphrase

Displays the simple mode global passphrase in plain text from the system configuration.

4.7.47.1 Description
Online only command that displays the simple mode global passphrase in plain text from the system configuration.

4.7.47.2 Syntax
displaySimpleModeGlobalPassphrase

There are no arguments for this command.

4.7.47.3 Example
The following invocation illustrates this command.
displaySimpleModeGlobalPassphrase

4.7.48  exportSelectedPartners
Exports selected OAM Partners.

4.7.48.1 Description
Exports selected OAM Partners to the intermediate OAM file specified.
### 4.7.48 exportSelectedPartners

**Syntax**

```
exportSelectedPartners
```

**Argument**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pathTempOAMPartnerFile</td>
<td>The temporary file containing partners to be migrated.</td>
</tr>
<tr>
<td>partnersNameList</td>
<td>comma separated list of partner ids to be migrated</td>
</tr>
</tbody>
</table>

**Example**

The following invocation illustrates this command.

```
exportSelectedPartners (pathTempOAMPartnerFile="/exampleroot/parent/tempfile.extn"
partnersNameList="partner1,partner2"
```

### 4.7.49 migrateArtifacts

**Description**

Migrates artifacts based on the input artifact file.

**Syntax**

```
migrateArtifacts
```

**Argument**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>Location of the artifacts file is present</td>
</tr>
<tr>
<td>password</td>
<td>Password used while generating original artifacts.</td>
</tr>
<tr>
<td>type</td>
<td>InPlace or OutOfPlace</td>
</tr>
<tr>
<td>isIncremental</td>
<td>true or false. If true, an incremental upgrade is done.</td>
</tr>
</tbody>
</table>

**Example**

The following invocation illustrates this command.

```
migrateArtifacts(path="/exampleroot/parent/t", password="password", type="InPlace", isIncremental="false")
```

### 4.7.50 registerThirdPartyTAPPartner

**Description**

Registers any third party as a Trusted Authentication Protocol (TAP) Partner.

**Syntax**

```
registerThirdPartyTAPPartner
```

**Argument**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>Location of the artifacts file is present</td>
</tr>
<tr>
<td>password</td>
<td>Password used while generating original artifacts.</td>
</tr>
</tbody>
</table>

**Example**

```
registerThirdPartyTAPPartner(path="/exampleroot/parent/t", password="password")
```
4.7.50.3 Example

```java
```

4.8 Oracle Security Token Service

Table 4-7 describes the various types of WLST commands available for the Oracle Security Token Service.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerName</td>
<td>Name of partner. Can be any name used for identifying the third party partner.</td>
</tr>
<tr>
<td>keystoreLocation</td>
<td>The jceks file location.</td>
</tr>
<tr>
<td>password</td>
<td>password</td>
</tr>
<tr>
<td>tapScheme</td>
<td>Trusted Authentication Protocol Authn Scheme (TAPScheme, out of the box.)</td>
</tr>
<tr>
<td>tapRedirectUrl</td>
<td>Third party access URL.</td>
</tr>
</tbody>
</table>

### Table 4–7 WLST Oracle Security Token Service Command Groups

<table>
<thead>
<tr>
<th>OSTS Command Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partner Commands</td>
<td>WLST commands related to tasks involving partners.</td>
</tr>
<tr>
<td>Relying Party Partner Mapping Commands</td>
<td>The WS Prefix to Relying Party Partner mappings are used to map a service URL, specified in the AppliesTo field of a WS-Trust RST request, to a partner of type Relying Party. The WS prefix string can be an exact service URL, or a URL with a parent path to the service URL. For example, if a mapping is defined to map a WS Prefix (<a href="http://test.com/service">http://test.com/service</a>) to a Relying Party (RelyingPartyPartnerTest), then the following service URLs would be mapped to the Relying Party: <a href="http://test.com/service">http://test.com/service</a>, <a href="http://test.com/service/calculatorService">http://test.com/service/calculatorService</a>, <a href="http://test.com/service/shop/cart">http://test.com/service/shop/cart</a>...</td>
</tr>
<tr>
<td>Partner Profiles Commands</td>
<td>WLST commands related to tasks involving partner profiles.</td>
</tr>
<tr>
<td>Issuance Templates Commands</td>
<td>WLST commands related to tasks involving issuance templates.</td>
</tr>
<tr>
<td>Validation Templates Commands</td>
<td>WLST commands related to tasks involving validation templates.</td>
</tr>
</tbody>
</table>

Use the WLST commands listed in Table 4–8 to manage Oracle Security Token Service.

### Table 4–8 WLST Commands Oracle Security Token Service

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partner Commands</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getPartner</code></td>
<td>Retrieve a partner and print result.</td>
<td>Online</td>
</tr>
<tr>
<td><code>getAllRequesterPartners</code></td>
<td>Retrieve the names of Requester partners.</td>
<td>Online</td>
</tr>
<tr>
<td><code>getAllRelyingPartyPartners</code></td>
<td>Retrieve the names of all Relying Party partners.</td>
<td>Online</td>
</tr>
<tr>
<td><code>getAllIssuingAuthorityPartners</code></td>
<td>Retrieve the names of all Issuing Authority partners.</td>
<td>Online</td>
</tr>
</tbody>
</table>
Table 4–8 (Cont.) WLST Commands Oracle Security Token Service

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>isPartnerPresent</td>
<td>Query OSTS to determine whether or not the partner exists in the Partner store.</td>
<td>Online</td>
</tr>
<tr>
<td>createPartner</td>
<td>Create a new Partner entry.</td>
<td>Online</td>
</tr>
<tr>
<td>updatePartner</td>
<td>Update an existing Partner entry based on the provided information.</td>
<td>Online</td>
</tr>
<tr>
<td>deletePartner</td>
<td>Delete a partner entry.</td>
<td>Online</td>
</tr>
<tr>
<td>getPartnerUsernameTokenUsername</td>
<td>Retrieve the partner’s username value.</td>
<td>Online</td>
</tr>
<tr>
<td>getPartnerUsernameTokenPassword</td>
<td>Retrieve the partner’s password value.</td>
<td>Online</td>
</tr>
<tr>
<td>setPartnerUsernameTokenCredential</td>
<td>Set the username and password values of a partner entry.</td>
<td>Online</td>
</tr>
<tr>
<td>deletePartnerUsernameTokenCredential</td>
<td>Remove the username and password values from a partner entry.</td>
<td>Online</td>
</tr>
<tr>
<td>getPartnerSigningCert</td>
<td>Retrieve the Base64 encoded signing certificate for the partner.</td>
<td>Online</td>
</tr>
<tr>
<td>getPartnerEncryptionCert</td>
<td>Retrieve the Base64 encoded encryption certificate for the partner.</td>
<td>Online</td>
</tr>
<tr>
<td>setPartnerSigningCert</td>
<td>Upload the signing certificate to the partner entry.</td>
<td>Online</td>
</tr>
<tr>
<td>setPartnerEncryptionCert</td>
<td>Upload the encryption certificate to the partner entry.</td>
<td>Online</td>
</tr>
<tr>
<td>deletePartnerSigningCert</td>
<td>Remove the signing certificate from the partner entry.</td>
<td>Online</td>
</tr>
<tr>
<td>deletePartnerEncryptionCert</td>
<td>Remove the encryption certificate from the partner entry.</td>
<td>Offline</td>
</tr>
<tr>
<td>getPartnerAllIdentityAttributes</td>
<td>Retrieve and display all Identity mapping attributes used to map a token to a requester partner.</td>
<td>Online</td>
</tr>
<tr>
<td>getPartnerIdentityAttribute</td>
<td>Retrieve and display the identity mapping attribute.</td>
<td>Offline</td>
</tr>
<tr>
<td>setPartnerIdentityAttribute</td>
<td>Set the identity mapping attribute for a requester partner.</td>
<td>Offline</td>
</tr>
<tr>
<td>deletePartnerIdentityAttribute</td>
<td>Delete the identity mapping attribute for a requester partner.</td>
<td>Offline</td>
</tr>
</tbody>
</table>

Relying Party Partner Mapping Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>getAllWSPrefixAndPartnerMappings</td>
<td>Retrieve and display all WS Prefixes.</td>
<td>Online</td>
</tr>
<tr>
<td>getWSPrefixAndPartnerMapping</td>
<td>Retrieve and display the Relying Party Partner mapped to the specified wsprefix parameter.</td>
<td>Online</td>
</tr>
<tr>
<td>createWSPrefixAndPartnerMapping</td>
<td>Create a new WS Prefix mapping to a Relying Partner.</td>
<td>Online</td>
</tr>
</tbody>
</table>
### Table 4–8 (Cont.) WLST Commands Oracle Security Token Service

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>deleteWSPrefixAndPartnerMapping</td>
<td>Delete an existing WS Prefix mapping to a Relying Partner.</td>
<td>Online</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Offline</td>
</tr>
</tbody>
</table>

**Partner Profiles Commands**

| getPartnerProfile                          | Retrieve partner profile configuration data.                         | Online           |
|                                            |                                                                       | Offline          |
| createRequesterPartnerProfile              | Create a new Requester Partner profile with default configuration data.| Online           |
| createRelyingPartyPartnerProfile           | Create a new Relying Party Partner profile with default configuration data. | Online           |
| createIssuingAuthorityPartnerProfile       | Create a new Issuing Authority Partner profile with default configuration data. | Online           |
| deletePartnerProfile                      | Delete an existing partner profile.                                  | Online           |

**Issuance Template Commands**

| getAllIssuanceTemplates                    | Retrieve the names of all the existing Issuance Templates.          | Online           |
|                                            |                                                                       | Offline          |
| getIssuanceTemplate                        | Retrieve configuration data of a specific Issuance Template.        | Online           |
|                                            |                                                                       | Offline          |
| createIssuanceTemplate                     | Create a new Issuance Template with default configuration data.      | Online           |
| deleteIssuanceTemplate                     | Delete an existing Issuance Template.                               | Online           |
|                                            |                                                                       | Offline          |

**Validation Template Commands**

| getAllValidationTemplates                  | Retrieve the names of all the existing Validation Templates.        | Online           |
|                                            |                                                                       | Offline          |
| getValidationTemplate                      | Retrieve configuration data of a specific Validation Template.       | Online           |
|                                            |                                                                       | Offline          |
| createWSSValidationTemplate                | Create a new WS Security Validation Template with default configuration data. | Online           |
| createWSTrustValidationTemplate            | Create a new WS Trust Validation Template with default configuration data. | Online           |
| deleteValidationTemplate                   | Delete an existing Issuance Template.                               | Online           |
|                                            |                                                                       | Offline          |

### 4.8.1 getPartner

Online command that retrieves the Partner entry and prints out the configuration for this partner.

#### 4.8.1.1 Description

Retrieves the Partner entry and prints out the configuration for this partner.
4.8.1.2 Syntax
getPartner(partnerId)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the partnerId: the ID of the partner.</td>
</tr>
</tbody>
</table>

4.8.1.3 Example
The following invocation retrieves the Partner entry and prints out the configuration for customPartner:
getPartner(partnerId="customPartner")

4.8.2 getAllRequesterPartners
Online command that retrieves Requester type partners.

4.8.2.1 Description
Retrieves Requester type partners.

4.8.2.2 Syntax
getAllRequesterPartners()

4.8.2.3 Example
The following invocation retrieves Requester type partners:
getAllRequesterPartners()

4.8.3 getAllRelyingPartyPartners
Online command that retrieves Relying Party partners.

4.8.3.1 Description
Retrieves the Relying Party partners.

4.8.3.2 Syntax
getAllRelyingPartyPartners()

4.8.3.3 Example
The following invocation retrieves Relying Party partners:
getAllRelyingPartyPartners()

4.8.4 getAllIssuingAuthorityPartners
Online command that retrieves Issuing Authority partners and prints out the result.

4.8.4.1 Description
Retrieves the Issuing Authority partners and prints out the result.

4.8.4.2 Syntax
getAllIssuingAuthorityPartners()
4.8.4.3 Example
The following invocation retrieves Issuing Authority partners and prints out the result:

getAllIssuingAuthorityPartners()  

4.8.5 isPartnerPresent
Online command that queries OSTS to determine whether or not the specified partner exists in the Partner store.

4.8.5.1 Description
Queries OSTS to determine whether or not the specified partner exists in the Partner store, and prints out the result.

4.8.5.2 Syntax
isPartnerPresent(partnerId)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the partner.</td>
</tr>
</tbody>
</table>

4.8.5.3 Example
The following invocation queries OSTS to determine whether or not customPartner exists in the Partner store, and prints out the result:

isPartnerPresent(partnerId="customPartner")

4.8.6 createPartner
Online command that creates a new Partner entry.

4.8.6.1 Description
Creates a new Partner entry based on provided information. Displays a message indicating the result of the operation.

4.8.6.2 Syntax
createPartner(partnerId, partnerType, partnerProfileId, description, bIsTrusted)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the new partner to be created.</td>
</tr>
<tr>
<td>partnerType</td>
<td>Specifies the type of partner. Values can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>■ STS_REQUESTER for Requester</td>
</tr>
<tr>
<td></td>
<td>■ STS_RELYING_PARTY for Relying Party</td>
</tr>
<tr>
<td></td>
<td>■ STS_ISSUING_AUTHORITY for Issuing Authority</td>
</tr>
<tr>
<td>partnerProfileId</td>
<td>Specifies the profile ID to be attached to this partner. It must reference</td>
</tr>
<tr>
<td></td>
<td>an existing partner profile, and the type of the partner profile must be</td>
</tr>
<tr>
<td></td>
<td>compliant with the type of the new partner entry.</td>
</tr>
<tr>
<td>description</td>
<td>Specifies the optional description of this new partner entry.</td>
</tr>
</tbody>
</table>
4.8.6.3 Example
The following invocation creates STS_Requestor partner, customPartner, custom-partnerprofile with a description (custom requester), with a trust value of true, displays a message indicating the result of the operation:

```
cREATEPARTNER(partnerId="customPartner", partnerType="STS_REQUESTER", partnerProfileId="custom-partnerprofile", description="custom requester", bIsTrusted="true")
```

4.8.7 updatePartner
Online command that updates an existing Partner entry.

4.8.7.1 Description
Updates an existing Partner entry based on the provided information. Displays a message indicating the result of the operation.

4.8.7.2 Syntax
```
UPDATEPARTNER(partnerId, partnerProfileId, description, bIsTrusted)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the new partner to be updated.</td>
</tr>
<tr>
<td>partnerProfileId</td>
<td>Specifies the partner profile ID. It must reference an existing partner profile, and the type of the partner profile must be compliant with the type of the new partner entry.</td>
</tr>
<tr>
<td>description</td>
<td>Specifies the optional description of this new partner entry.</td>
</tr>
<tr>
<td>bIsTrusted</td>
<td>A value that indicates whether or not this new partner is trusted. Value can be either:</td>
</tr>
<tr>
<td></td>
<td>• true for trusted</td>
</tr>
<tr>
<td></td>
<td>• false if not trusted</td>
</tr>
</tbody>
</table>

4.8.7.3 Example
The following invocation updates customPartner with a new profile ID, (x509-wss-validtemp), description (custom requester with new profile id), and a trust value of false. A message indicates the result of the operation:

```
UPDATEPARTNER(partnerId="customPartner", partnerProfileId="x509-wss-validtemp", description="custom requester with new profile id", bIsTrusted=false)
```

4.8.8 deletePartner
Online command that deletes a partner entry from OSTS.
4.8.8.1 Description
Deletes an existing Partner entry referenced by the partnerId parameter from OSTS, and prints out the result of the operation.

4.8.8.2 Syntax
\[
deletePartner(partnerId)
\]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the partner to be deleted.</td>
</tr>
</tbody>
</table>

4.8.8.3 Example
The following invocation deletes the customPartner partner entry referenced by the partnerId parameter from OSTS, and prints out the result of the operation:
\[
deletePartner(partnerId="customPartner")
\]

4.8.9 getPartnerUsernameTokenUsername
Online command that retrieves a partner’s username value that will be used for UNT credentials partner validation or mapping operation.

4.8.9.1 Description
Retrieves a partner’s username value that will be used for UNT credentials partner validation or mapping operation, and displays the value.

4.8.9.2 Syntax
\[
getPartnerUsernameTokenUsername(partnerId)
\]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the partner.</td>
</tr>
</tbody>
</table>

4.8.9.3 Example
The following invocation retrieves the customPartner partner username value that will be used for UNT credentials partner validation or mapping operation, and displays the value:
\[
getPartnerUsernameTokenUsername(partnerId="customPartner")
\]

4.8.10 getPartnerUsernameTokenPassword
Online command that retrieves a partner’s password value that will be used for UNT credentials partner validation or mapping operation.

4.8.10.1 Description
Retrieves a partner password value that will be used for UNT credentials partner validation or mapping operation, and displays the value.

4.8.10.2 Syntax
\[
getPartnerUsernameTokenPassword(partnerId)
\]
4.8.10.3 Example
The following invocation retrieves customPartner partner password value that will be used for UNT credentials partner validation or mapping operation, and displays the value:

```
getPartnerUsernameTokenPassword(partnerId="customPartner")
```

4.8.11 setPartnerUsernameTokenCredential
Online command that sets the username and password values of a partner entry, that will be used for UNT credentials partner validation or mapping operation.

4.8.11.1 Description
Sets the username and password values of a partner entry, that will be used for UNT credentials partner validation or mapping operation. Displays the result of the operation.

4.8.11.2 Syntax
```
setPartnerUsernameTokenCredential(partnerId, UTUsername, UTPassword)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the partner.</td>
</tr>
<tr>
<td>UTUsername</td>
<td>Specifies the username value used for UNT credentials validation or mapping operations.</td>
</tr>
<tr>
<td>UTPassword</td>
<td>Specifies the username value used for UNT credentials validation or mapping operations.</td>
</tr>
</tbody>
</table>

4.8.11.3 Example
The following invocation sets the username and password values of the customPartner partner entry, and displays the result of the operation:

```
setPartnerUsernameTokenCredential(partnerId="customPartner", UTUsername="test", UTPassword="password")
```

4.8.12 deletePartnerUsernameTokenCredential
Online command that removes the username and password values from a partner entry that are used for UNT credentials partner validation or mapping operation, and displays the result of the operation.

4.8.12.1 Description
Removes the username and password values from a partner entry that are used for UNT credentials partner validation or mapping operation, and displays the result of the operation.

4.8.12.2 Syntax
```
deletePartnerUsernameTokenCredential(partnerId)
```
4.8.12.3 Example
The following invocation removes the username and password values from a partner entry that are used for UNT credentials partner validation or mapping operation, and displays the result of the operation:

deletePartnerUsernameTokenCredential(partnerId="customPartner")

4.8.13 getPartnerSigningCert
Online command that retrieves the Base64 encoded signing certificate for the partner referenced by the partnerId parameter, and displays its value, as a Base64 encoded string.

4.8.13.1 Description
Retrieves the Base64 encoded signing certificate for the partner referenced by the partnerId parameter, and displays its value, as a Base64 encoded string.

4.8.13.2 Syntax
getPartnerSigningCert(partnerId)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the partner.</td>
</tr>
</tbody>
</table>

4.8.13.3 Example
The following invocation retrieves Base64 encoded signing certificate for the partner referenced by the partnerId parameter, and displays its value, as a Base64 encoded string:

getPartnerSigningCert(partnerId="customPartner")

4.8.14 getPartnerEncryptionCert
Online command that retrieves the Base64 encoded encryption certificate, and displays its value as a Base64 encoded string.

4.8.14.1 Description
Retrieves the Base64 encoded encryption certificate for the partner referenced by the partnerId parameter, and displays its value as a Base64 encoded string.

4.8.14.2 Syntax
getPartnerEncryptionCert(partnerId)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the partner.</td>
</tr>
</tbody>
</table>
4.8.14.3 Example
The following invocation retrieves the Base64 encoded encryption certificate for the partner referenced by the partnerId parameter, and displays its value, as a Base64 encoded string:

```
getPartnerEncryptionCert(partnerId="customPartner")
```

4.8.15 setPartnerSigningCert

Online command that Uploads the provided certificate to the partner entry as the signing certificate. Displays the result of the operation.

4.8.15.1 Description
Uploads the provided certificate to the partner entry (referenced by the partnerId parameter) as the signing certificate. The supported formats of the certificate are DER and PEM. Displays the result of the operation.

4.8.15.2 Syntax
```
setPartnerSigningCert(partnerId, certFile)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the partner.</td>
</tr>
<tr>
<td>certFile</td>
<td>Specifies the location of the certificate on the local filesystem.</td>
</tr>
<tr>
<td></td>
<td>Supported formats of the certificate are DER and PEM.</td>
</tr>
</tbody>
</table>

4.8.15.3 Example
The following invocation uploads the provided certificate to the partner entry customPartner as the signing certificate. Displays the result of the operation:

```
setPartnerSigningCert(partnerId="customPartner", certFile="/temp/signing_cert")
```

4.8.16 setPartnerEncryptionCert

Online command that Uploads the provided certificate to the partner entry as the encryption certificate. Displays the result of the operation.

4.8.16.1 Description
Uploads the provided certificate to the partner entry (referenced by the partnerId parameter) as the encryption certificate. Displays the result of the operation.

4.8.16.2 Syntax
```
setPartnerEncryptionCert(partnerId, certFile)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the partner.</td>
</tr>
<tr>
<td>certFile</td>
<td>Specifies the location of the certificate on the local filesystem.</td>
</tr>
<tr>
<td></td>
<td>Supported formats of the certificate are DER and PEM.</td>
</tr>
</tbody>
</table>

4.8.16.3 Example
The following invocation uploads the provided certificate to the partner entry customPartner as the signing certificate. Displays the result of the operation:
4.8.17 deletePartnerSigningCert
Online command that removes the encryption certificate from the partner entry and displays the result of the operation.

4.8.17.1 Description
Removes the encryption certificate from the partner entry, referenced by the partnerId parameter, and displays the result of the operation.

4.8.17.2 Syntax
deletePartnerSigningCert(partnerId)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the partner.</td>
</tr>
</tbody>
</table>

4.8.17.3 Example
The following invocation removes the encryption certificate from the partner entry, customPartner, and displays the result of the operation:

```
deletePartnerSigningCert(partnerId="customPartner")
```

4.8.18 deletePartnerEncryptionCert
Online command that removes the signing certificate from the partner entry and displays the result of the operation.

4.8.18.1 Description
Removes the signing certificate from the partner entry, referenced by the partnerId parameter, and displays the result of the operation.

4.8.18.2 Syntax
deletePartnerEncryptionCert(partnerId)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the partner.</td>
</tr>
</tbody>
</table>

4.8.18.3 Example
The following invocation removes the signing certificate from the partner entry, customPartner, and displays the result of the operation:

```
deletePartnerEncryptionCert(partnerId="customPartner")
```

4.8.19 getPartnerAllIdentityAttributes
Online command that retrieves and displays all the identity mapping attributes used to map a token to a requester partner, or to map binding data (SSL Client certificate or HTTP Basic Username) to a requester partner.
4.8.19.1 Description
Retrieves and displays all the identity mapping attributes used to map a token to a requester partner, or to map binding data (SSL Client certificate or HTTP Basic Username) to a requester partner.

The identity mapping attributes only exist for partners of type Requester.

4.8.19.2 Syntax
getPartnerAllIdentityAttributes(partnerId)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the Requester partner. Identity mapping attributes only exist for partners of type Requester</td>
</tr>
</tbody>
</table>

4.8.19.3 Example
The following invocation retrieves and displays all the identity mapping attributes used to map a token to a requester partner, or to map binding data (SSL Client certificate or HTTP Basic Username) to a requester partner: customPartner.

getPartnerAllIdentityAttributes(partnerId="customPartner")

4.8.20 getPartnerIdentityAttribute
Online command that retrieves and displays identity mapping attributes used to map a token or to map binding data to a requester partner.

4.8.20.1 Description
Retrieves and displays an identity mapping attribute used to map a token to a requester partner, or to map binding data (SSL Client certificate or HTTP Basic Username) to a requester partner.

The identity mapping attributes only exist for partners of type Requester.

4.8.20.2 Syntax
getPartnerIdentityAttribute(partnerId, identityAttributeName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the Requester partner.</td>
</tr>
<tr>
<td>IdentityAttributeName</td>
<td>Specifies the name of the identity mapping attribute to retrieve and display. For example: httpbasicusername.</td>
</tr>
</tbody>
</table>

4.8.20.3 Example
The following invocation retrieves and displays one identityAttribute and its value as specified by identityAttributeName.

getPartnerIdentityAttribute(partnerId="customPartner", identityAttributeName="httpbasicusername")

4.8.21 setPartnerIdentityAttribute
Online command that sets the identity mapping attribute for the Requester partner.
4.8.21.1 Description
Set the identity mapping attribute specified by `identityAttributeName` for the partner of type requester specified by the `partnerId` parameter. These identity mapping attributes only exist for Requester partners. Displays the result of the operation.

4.8.21.2 Syntax

```java
setPartnerIdentityAttribute(partnerId, identityAttributeName, identityAttributeValue)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>partnerId</code></td>
<td>Specifies the ID of the partner of type Requester.</td>
</tr>
<tr>
<td><code>identityAttributeName</code></td>
<td>Specifies the name of the identity mapping attribute to retrieve and display.</td>
</tr>
<tr>
<td><code>identityAttributeValue</code></td>
<td>Specifies the value of the identity mapping attribute to set.</td>
</tr>
</tbody>
</table>

4.8.21.3 Example
The following invocation sets the identity mapping attribute specified by `identityAttributeName` for the Requester partner of type requester specified by the `partnerId` parameter. Displays the result of the operation.

```java
setPartnerIdentityAttribute(partnerId="customPartner", identityAttributeName="httpbasicusername", identityAttributeValue="test")
```

4.8.22 `deletePartnerIdentityAttribute`

Online command that deletes the identity mapping attribute.

4.8.22.1 Description
Deletes the identity mapping attribute specified by `identityAttributeName`. The identity mapping attributes used to map a token to a requester partner, or to map binding data (SSL Client certificate or HTTP Basic Username) to a requester partner, and they only exist for Requester partners.

4.8.22.2 Syntax

```java
deletePartnerIdentityAttribute(partnerId, identityAttributeName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>partnerId</code></td>
<td>Specifies the ID of the partner.</td>
</tr>
<tr>
<td><code>identityAttributeName</code></td>
<td>Specifies the name of the identity mapping attribute to delete.</td>
</tr>
</tbody>
</table>

4.8.22.3 Example
The following invocation deletes the identity mapping attribute specified by `identityAttributeName` for Requester partner `customPartner`.

```java
deletePartnerIdentityAttribute(partnerId="customPartner", identityAttributeName="httpbasicusername")
```
4.8.23  getAllWSPrefixAndPartnerMappings
Online command that retrieves and displays all WS Prefixes to Relying Party Partner mappings.

4.8.23.1  Description
Retrieves and displays all WS Prefixes to Relying Party Partner mappings.

4.8.23.2  Syntax
getAllWSPrefixAndPartnerMappings()

4.8.23.3  Example
The following invocation retrieves and displays the WS Prefixes.
getAllWSPrefixAndPartnerMappings()

4.8.24  getWSPrefixAndPartnerMapping
Online command that retrieves and displays the Relying Party Partner mapped to the specified wsprefix parameter, if a mapping for that WS Prefix exists.

4.8.24.1  Description
Retrieves and displays the Relying Party Partner mapped to the specified wsprefix parameter, if a mapping for that WS Prefix exists.

4.8.24.2  Syntax
getWSPrefixAndPartnerMapping(wsprefix)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>wsprefix</td>
<td>Specifies the WS Prefix entry to retrieve and display. The path is optional. If specified, it should take the following form: http_protocol://hostname_ip/path</td>
</tr>
</tbody>
</table>

4.8.24.3  Example
The following invocation retrieves nd displays the Relying Party Partner mapped to the specified wsprefix parameter, if a mapping for that WS Prefix exists.
getWSPrefixAndPartnerMapping(wsprefix="http://host1.example.com/path")

4.8.25  createWSPrefixAndPartnerMapping
Online command that creates a new WS Prefix mapping to a Relying Partner.

4.8.25.1  Description
Creates a new WS Prefix mapping to a Relying Partner referenced by the partnerid parameter, and displays the result of the operation.

4.8.25.2  Syntax
createWSPrefixAndPartnerMapping(wsprefix, partnerid, description)
4.8.25.3 Example
The following invocation creates a new WS Prefix mapping to a Relying Partner Partner referenced by the partnerid parameter, and displays the result of the operation.

createWSPrefixAndPartnerMapping(wsprefix="http://host1.example.com/path", partnerid="customRPpartner", description="some description")

4.8.26 deleteWSPrefixAndPartnerMapping
Online command that deletes an existing mapping of WS Prefix to a Relying Partner Partner.

4.8.26.1 Description
Deletes an existing mapping of WS Prefix to a Relying Partner, and displays the result of the operation.

4.8.26.2 Syntax
deleteWSPrefixAndPartnerMapping(wsprefix)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>wsprefix</td>
<td>Specifies the WS Prefix entry to retrieve and display. The path is optional.</td>
</tr>
<tr>
<td></td>
<td>If specified, it should take the following form:</td>
</tr>
<tr>
<td></td>
<td>http://hostname_ip/path</td>
</tr>
<tr>
<td>partnerId</td>
<td>Specifies the ID of the partner.</td>
</tr>
<tr>
<td>description</td>
<td>Specifies an optional description.</td>
</tr>
</tbody>
</table>

4.8.26.3 Example
The following invocation deletes the existing mapping of WS Prefix to a Relying Partner, and displays the result of the operation.

deleteWSPrefixAndPartnerMapping(wsprefix="http://host1.example.com/path")

4.8.27 getAllPartnerProfiles
Online command that retrieves the names of all the existing partner profiles and displays them.

4.8.27.1 Description
Retrieves the names of all the existing partner profiles and displays them.

4.8.27.2 Syntax
getAllPartnerProfiles()
4.8.27.3 Example
The following invocation retrieves the names of all the existing partner profiles and displays them.

getAllPartnerProfiles()

4.8.28 getPartnerProfile
Online command that retrieves the configuration data of a specific partner profile, and displays the content of the profile.

4.8.28.1 Description
Retrieves the configuration data of the partner profile referenced by the partnerProfileId parameter, and displays the content of the profile.

4.8.28.2 Syntax
getPartnerProfile(partnerProfileId)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerProfileId</td>
<td>Specifies the name of the partner profile.</td>
</tr>
</tbody>
</table>

4.8.28.3 Example
The following invocation retrieves the configuration data of the partner profile referenced by the partnerProfileId parameter, and displays the content of the profile.

getPartnerProfile(partnerProfileId="custom-partnerprofile")

4.8.29 createRequesterPartnerProfile
Online command that creates a new requester partner profile with default configuration data.

4.8.29.1 Description
Creates a new requester partner profile with default configuration data, and displays the result of the operation.

Table 4-9 describes the default configuration created with this command.

Table 4–9 Default Configuration: createRequesterPartnerProfile
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Error for Missing Claims</td>
<td>Default: false</td>
</tr>
<tr>
<td>Allow Unmapped Claims</td>
<td>Default: false</td>
</tr>
</tbody>
</table>
4.8.29.2 Syntax

createRequesterPartnerProfile(partnerProfileId, defaultRelyingPartyPPID, description)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerProfileId</td>
<td>Specifies the name of the partner profile.</td>
</tr>
<tr>
<td>defaultRelyingPartyPPID</td>
<td>Specifies the relying party partner profile to use, if the AppliesTo field is missing from the RST or if it could not be mapped to a Relying Party Partner.</td>
</tr>
<tr>
<td>description</td>
<td>Specifies the optional description for this partner profile</td>
</tr>
</tbody>
</table>

4.8.29.3 Example

The following invocation creates a new requester partner profile with default configuration data, and displays the result of the operation. For default data descriptions, see Table 4–9.

createRequesterPartnerProfile(partnerProfileId="custom-partnerprofile", defaultRelyingPartyPPID="rpPartnerProfileTest", description="custom partner profile")
4.8.30 createRelyingPartyPartnerProfile

Online command that creates a new relying party partner profile with default configuration data.

4.8.30.1 Description

Creates a new relying party partner profile with default configuration data, and displays the result of the operation.

Table 4–10 describes the default configuration created with this command.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Download Policy</td>
<td>Default: false</td>
</tr>
<tr>
<td>Allow Unmapped Claims</td>
<td>Default: false</td>
</tr>
<tr>
<td>Token Type Configuration</td>
<td>The Token Type Configuration will contain a single entry, with:</td>
</tr>
<tr>
<td></td>
<td>■ The token type set to the type of Issuance Template referenced by defaultIssuanceTemplateID</td>
</tr>
<tr>
<td></td>
<td>■ The Issuance template set to defaultIssuanceTemplateID</td>
</tr>
<tr>
<td><strong>Note</strong>: For the token type of the issuance template referenced by defaultIssuanceTemplateID, it will be linked to the issuance template, while the other token types will not be linked to any issuance template. If the issuance template referenced by defaultIssuanceTemplateID is of custom token type, the table will only contain one entry, with the custom token type, mapped to the custom token type as the external URI, and mapped to the issuance template referenced by defaultIssuanceTemplateID</td>
<td></td>
</tr>
<tr>
<td>Attribute Name Mapping</td>
<td>The Attribute Name Mapping table is empty by default.</td>
</tr>
</tbody>
</table>

4.8.30.2 Syntax

createRelyingPartyPartnerProfile(partnerProfileId, defaultIssuanceTemplateID, description)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerProfileId</td>
<td>Specifies the name of the partner profile.</td>
</tr>
<tr>
<td>defaultIssuanceTemplateID</td>
<td>Specifies the default issuance template and token type to issue if no token type was specified in the RST.</td>
</tr>
<tr>
<td>description</td>
<td>Specifies the optional description for this partner profile</td>
</tr>
</tbody>
</table>

4.8.30.3 Example

The following invocation creates a new relying party partner profile with default configuration data, and displays the result of the operation.

createRelyingPartyPartnerProfile(partnerProfileId="custom-partnerprofile", defaultIssuanceTemplateID="saml11-issuance-template", description="custom partner profile")
4.8.31 createIssuingAuthorityPartnerProfile

Online command that creates a new issuing authority partner profile with default configuration data.

4.8.31.1 Description

Creates a new issuing authority partner profile with the default configuration data in Table 4–11, and displays the result of the operation.

Table 4–11 Default Configuration: createIssuingAuthorityPartnerProfile

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server Clockdrift</td>
<td>Default: 600 seconds</td>
</tr>
<tr>
<td>Token Mapping</td>
<td>The Token Mapping Section will be configured as follows:</td>
</tr>
<tr>
<td></td>
<td>- Override Simple User Mapping: false</td>
</tr>
<tr>
<td></td>
<td>- Override User NameID Mapping: false</td>
</tr>
<tr>
<td></td>
<td>- Override Attribute Based User Mapping: false</td>
</tr>
<tr>
<td></td>
<td>- Override Simple Partner Mapping: false</td>
</tr>
<tr>
<td></td>
<td>- Override Partner NameID Mapping: false</td>
</tr>
<tr>
<td></td>
<td>Empty fields</td>
</tr>
<tr>
<td></td>
<td>- simple user mapping</td>
</tr>
<tr>
<td></td>
<td>- attribute based user mapping</td>
</tr>
<tr>
<td></td>
<td>- simple partner mapping</td>
</tr>
<tr>
<td>Partner NameID Mapping</td>
<td>The Partner NameID Mapping table will be provisioned with the following entries as NameID format. However, without any data in the datastore column the issuance template referenced by defaultIssuanceTemplateID is of token type SAML 1.1, SAML 2.0, or Username. The table will contain the following entries:</td>
</tr>
<tr>
<td></td>
<td>- urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomain QualifiedName</td>
</tr>
<tr>
<td></td>
<td>- urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName</td>
</tr>
<tr>
<td></td>
<td>- urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</td>
</tr>
<tr>
<td></td>
<td>- urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified</td>
</tr>
<tr>
<td></td>
<td>- urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos</td>
</tr>
<tr>
<td></td>
<td>- urn:oasis:names:tc:SAML:2.0:nameid-format:persistent</td>
</tr>
<tr>
<td>User NameID Mapping</td>
<td>The User NameID Mapping table will be provisioned with the following entries as NameID format:</td>
</tr>
<tr>
<td></td>
<td>- urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomain QualifiedName, empty datastore column</td>
</tr>
<tr>
<td></td>
<td>- urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName, dn set in the datastore column</td>
</tr>
<tr>
<td></td>
<td>- urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress, mail set in the datastore column</td>
</tr>
<tr>
<td></td>
<td>- urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified, empty datastore column</td>
</tr>
<tr>
<td></td>
<td>- urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos, empty datastore column</td>
</tr>
<tr>
<td></td>
<td>- urn:oasis:names:tc:SAML:2.0:nameid-format:persistent, empty datastore column</td>
</tr>
</tbody>
</table>
4.8.31.2 Syntax
createIssuingAuthorityPartnerProfile(partnerProfileId, description)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerProfileId</td>
<td>Specifies the name of the partner profile.</td>
</tr>
<tr>
<td>description</td>
<td>Specifies the optional description for this partner profile</td>
</tr>
</tbody>
</table>

4.8.31.3 Example
The following invocation a new issuing authority partner profile with default configuration data, and displays the result of the operation.
createIssuingAuthorityPartnerProfile(partnerProfileId="custom-partnerprofile" description="custom partner profile")

4.8.32 deletePartnerProfile
Online command that deletes an partner profile referenced by the partnerProfileId parameter.

4.8.32.1 Description
Deletes an partner profile referenced by the partnerProfileId parameter, and displays the result of the operation.

4.8.32.2 Syntax
deletePartnerProfile(partnerProfileId)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partnerProfileId</td>
<td>Specifies the name of the partner profile to be removed.</td>
</tr>
</tbody>
</table>

4.8.32.3 Example
The following invocation deletes an partner profile referenced by the partnerProfileId parameter, and displays the result of the operation.
deletePartnerProfile(partnerProfileId="custom-partnerprofile")

4.8.33 getAllIssuanceTemplates
Online command that retrieves the names of all the existing issuance templates.

4.8.33.1 Description
Retrieves the names of all the existing issuance templates and displays them.

4.8.33.2 Syntax
getAllIssuanceTemplates
4.8.33.3 Example
The following invocation retrieves the names of all the existing issuance templates and displays them.

getAllIssuanceTemplates

4.8.34 getIssuanceTemplate
Online command that retrieves the configuration data of a specific issuance template.

4.8.34.1 Description
Retrieves the configuration data of the issuance template referenced by the issuanceTemplateId parameter, and displays the content of the template.

4.8.34.2 Syntax
getIssuanceTemplate(issuanceTemplateId)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>issuanceTemplateId</td>
<td>Specifies the name of the issuance template.</td>
</tr>
</tbody>
</table>

4.8.34.3 Example
The following invocation retrieves the configuration data of the issuance template referenced by the issuanceTemplateId parameter, and displays the content of the template.

getIssuanceTemplate(issuanceTemplateId="custom-issuancetemp")

4.8.35 createIssuanceTemplate
Online command that creates a new issuance template with default configuration data.

4.8.35.1 Description
Creates a new issuance template with default configuration data, and displays the result of the operation.

Table 4–12 describes the default configuration for this command.

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username</td>
<td>The issuance template will be created with the following default values:</td>
</tr>
<tr>
<td></td>
<td>■ Send Encrypted Token: false</td>
</tr>
<tr>
<td></td>
<td>■ NameID User Attribute: uid</td>
</tr>
<tr>
<td></td>
<td>■ NameID User Attribute Store: User Store</td>
</tr>
<tr>
<td></td>
<td>■ Password Attribute: (empty)</td>
</tr>
<tr>
<td></td>
<td>■ Include Nonce: true</td>
</tr>
<tr>
<td></td>
<td>■ Include Timestamp: true</td>
</tr>
</tbody>
</table>

Table 4–12  Default Configuration: createIssuanceTemplate
### Syntax

```java
createIssuanceTemplate(issuanceTemplateId, tokenType, signingKeyId, description)
```

### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>issuanceTemplateId</td>
<td>Specifies the name of the issuance template to be created.</td>
</tr>
<tr>
<td>tokenType</td>
<td>Possible values can be:</td>
</tr>
<tr>
<td></td>
<td>■ username: indicates that the token type is UsernameToken</td>
</tr>
<tr>
<td></td>
<td>■ saml11: indicates that the token type is a SAML 1.1 Assertion</td>
</tr>
<tr>
<td></td>
<td>■ saml20: indicates that the token type is a SAML 2.0 Assertion</td>
</tr>
<tr>
<td></td>
<td>■ &lt;other&gt;: in this case, the token type is assumed to be a custom</td>
</tr>
<tr>
<td></td>
<td>token type, referenced by &lt;other&gt; (replace &lt;other&gt; by a value)</td>
</tr>
<tr>
<td>signingKeyId</td>
<td>Specifies the keyID referencing the key entry (defined in the STS</td>
</tr>
<tr>
<td></td>
<td>General Settings UI section) that will be used to sign outgoing SAML</td>
</tr>
<tr>
<td></td>
<td>Assertions. Only required when token type is saml11 or saml20.</td>
</tr>
<tr>
<td>description</td>
<td>An optional description.</td>
</tr>
</tbody>
</table>

### Example

The following invocation creates a new issuance template with default configuration data, and displays the result of the operation.

```java
createIssuanceTemplate(issuanceTemplateId="custom-issuancetemp",
```
4.8.36 deleteIssuanceTemplate

Online command that deletes an issuance template referenced by the issuanceTemplateId parameter, and displays the result of the operation.

4.8.36.1 Description
Deletes an issuance template referenced by the issuanceTemplateId parameter, and displays the result of the operation.

4.8.36.2 Syntax
deleteIssuanceTemplate(issuanceTemplateId)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>issuanceTemplateId</td>
<td>Specifies the name of the existing issuance template to be removed.</td>
</tr>
</tbody>
</table>

4.8.36.3 Example
The following invocation deletes an issuance template referenced by the issuanceTemplateId parameter, and displays the result of the operation.

deleteIssuanceTemplate(issuanceTemplateId="custom-issuancetemp")

4.8.37 getAllValidationTemplates

Online command that retrieves the names of all the existing validation templates.

4.8.37.1 Description
Retrieves the names of all the existing validation templates and displays them.

4.8.37.2 Syntax
getAllValidationTemplates()

4.8.37.3 Example
The following invocation retrieves the names of all the existing validation templates and displays them.
getAllValidationTemplates()

4.8.38 getValidationTemplate

Online command that retrieves the configuration data of a specific validation template, and displays the content of the template.

4.8.38.1 Description
Retrieves the configuration data of the validation template referenced by the validationTemplateId parameter, and displays the content of the template.

4.8.38.2 Syntax
getValidationTemplate(validationTemplateId)
4.8.38.3 Example
The following invocation retrieves the configuration data of a specific validation template, and displays the content of the template.
```
getValidationTemplate(validationTemplateId="custom-wss-validtemp")
```

4.8.39 createWSSValidationTemplate
Online command that creates a new validation template with default configuration data.

4.8.39.1 Description
Creates a new validation template with default configuration data, and displays the result of the operation.

The WSS validation template is created with the values in Table 4–13, depending on the token type.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>validationTemplateId</td>
<td>Specifies the name of the existing validation template.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>validationTemplateId</td>
<td>Specifies the name of the existing validation template.</td>
</tr>
</tbody>
</table>

Table 4–13 Default Configuration: createWSSValidationTemplate

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username</td>
<td>The validation template will be created with the following default values:</td>
</tr>
<tr>
<td></td>
<td>■ Timestamp Lifespan: 600 seconds</td>
</tr>
<tr>
<td></td>
<td>■ Enable Credential Validation: true</td>
</tr>
<tr>
<td></td>
<td>■ Validation Source: Partner</td>
</tr>
<tr>
<td></td>
<td>■ Token Mapping: Map token to Partner</td>
</tr>
<tr>
<td></td>
<td>■ Enable Simple Partner Mapping: true</td>
</tr>
<tr>
<td></td>
<td>■ Partner Datastore Attribute: username</td>
</tr>
</tbody>
</table>

SAML 1.1
or
SAML 2.0

The validation template will be created with the following default values:

- Authentication Timeout: 3600 seconds
- Timestamp Lifespan: 3600 seconds

The Token Mapping section will be created with the following default values:

- Map token: Map token to Partner
- Enable Simple User Mapping: false
- Enable User NameID Mapping: false
- Enable Attribute Based User Mapping: false
- Enable Simple Partner Mapping: false
- Enable Partner NameID Mapping: false

Empty fields: User Token Attribute, User Datastore Attribute and Attribute Based User Mapping

Also:

- Partner Token Attribute: NameID
- Partner Datastore Attribute: username

Partner NameID Mapping table will be provisioned with the following entries as NameID format, but without any data in the datastore column:

- `urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualified Name`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress`
- `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`

User NameID Mapping table will be provisioned with the following entries as NameID format:

- `urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualified Name`, empty datastore column
- `urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName`, dn set in the datastore column
- `urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress`, mail set in the datastore column
- `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`, empty datastore column
- `urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos`, empty datastore column
- `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`, empty datastore column

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAML 1.1</td>
<td>The validation template will be created with the following default values:</td>
</tr>
<tr>
<td>or</td>
<td>■ Authentication Timeout: 3600 seconds</td>
</tr>
<tr>
<td>or</td>
<td>■ Timestamp Lifespan: 3600 seconds</td>
</tr>
<tr>
<td>SAML 2.0</td>
<td>The validation template will be created with the following default values:</td>
</tr>
</tbody>
</table>

Table 4–13 (Cont.) Default Configuration: createWSSValidationTemplate
Table 4–13 (Cont.) Default Configuration: createWSSValidationTemplate

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| X.509      | The Token Mapping section will be created with the following default values:  
|            | • Map token: Map token to Partner  
|            | • Enable Simple User Mapping: false  
|            | • Enable Attribute Based User Mapping: false  
|            | • Enable Simple Partner Mapping: true  
|            | Empty fields: User Token Attribute, User Datastore Attribute and Attribute Based User Mapping  
|            | Also:  
|            | • Partner Token Attribute: DN  
|            | • Partner Datastore Attribute: sslclientcertdn |
| Kerberos   | The Token Mapping section will be created with the following default values:  
|            | • Map token: Map token to User  
|            | • Enable Simple User Mapping: true  
|            | • Enable Attribute Based User Mapping: false  
|            | • Enable Simple Partner Mapping: false  
|            | Empty fields: Partner Token Attribute, Partner Datastore Attribute and Attribute Based User Mapping  
|            | Also:  
|            | • User Token Attribute: TPE_KERBEROS_PRINCIPAL_FULL  
|            | • User Datastore Attribute: mail |

4.8.39.2 Syntax

createWSSValidationTemplate(templateId, tokenType, defaultRequesterPPID, description)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>templateId</td>
<td>Specifies the name of the name of the validation template to be created.</td>
</tr>
</tbody>
</table>
| tokenType    | Specifies the token type of the validation template. Possible values can be:  
|              | • username: indicates that the token type is UsernameToken  
|              | • saml11: indicates that the token type is a SAML 1.1 Assertion  
|              | • saml20: indicates that the token type is a SAML 2.0 Assertion  
|              | • x509: indicates that the token type is an X.509 certificate  
|              | • kerberos: indicates that the token type is a Kerberos token  
|              | • oam: indicates that the token type is OAM |
| defaultRequesterPPID | Specifies the Requester partner profile to use if OSTS is configured not to map the incoming message to a requester. |
| description  | Specifies an optional description. |
4.8.39.3 Example
The following invocation creates a new validation template with default configuration data, and displays the result of the operation.

createWSSValidationTemplate(templateId="custom-wss-validtemp", tokenType="custom", defaultRequesterPPID="requesterPartnerProfileTest", description="custom validation template")

4.8.40 createWSTrustValidationTemplate
Online command that creates a new WS-Trust validation template with default configuration data.

4.8.40.1 Description
Creates a new WS-Trust validation template with default configuration data, and displays the result of the operation.

The WS-Trust validation template is created with the values in Table 4–14, depending on the token type.

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Username</td>
<td>The WS-Trust validation template will be created with the following default values:</td>
</tr>
<tr>
<td></td>
<td>■ Timestamp Lifespan: 600 seconds</td>
</tr>
<tr>
<td></td>
<td>■ Enable Credential Validation: false</td>
</tr>
<tr>
<td></td>
<td>■ Validation Source: User Store</td>
</tr>
<tr>
<td></td>
<td>■ Token Mapping: Map token to User</td>
</tr>
<tr>
<td></td>
<td>■ Enable Simple User Mapping: true</td>
</tr>
<tr>
<td></td>
<td>■ UUser Datastore Attribute: uid</td>
</tr>
</tbody>
</table>
The WS-Trust validation template will be created with the following default values:

- Authentication Timeout: 3600 seconds
- Timestamp Lifespan: 3600 seconds

The Token Mapping section will be created with the following default values:

- Map token: Map token to User
- Enable Simple User Mapping: false
- Enable User NameID Mapping: true
- Enable Attribute Based User Mapping: false

Empty fields: User Datastore Attribute, Attribute Based User Mapping

User NameID Mapping table will be provisioned with the following entries as NameID format:

- urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName, empty datastore column
- urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName, dn set in the datastore column
- urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress, mail set in the datastore column
- urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified, empty datastore column
- urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos, empty datastore column
- urn:oasis:names:tc:SAML:2.0:nameid-format:persistent, empty datastore column

The WS-Trust Token Mapping section will be created with the following default values:

- Map token: Map token to User
- Enable Simple User Mapping: true
- Enable Attribute Based User Mapping: false
- Enable Simple Partner Mapping: true
- User Token Attribute: CN
- User Datastore Attribute: CN
- Attribute Based User Mapping (empty)

The WS-Trust Token Mapping section will be created with the following default values:

- Map token: Map token to User
- Enable Simple User Mapping: true
- Enable Attribute Based User Mapping: false
- Attribute Based User Mapping (empty)
- User Token Attribute: TPE_KERBEROS_PRINCIPAL_FULL
- User Datastore Attribute: mail

<table>
<thead>
<tr>
<th>Token Type</th>
<th>Description</th>
</tr>
</thead>
</table>
| SAML 1.1 or SAML 2.0 | The WS-Trust validation template will be created with the following default values:  
- Authentication Timeout: 3600 seconds  
- Timestamp Lifespan: 3600 seconds  
The Token Mapping section will be created with the following default values:  
- Map token: Map token to User  
- Enable Simple User Mapping: false  
- Enable User NameID Mapping: true  
- Enable Attribute Based User Mapping: false  
Empty fields: User Datastore Attribute, Attribute Based User Mapping  
User NameID Mapping table will be provisioned with the following entries as NameID format:  
- urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName, empty datastore column  
- urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName, dn set in the datastore column  
- urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress, mail set in the datastore column  
- urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified, empty datastore column  
- urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos, empty datastore column  
- urn:oasis:names:tc:SAML:2.0:nameid-format:persistent, empty datastore column  |
| X.509 | The WS-Trust Token Mapping section will be created with the following default values:  
- Map token: Map token to User  
- Enable Simple User Mapping: true  
- Enable Attribute Based User Mapping: false  
- Enable Simple Partner Mapping: true  
- User Token Attribute: CN  
- User Datastore Attribute: CN  
- Attribute Based User Mapping (empty)  |
| Kerberos | The WS-Trust Token Mapping section will be created with the following default values:  
- Map token: Map token to User  
- Enable Simple User Mapping: true  
- Enable Attribute Based User Mapping: false  
- Attribute Based User Mapping (empty)  
- User Token Attribute: TPE_KERBEROS_PRINCIPAL_FULL  
- User Datastore Attribute: mail |
4.8.40.2 Syntax

`createWSTrustValidationTemplate(templateId, tokenType, description)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>templateId</code></td>
<td>Specifies the name of the WS-Trust validation template to be created.</td>
</tr>
<tr>
<td><code>tokenType</code></td>
<td>Specifies the token type of the WS-Trust validation template. Possible values can be:</td>
</tr>
<tr>
<td></td>
<td>■ username: indicates that the token type is UsernameToken</td>
</tr>
<tr>
<td></td>
<td>■ saml11: indicates that the token type is a SAML 1.1 Assertion</td>
</tr>
<tr>
<td></td>
<td>■ saml20: indicates that the token type is a SAML 2.0 Assertion</td>
</tr>
<tr>
<td></td>
<td>■ x509: indicates that the token type is an X.509 certificate</td>
</tr>
<tr>
<td></td>
<td>■ kerberos: indicates that the token type is a Kerberos token</td>
</tr>
<tr>
<td></td>
<td>■ oam: indicates that the token type is an Oracle Access Manager token, supported by default</td>
</tr>
<tr>
<td></td>
<td>■ &lt;other&gt;: in this case, the token type is assumed to be a custom token type, referenced by &lt;other&gt; (replace &lt;other&gt; by a value)</td>
</tr>
<tr>
<td><code>description</code></td>
<td>Specifies an optional description.</td>
</tr>
</tbody>
</table>

4.8.40.3 Example

The following invocation creates a new WS-Trust validation template with default configuration data, and displays the result of the operation.

`createWSTrustValidationTemplate(templateId="custom-wss-validtemp", tokenType="custom", description="custom validation template")`
4.8.41.1 Description
Deletes a validation template referenced by the validationTemplateId parameter, and
displays the result of the operation.

4.8.41.2 Syntax
deleteValidationTemplate(validationTemplateId)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>validationTemplateId</td>
<td>Specifies the name of the validation template to be removed.</td>
</tr>
</tbody>
</table>

4.8.41.3 Example
The following invocation deletes a validation template referenced by the
validationTemplateId parameter, and displays the result of the operation.

deleteValidationTemplate(validationTemplateId="custom-wss-validtemp")

4.9 Oracle Keystore Service
This section contains commands used with the OPSS keystore service.

Table 4–15 lists the WLST commands used to manage the keystore service.

Table 4–15  OPSS Keystore Service Commands

<table>
<thead>
<tr>
<th>Use this Command...</th>
<th>to...</th>
</tr>
</thead>
<tbody>
<tr>
<td>changeKeyPassword</td>
<td>Change the password for a key.</td>
</tr>
<tr>
<td>changeKeyStorePassword</td>
<td>Change the password on a keystore.</td>
</tr>
<tr>
<td>createKeyStore</td>
<td>Create a keystore.</td>
</tr>
<tr>
<td>deleteKeyStore</td>
<td>Delete a keystore.</td>
</tr>
<tr>
<td>deleteKeyStoreEntry</td>
<td>Delete an entry in a keystore.</td>
</tr>
<tr>
<td>exportKeyStore</td>
<td>Export a keystore to file.</td>
</tr>
<tr>
<td>exportKeyStoreCertificate</td>
<td>Export a certificate to a file.</td>
</tr>
<tr>
<td>exportKeyStoreCertificateRequest</td>
<td>Export a certificate request to a file.</td>
</tr>
<tr>
<td>generateKeyPair</td>
<td>Generate a keypair.</td>
</tr>
<tr>
<td>generateSecretKey</td>
<td>Generate a secret key.</td>
</tr>
<tr>
<td>getKeyStoreCertificates</td>
<td>Get information about a certificate or trusted certificate.</td>
</tr>
<tr>
<td>getKeyStoreSecretKeyProperties</td>
<td>Get the secret key properties.</td>
</tr>
<tr>
<td>importKeyStore</td>
<td>Import a keystore from file.</td>
</tr>
<tr>
<td>importKeyStoreCertificate</td>
<td>Import a certificate or other object.</td>
</tr>
</tbody>
</table>
4.9.1 changeKeyPassword

Changes a key password.

4.9.1.1 Description
Changes the password for a key.

4.9.1.2 Syntax
svc.changeKeyPassword(appStripe='stripe', name='keystore', password='password',
alias='alias', currentkeypassword='currentkeypassword',
newkeypassword='newkeypassword')

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>svc</td>
<td>Specifies the service command object obtained through a call to getOpssService().</td>
</tr>
<tr>
<td>appStripe</td>
<td>Specifies the name of the stripe containing the keystore</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the keystore</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the keystore password</td>
</tr>
<tr>
<td>alias</td>
<td>Specifies the alias of the key entry whose password is changed</td>
</tr>
<tr>
<td>currentkeypassword</td>
<td>Specifies the current key password</td>
</tr>
<tr>
<td>newkeypassword</td>
<td>Specifies the new key password</td>
</tr>
</tbody>
</table>

4.9.1.3 Example
This example changes the password on the key entry orakey:
svc.changeKeyPassword(appStripe='system', name='keystore', password='password',
alias='orakey', currentkeypassword='currentkeypassword',
newkeypassword='newkeypassword')

4.9.2 changeKeyStorePassword

Changes the password of a keystore.

4.9.2.1 Description
Changes the password of the specified keystore.

4.9.2.2 Syntax
svc.changeKeyStorePassword(appStripe='stripe', name='keystore',
currentpassword='currentpassword', newpassword='newpassword')
4.9.2.3 Example
This example changes the password for keystore2.

```
svc.changeKeyStorePassword(appStripe='system', name='keystore2',
currentpassword='currentpassword', newpassword='newpassword')
```

4.9.3 createKeyStore
This keystore service command creates a new keystore.

4.9.3.1 Description
Creates a new keystore on the given application stripe.

4.9.3.2 Syntax
```
svc.createKeyStore(appStripe='stripe', name='keystore',
password='password',permission=true|false)
```

4.9.3.3 Example
This example creates a keystore named keystore1.

```
svc.createKeyStore(appStripe='system', name='keystore1',
password='password',permission=true)
```

4.9.4 deleteKeyStore
Deletes the named keystore.

4.9.4.1 Description
This keystore service command deletes a specified keystore.
4.9.4.2 Syntax

svc.deleteKeyStore(appStripe='stripe', name='keystore', password='password')

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>svc</td>
<td>Specifies the service command object obtained through a call to getOpssService().</td>
</tr>
<tr>
<td>appStripe</td>
<td>Specifies the name of the stripe where the keystore resides.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the keystore to be deleted.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the keystore password.</td>
</tr>
</tbody>
</table>

4.9.4.3 Example

This example deletes the keystore named keystore1.

svc.deleteKeyStore(appStripe='system', name='keystore1', password='password')

4.9.5 deleteKeyStoreEntry

Deletes a keystore entry.

4.9.5.1 Description

This command deletes the specified entry in a keystore.

4.9.5.2 Syntax

svc.deleteKeyStoreEntry(appStripe='stripe', name='keystore', password='password', alias='alias', keypassword='keypassword')

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>svc</td>
<td>Specifies the service command object obtained through a call to getOpssService().</td>
</tr>
<tr>
<td>appStripe</td>
<td>Specifies the name of the stripe where the keystore resides.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the keystore.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the keystore password.</td>
</tr>
<tr>
<td>alias</td>
<td>Specifies the alias of the entry to be deleted</td>
</tr>
<tr>
<td>keypassword</td>
<td>Specifies the key password of the entry to be deleted</td>
</tr>
</tbody>
</table>

4.9.5.3 Example

This example deletes a keystore entry denoted by alias orakey.

svc.deleteKeyStoreEntry(appStripe='system', name='keystore2', password='password', alias='orakey', keypassword='keypassword')

4.9.6 exportKeyStore

Exports a keystore to a file.

4.9.6.1 Description

Exports a keystore to the specified file.
4.9.6.2 Syntax

svc.exportKeyStore(appStripe='stripe', name='keystore', password='password',
aliases='comma-separated-aliases', keypasswords='comma-separated-keypasswords',
type='keystore-type', filepath='absolute_file_path')

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>svc</td>
<td>Specifies the service command object obtained through a call to getOppoService().</td>
</tr>
<tr>
<td>appStripe</td>
<td>Specifies the name of the stripe where the keystore resides.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the keystore.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the keystore password.</td>
</tr>
<tr>
<td>aliases</td>
<td>Comma separated list of aliases to be exported.</td>
</tr>
<tr>
<td>keypasswords</td>
<td>Comma separated list of the key passwords corresponding to aliases.</td>
</tr>
<tr>
<td>type</td>
<td>Exported keystore type. Valid values are 'JKS' or 'JCEKS'.</td>
</tr>
<tr>
<td>filepath</td>
<td>Absolute path of the file where keystore is exported.</td>
</tr>
</tbody>
</table>

4.9.6.3 Example

This example exports two aliases from the specified keystore.

svc.exportKeyStore(appStripe='system', name='keystore2',
password='password', aliases='orakey,seckey',
keypasswords='keypassword1,keypassword2',
type='JKS', filepath='/tmp/file.jks')

4.9.7 exportKeyStoreCertificate

Exports a certificate.

4.9.7.1 Description

Exports a certificate, trusted certificate or certificate chain.

4.9.7.2 Syntax

svc.exportKeyStoreCertificate(appStripe='stripe', name='keystore',
password='password', alias='alias', keypassword='keypassword',
type='entrytype', filepath='absolute_file_path')

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>svc</td>
<td>Specifies the service command object obtained through a call to getOppoService().</td>
</tr>
<tr>
<td>appStripe</td>
<td>Specifies the name of the stripe where the keystore resides.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the keystore.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the keystore password.</td>
</tr>
<tr>
<td>alias</td>
<td>Specifies the alias of the entry to be exported.</td>
</tr>
<tr>
<td>keypassword</td>
<td>Specifies the key password.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of keystore entry to be exported. Valid values are 'Certificate', 'TrustedCertificate' or 'CertificateChain'.</td>
</tr>
</tbody>
</table>
This example exports a certificate corresponding to the orakey alias:

```java
svc.exportKeyStoreCertificate(appStripe='system', name='keystore2', password='password', alias='orakey', keypassword='keypassword', type='Certificate', filepath='/tmp/cert.txt')
```

### 4.9.8 exportKeyStoreCertificateRequest

Exports a certificate request.

#### 4.9.8.1 Description

Generates and exports a certificate request from a keystore.

#### 4.9.8.2 Syntax

```java
svc.exportKeyStoreCertificateRequest(appStripe='stripe', name='keystore', password='password', alias='alias', keypassword='keypassword', filepath='absolute_file_path')
```

This example exports a certificate request corresponding to the orakey alias.

```java
svc.exportKeyStoreCertificateRequest(appStripe='system', name='keystore2', password='password', alias='orakey', keypassword='keypassword', filepath='/tmp/certreq.txt')
```

### 4.9.9 generateKeyPair

Generates a key pair in a keystore.

#### 4.9.9.1 Description

Generates a key pair in a keystore and wraps it in a demo CA-signed certificate.

#### 4.9.9.2 Syntax

```java
svc.generateKeyPair(appStripe='stripe', name='keystore', password='password',
```
Example
This example generates a keypair in keystore2.

```
svc.generateKeyPair(appStripe='system', name='keystore2', password='password',
    dn='cn=www.oracle.com', keysize='1024', alias='orakey', keypassword='keypassword')
```

### 4.9.10 generateSecretKey
Generates a secret key.

#### 4.9.10.1 Description
Generates a symmetric key in a keystore.

#### 4.9.10.2 Syntax
```
svc.generateSecretKey(appStripe='stripe', name='keystore', password='password',
    algorithm='algorithm', keysize='keysize', alias='alias',
    keypassword='keypassword')
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>svc</td>
<td>Specifies the service command object obtained through a call to getOpssService().</td>
</tr>
<tr>
<td>appStripe</td>
<td>Specifies the name of the stripe where the keystore resides.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the keystore.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the keystore password.</td>
</tr>
<tr>
<td>dn</td>
<td>Specifies the distinguished name of the certificate wrapping the key pair.</td>
</tr>
<tr>
<td>keysize</td>
<td>Specifies the key size.</td>
</tr>
<tr>
<td>alias</td>
<td>Specifies the alias of the key pair entry.</td>
</tr>
<tr>
<td>keypassword</td>
<td>Specifies the key password.</td>
</tr>
</tbody>
</table>

Example
This example generates a keypair with keysize 128 in keystore2.

```
svc.generateSecretKey(appStripe='system', name='keystore2', password='password',
    keysize='128')
```
algorithm='AES', keysize='128', alias='seckey', keypassword='keypassword')

4.9.11 getKeyStoreCertificates

Gets a certificate from the keystore.

4.9.11.1 Description
Retrieves information about a certificate or trusted certificate.

4.9.11.2 Syntax
svc.getKeyStoreCertificates(appStripe='stripe', name='keystore', password='password', alias='alias', keypassword='keypassword')

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>svc</td>
<td>Specifies the service command object obtained through a call to getOpssService().</td>
</tr>
<tr>
<td>appStripe</td>
<td>Specifies the name of the stripe where the keystore resides.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the keystore.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the keystore password.</td>
</tr>
<tr>
<td>alias</td>
<td>Specifies the alias of the certificate, trusted certificate or certificate chain to be displayed.</td>
</tr>
<tr>
<td>keypassword</td>
<td>Specifies the key password.</td>
</tr>
</tbody>
</table>

4.9.11.3 Example
This example gets certificates associated with keystore3.
svc.getKeyStoreCertificates(appStripe='system', name='keystore3', password='password', alias='alias', keypassword='keypassword')

4.9.12 getKeyStoreSecretKeyProperties

Retrieves secret key properties.

4.9.12.1 Description
Retrieves secret key properties like the algorithm.

4.9.12.2 Syntax
svc.getKeyStoreSecretKeyProperties(appStripe='stripe', name='keystore', password='password', alias='alias', keypassword='keypassword')

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>svc</td>
<td>Specifies the service command object obtained through a call to getOpssService().</td>
</tr>
<tr>
<td>appStripe</td>
<td>Specifies the name of the stripe where the keystore resides.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the keystore.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the keystore password.</td>
</tr>
<tr>
<td>alias</td>
<td>Specifies the alias of the secret key whose properties are displayed.</td>
</tr>
</tbody>
</table>
4.9.12.3 Example
This example gets properties for secret key seckey:

svc.getKeyStoreSecretKeyProperties(appStripe='system', name='keystore3',
password='password', alias='seckey', keypassword='keypassword')

4.9.13 importKeyStore
Imports a keystore from file.

4.9.13.1 Description
Imports a keystore from a system file.

4.9.13.2 Syntax
svc.importKeyStore(appStripe='stripe', name='keystore', password='password',
aliases='comma-separated-aliases', keypasswords='comma-separated-keypasswords',
type='keystore-type', permission=true|false, filepath='absolute_file_path')

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>svc</td>
<td>Specifies the service command object obtained through a call to getOpssService().</td>
</tr>
<tr>
<td>appStripe</td>
<td>Specifies the name of the stripe where the keystore resides.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the keystore.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the keystore password.</td>
</tr>
<tr>
<td>aliases</td>
<td>Specifies the comma-separated aliases of the entries to be imported from file.</td>
</tr>
<tr>
<td>keypasswords</td>
<td>Specifies the comma-separated passwords of the keys in file.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the imported keystore type. Valid values are 'JKS' or 'JCEKS'.</td>
</tr>
<tr>
<td>filepath</td>
<td>Specifies the absolute path of the keystore file to be imported.</td>
</tr>
<tr>
<td>permission</td>
<td>Specifies true if keystore is protected by permission only, false if protected by both permission and password.</td>
</tr>
</tbody>
</table>

4.9.13.3 Example
This example imports a file to keystore2:

svc.importKeyStore(appStripe='system', name='keystore2',
password='password', aliases='orakey,seckey', keypasswords='keypassword1,
keypassword2', type='JKS', permission=true, filepath='/tmp/file.jks')

4.9.14 importKeyStoreCertificate
Imports a certificate or other specified object.

4.9.14.1 Description
Imports a certificate, trusted certificate or certificate chain.
4.9.14.2 Syntax

```
svc.importKeyStoreCertificate(appStripe='stripe', name='keystore',
password='password', alias='alias', keypassword='keypassword',
type='entrytype', filepath='absolute_file_path')
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>svc</code></td>
<td>Specifies the service command object obtained through a call to getOpssService().</td>
</tr>
<tr>
<td><code>appStripe</code></td>
<td>Specifies the name of the stripe where the keystore resides.</td>
</tr>
<tr>
<td><code>name</code></td>
<td>Specifies the name of the keystore.</td>
</tr>
<tr>
<td><code>password</code></td>
<td>Specifies the keystore password.</td>
</tr>
<tr>
<td><code>alias</code></td>
<td>Specifies the alias of the entry to be imported.</td>
</tr>
<tr>
<td><code>keypassword</code></td>
<td>Specifies the key password of the newly imported entry.</td>
</tr>
<tr>
<td><code>type</code></td>
<td>Specifies the type of keystore entry to be imported. Valid values are 'Certificate', 'TrustedCertificate' or 'CertificateChain'.</td>
</tr>
<tr>
<td><code>filepath</code></td>
<td>Specifies the absolute path of the file from where certificate, trusted certificate or certificate chain is imported.</td>
</tr>
</tbody>
</table>

4.9.14.3 Example

This example imports a certificate into `keystore2`.

```
svc.importKeyStoreCertificate(appStripe='system', name='keystore2',
password='password', alias='orakey', keypassword='keypassword',
type='Certificate', filepath='/tmp/cert.txt')
```

4.9.15 `listExpiringCertificates`

Lists expiring certificates.

4.9.15.1 Description

Lists expiring certificates and optionally renews them.

4.9.15.2 Syntax

```
svc.listExpiringCertificates(days='days', autorenew=true|false)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>svc</code></td>
<td>Specifies the service command object obtained through a call to getOpssService().</td>
</tr>
<tr>
<td><code>days</code></td>
<td>Specifies that the list should only include certificates within this many days from expiration.</td>
</tr>
<tr>
<td><code>autorenew</code></td>
<td>Specifies true for automatically renewing expiring certificates, false for only listing them.</td>
</tr>
</tbody>
</table>

4.9.15.3 Example

This example lists certificates expiring within one year, and requests that they be renewed:

```
svc.listExpiringCertificates(days='365', autorenew=true)
```
4.9.16 listKeyStoreAliases

Lists the aliases in a keystore.

4.9.16.1 Description
Lists the aliases in a keystore for a given type of entry.

4.9.16.2 Syntax
The syntax is as follows:

```
svc.listKeyStoreAliases(appStripe='stripe', name='keystore', password='password', type='entrytype')
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>svc</td>
<td>Specifies the service command object obtained through a call to getOpssService().</td>
</tr>
<tr>
<td>appStripe</td>
<td>Specifies the name of the stripe where the keystore resides.</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name of the keystore.</td>
</tr>
<tr>
<td>password</td>
<td>Specifies the keystore password.</td>
</tr>
<tr>
<td>type</td>
<td>Specifies the type of entry for which aliases are listed. Valid values are 'Certificate', 'TrustedCertificate', 'SecretKey' or '*'.</td>
</tr>
</tbody>
</table>

4.9.16.3 Example
This example lists secret keys in keystore2:

```
svc.listKeyStoreAliases(appStripe='system', name='keystore2', password='password', type='SecretKey')
```

4.9.17 listKeyStores

Lists all the keystores in a stripe.

4.9.17.1 Description
Lists all the keystores in the specified stripe.

4.9.17.2 Syntax
```
svc.listKeyStores(appStripe='stripe')
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>svc</td>
<td>Specifies the service command object obtained through a call to getOpssService().</td>
</tr>
<tr>
<td>appStripe</td>
<td>Specifies the name of the stripe whose keystores are listed.</td>
</tr>
</tbody>
</table>

4.9.17.3 Example
This example lists all keystores on all stripes.

```
svc.listKeyStores(appStripe='*')
```
4.10 **Library Oracle Virtual Directory (libOVD) Commands**

Use the WLST commands listed in [Table 4–16](#) to manage a libOVD configuration associated with a specific Oracle Platform Security Services (OPSS) context.

**Table 4–16  WLST libOVD Commands**

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>addDNAttribute</td>
<td>Add an attribute to the DN attributes list for an existing adapter.</td>
<td>Online</td>
</tr>
<tr>
<td>activateLibOVDConfigChanges</td>
<td>Reload the libOVD configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>addAttributeExclusionRule</td>
<td>Add a attribute exclusion rule.</td>
<td>Online</td>
</tr>
<tr>
<td>addAttributeRule</td>
<td>Add a new attribute mapping rule.</td>
<td>Online</td>
</tr>
<tr>
<td>addDomainExclusionRule</td>
<td>Add a domain exclusion rule.</td>
<td>Online</td>
</tr>
<tr>
<td>addDomainRule</td>
<td>Add a new domain mapping rule.</td>
<td>Online</td>
</tr>
<tr>
<td>addJoinRule</td>
<td>Add a join rule to an existing Join Adapter for a libOVD configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>addLDAPHost</td>
<td>Add a new remote host to an existing LDAP adapter.</td>
<td>Online</td>
</tr>
<tr>
<td>addMappingContext</td>
<td>Create a new mapping context.</td>
<td>Online</td>
</tr>
<tr>
<td>addPlugin</td>
<td>Add a plug-in to an existing adapter or at the global level.</td>
<td>Online</td>
</tr>
<tr>
<td>addPluginParam</td>
<td>Add new parameter values to the existing adapter level plug-in or global plug-in.</td>
<td>Online</td>
</tr>
<tr>
<td>addToRequestControlExcludeList</td>
<td>Add a control to the Request Control Exclude List for an existing LDAP adapter configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>addToRequestControlIncludeList</td>
<td>Add a control to the Request Control Include List for an existing LDAP adapter configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>assignViewToAdapter</td>
<td>Assign the given view to an adapter.</td>
<td>Online</td>
</tr>
<tr>
<td>createJoinAdapter</td>
<td>Create a new Join Adapter for a libOVD configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>createLDAPAdapter</td>
<td>Create a new LDAP adapter for a libOVD configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>createLDAPAdapterWithDefaultPlugins</td>
<td>Create a new LDAP adapter with default plug-ins based on the specified directory type.</td>
<td>Online</td>
</tr>
<tr>
<td>createView</td>
<td>Create a new view.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAdapter</td>
<td>Delete an existing adapter for a libOVD configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAttributeExclusionRule</td>
<td>Delete a attribute exclusion rule.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAttributeRule</td>
<td>Delete a attribute mapping rule.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteDomainExclusionRule</td>
<td>Delete a domain exclusion rule.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteDomainRule</td>
<td>Delete a domain mapping rule.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteMappingContext</td>
<td>Delete the specified mapping context.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteView</td>
<td>Delete the specified view.</td>
<td>Online</td>
</tr>
</tbody>
</table>
Table 4–16  (Cont.) WLST libOVD Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>getAdapterDetails</td>
<td>Display the details of an existing adapter for a libOVD configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>listAdapters</td>
<td>List the name and type of all adapters that are configured for a libOVD configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>listAllMappingContextIds</td>
<td>List all the mapping contexts.</td>
<td>Online</td>
</tr>
<tr>
<td>listAttributeRules</td>
<td>List all the attribute rules.</td>
<td>Online</td>
</tr>
<tr>
<td>listDomainRules</td>
<td>List all the domain rules.</td>
<td>Online</td>
</tr>
<tr>
<td>listViews</td>
<td>List all views</td>
<td>Online</td>
</tr>
<tr>
<td>modifyLDAPAdapter</td>
<td>Modify the existing LDAP adapter configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>modifySocketOptions</td>
<td>Modify the socket options for an existing LDAP adapter configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>removeAllRequestControlExcludeList</td>
<td>Remove all controls from the Request Control Exclude List for an existing LDAP adapter configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>removeAllRequestControlIncludeList</td>
<td>Remove all controls from a Request Control Include List for an existing LDAP adapter configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>removeFromRequestControlExcludeList</td>
<td>Remove an attribute from the DN attributes list for an existing LDAP adapter configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>removeDNAttribute</td>
<td>Remove a control from the Request Control Exclude List for an existing LDAP adapter configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>removeFromRequestControlIncludeList</td>
<td>Remove a control from the Request Control Include List for an existing LDAP adapter configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>removeJoinRule</td>
<td>Remove a join rule from a Join Adapter configured for a libOVD configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>removeLDAPHost</td>
<td>Remove a remote host from an existing LDAP adapter configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>removePlugin</td>
<td>Remove a plug-in from an existing adapter or at the global level.</td>
<td>Online</td>
</tr>
<tr>
<td>removePluginParam</td>
<td>Remove an existing parameter from a configured adapter level plug-in or global plug-in.</td>
<td>Online</td>
</tr>
<tr>
<td>replacePluginParam</td>
<td>Replace existing parameter values for an adapter level plug-in or global plug-in.</td>
<td>Online</td>
</tr>
<tr>
<td>unassignViewFromAdapter</td>
<td>Remove a view from an adapter.</td>
<td>Online</td>
</tr>
<tr>
<td>listSSLStoreType</td>
<td>List the type of SSL store in use for libOVD.</td>
<td>Online</td>
</tr>
<tr>
<td>enableKSSForSSL</td>
<td>Enable KSS for libOVD.</td>
<td>Online</td>
</tr>
<tr>
<td>enableJKSForSSL</td>
<td>Enable JKS for libOVD.</td>
<td>Online</td>
</tr>
<tr>
<td>createKeyStoreAndEnableJKSForSSL</td>
<td>Enable JKS for libOVD.</td>
<td>Online</td>
</tr>
</tbody>
</table>
4.10.1 addDNAttribute

Adds an attribute to the DN Attributes List.

4.10.1.1 Description

Adds an attribute to the DN Attributes List for an existing adapter configured for the libOVD configuration associated with an OPSS context.

4.10.1.2 Syntax

addDNAttribute(adapterName='ldap1', attributeName='memberof', [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the adapter to be updated.</td>
</tr>
<tr>
<td>attributeName</td>
<td>Name of the new DN attribute to be added.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is “default.”</td>
</tr>
</tbody>
</table>

4.10.1.3 Example

addDNAttribute(adapterName='ldap1', attributeName='memberof', contextName='default')

4.10.2 activateLibOVDConfigChanges

Reloads the libOVD configuration.

4.10.2.1 Description

Reloads the libOVD configuration associated with a specific OPSS context.

4.10.2.2 Syntax

activateLibOVDConfigChanges([contextName])
4.10.2.3 Example
activateLibOVDConfigChanges('default')

4.10.3 addAttributeExclusionRule
Adds an attribute exclusion rule.

4.10.3.1 Description
Adds an attribute exclusion rule to the exclusion list.

4.10.3.2 Syntax
addAttributeExclusionRule(attribute, mappingContextId, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>attribute</td>
<td>Name of the attribute to be added to the exclusion list.</td>
</tr>
<tr>
<td>mappingContextId</td>
<td>Name of the mapping context.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.3.3 Example
addAttributeExclusionRule('objectsid','userrole')

4.10.4 addAttributeRule
Adds a new attribute mapping rule.

4.10.4.1 Description
Adds a new attribute mapping rule to the libOVD configuration associated with a specific OPSS context.

4.10.4.2 Syntax
addAttributeRule(srcAttrs, srcObjectClass, srcAttrType, dstAttr, dstObjectClass, dstAttrType, mappingExpression, direction, mappingContextId, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>mappingContextId</td>
<td>Name of the mapping context.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is &quot;default&quot;.</td>
</tr>
</tbody>
</table>

4.10.4.3 Example
addAttributeRule('lastname','','','sn','','','Inbound','userrole')
4.10.5 addDomainExclusionRule

Adds a domain exclusion rule.

4.10.5.1 Description

Adds a domain exclusion rule to the exclusion list.

4.10.5.2 Syntax

addDomainExclusionRule(domain, mappingContextId, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>domain</td>
<td>Distinguished name (DN) of the attribute to be added to the exclusion list.</td>
</tr>
<tr>
<td>mappingContextId</td>
<td>Name of the mapping context.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.5.3 Example

addDomainExclusionRule('cn=group,dc=oracle,dc=com','userrole')

4.10.6 addDomainRule

Adds a new domain mapping rule.

4.10.6.1 Description

Adds a new domain mapping rule.

4.10.6.2 Syntax

addDomainRule(srcDomain, destDomain, domainConstructRule, mappingContextId, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>srcDomain</td>
<td>Source domain.</td>
</tr>
<tr>
<td>destDomain</td>
<td>Destination domain</td>
</tr>
<tr>
<td>domainConstructRule</td>
<td>Name of the attribute to be added to the exclusion list.</td>
</tr>
<tr>
<td>mappingContextId</td>
<td>Name of the mapping context.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.6.3 Example

addDomainRule('dc=oracle,dc=com', 'dc=oracle,dc=com', '', 'defaultContext', 'default')

4.10.7 addJoinRule

Adds a join rule to a Join Adapter.
4.10.7.1 Description
Adds a join rule to an existing Join Adapter for the libOVD configuration associated with the specified OPSS context.

4.10.7.2 Syntax
addJoinRule(adapterName, secondary, condition, [joinerType], [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the Join Adapter to be modified.</td>
</tr>
<tr>
<td>secondary</td>
<td>Name of the adapter to join to.</td>
</tr>
<tr>
<td>condition</td>
<td>The attribute(s) to join on.</td>
</tr>
<tr>
<td>joinerType</td>
<td>Optional. Defines the type of Join. Values can be Simple (default), Conditional, OneToMany, or Shadow.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.7.3 Examples
addJoinRule('join1', 'secondaryldap', 'cn=cn', 'Simple', 'default')
addJoinRule('join1', 'secondaryldap', 'cn=cn', 'Conditional', 'default')
addJoinRule(adapterName='join1', secondary='LDAP3', condition='uid=cn', joinerType='OneToMany')
addJoinRule(adapterName='join1', secondary='LDAP2', condition='uid=cn', contextName='myContext')

4.10.8 addLDAPHost
Adds a new remote host.

4.10.8.1 Description
Adds a new remote host (host and port) to an existing LDAP adapter. By default, the new host is configured in Read-Write mode with percentage set to 100.

4.10.8.2 Syntax
addLDAPHost(adapterName, host, port, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the Join Adapter to be modified.</td>
</tr>
<tr>
<td>host</td>
<td>Remote LDAP host to which the LDAP adapter will communicate.</td>
</tr>
<tr>
<td>port</td>
<td>Remote LDAP host port.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.8.3 Examples
addLDAPHost(adapterName='ldap1', host='myhost.example.com', port=389)
addLDAPHost('ldap1', 'myhost.example.com', 389, 'myContext')

**4.10.9 addMappingContext**

Creates a new mapping context.

**4.10.9.1 Description**

Creates a new mapping context for the libOVD configuration associated with the specified OPSS context.

**4.10.9.2 Syntax**

```
addMappingContext(mappingContextId, [contextName])
```

**Argument** | **Definition**
--- | ---
`mappingContextId` | Name of the mapping context.
`contextName` | Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is `default`.

**4.10.9.3 Example**

```
addMappingContext('defaultContext', 'context')
```

**4.10.10 addPlugin**

Adds a plug-in to an existing adapter or at the global level.

**4.10.10.1 Description**

Adds a plug-in to an existing adapter or at the global level. The "i"th key corresponds to "i"th value. The plug-in is added to default chain.

**4.10.10.2 Syntax**

```
addPlugin(pluginName, pluginClass, paramKeys, paramValues, [adapterName], [contextName])
```

**Argument** | **Definition**
--- | ---
`pluginName` | Name of the plug-in to be created.
`pluginClass` | Class of the plug-in.
`paramKeys` | Init Param. Keys separated by "|".
`paramValues` | Init Param. Values separated by "|".
`adapterName` | Optional. Name of the adapter to be modified. If not specified, the plug-in is added at the global level.
`contextName` | Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is `default`.

**4.10.10.3 Examples**

```
addPlugin(adapterName='ldap1',
    pluginName='VirtualAttr',
    pluginClass='oracle.ods.virtualization.engine.chain.plugins.virtualattr.VirtualAttributePlugin',
    paramKeys='AddAttribute | MatchFilter',
    paramValues='myContext')
```
Library Oracle Virtual Directory (libOVD) Commands

ContainerDN', paramValues='cn=%uid% | objectclass=person | dc=oracle,dc=com')

addPlugin(pluginName='VirtualAttr', pluginClass='oracle.ods.virtualization.engine.chain.plugins.virtualattr.VirtualAttributePlugin', paramKeys='AddAttribute | MatchFilter | ContainerDN', paramValues='cn=%uid% | objectclass=person | dc=oracle,dc=com')

addPlugin(pluginName='DMSMetrics', pluginClass='oracle.ods.virtualization.engine.chain.plugins.DMSMetrics.MonitorPerformance', paramKeys='None', paramValues='None', adapterName='ldap1', contextName='default')

4.10.11 addPluginParam

Adds new parameter values to the existing adapter level plug-in or global plug-in.

4.10.11.1 Description

Adds new parameter values to the existing adapter level plug-in or the global plug-in. If the parameter already exists, the new value is added to the existing set of values.

The 'i'th key corresponds to 'i'th value.

4.10.11.2 Syntax

addPluginParam(pluginName, paramKeys, paramValues, [adapterName], [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pluginName</td>
<td>Name of the plug-in to be modified.</td>
</tr>
<tr>
<td>paramKeys</td>
<td>Init Param Keys separated by &quot;</td>
</tr>
<tr>
<td>paramValues</td>
<td>Init Param Values separated by &quot;</td>
</tr>
<tr>
<td>adapterName</td>
<td>Optional. Name of the adapter to be modified. If not specified, the global plug-in is modified.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.11.3 Examples

addPluginParam(adapterName='ldap1', pluginName='VirtualAttr', paramKeys='ReplaceAttribute | MatchFilter', paramValues='cn=%uid% | objectclass=person')

addPluginParam(pluginName='VirtualAttr', paramKeys='ReplaceAttribute | MatchFilter', param)

4.10.12 addToRequestControlExcludeList

Adds a control to the Request Control Exclude List.

4.10.12.1 Description

Adds a control to the Request Control Exclude List for an existing LDAP adapter configuration.

4.10.12.2 Syntax

addToRequestControlExcludeList(adapterName, control, [contextName])
4.10.12.3 Example
addToRequestControlExcludeList(adapterName='ldap1',
control='2.16.840.1.113894.1.8.31', contextName='default')

4.10.13 addToRequestControlIncludeList
Adds a control to the Request Control Include List.

4.10.13.1 Description
Adds a control to the Request Control Include List for an existing LDAP adapter configuration.

4.10.13.2 Syntax
addToRequestControlIncludeList(adapterName, control, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the LDAP adapter to be modified.</td>
</tr>
<tr>
<td>control</td>
<td>LDAP control object identifier (OID).</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.13.3 Example
addToRequestControlIncludeList(adapterName='ldap1',
control='2.16.840.1.113894.1.8.31', contextName='default')

4.10.14 assignViewToAdapter
Assigns a view to an LDAP adapter.

4.10.14.1 Description
Assigns a view to an LDAP adapter in the libOVD configuration associated with an OPSS context.

4.10.14.2 Syntax
assignViewToAdapter(viewName, adapterName, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>viewName</td>
<td>Name of the view.</td>
</tr>
<tr>
<td>adapterName</td>
<td>Name of the LDAP adapter.</td>
</tr>
</tbody>
</table>
**4.10.14.3 Example**

`assignViewToAdapter('userView', 'ldap1', 'default')`

---

**4.10.15 `createJoinAdapter`**

Creates a new Join Adapter.

**4.10.15.1 Description**

Creates a new Join Adapter for the libOVD configuration associated with an OPSS context.

**4.10.15.2 Syntax**

`createJoinAdapter(adapterName, root, primaryAdapter, [bindAdapter], [contextName])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>adapterName</strong></td>
<td>Name of the Join Adapter to be created.</td>
</tr>
<tr>
<td><strong>root</strong></td>
<td>Root.</td>
</tr>
<tr>
<td><strong>primaryAdapter</strong></td>
<td>Specifies the identifier of the primary adapter, which is the adapter searched first in the join operation.</td>
</tr>
<tr>
<td><strong>bindAdapter</strong></td>
<td>Specifies identifier of the bind adapter(s), which are the adapter(s) whose proxy account is used to bind in the LDAP operation. By default, <code>primaryAdapter</code> is set as <code>bindAdapter</code>.</td>
</tr>
<tr>
<td><strong>contextName</strong></td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is &quot;default&quot;.</td>
</tr>
</tbody>
</table>

**4.10.15.3 Examples**

`createJoinAdapter('join1', 'dc=join', 'primaryldap', 'myldap', 'myContext')`

`createJoinAdapter(adapterName='join1', root='dc=join', primaryAdapter='myldap')`

---

**4.10.16 `createLDAPAdapter`**

Creates a new LDAP adapter.

**4.10.16.1 Description**

Creates a new LDAP adapter for the libOVD configuration associated with an OPSS context.

**4.10.16.2 Syntax**

`createLDAPAdapter(adapterName, root, host, port, remoteBase, [isSecure], [bindDN], [bindPasswd], [passCred], [contextName])`

---

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>adapterName</strong></td>
<td>Name of the LDAP adapter to be created.</td>
</tr>
<tr>
<td><strong>root</strong></td>
<td>Root.</td>
</tr>
<tr>
<td><strong>host</strong></td>
<td>Host.</td>
</tr>
<tr>
<td><strong>port</strong></td>
<td>Port.</td>
</tr>
<tr>
<td><strong>remoteBase</strong></td>
<td>Specifies the base DN of the LDAP server.</td>
</tr>
<tr>
<td><strong>isSecure</strong></td>
<td>Specifies whether SSL is enabled.</td>
</tr>
<tr>
<td><strong>bindDN</strong></td>
<td>Specifies the bind DN.</td>
</tr>
<tr>
<td><strong>bindPasswd</strong></td>
<td>Specifies the bind password.</td>
</tr>
<tr>
<td><strong>passCred</strong></td>
<td>Specifies the password credentials.</td>
</tr>
<tr>
<td><strong>contextName</strong></td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is &quot;default&quot;.</td>
</tr>
</tbody>
</table>
4.10.16.3 Examples

createLDAPAdapter("testLDAP", "dc=us,dc=oracle,dc=com", "myhost.example.com", 3060, "dc=uk,dc=oid"); false, "cn=testuser", "welcome1", "Always", "myContext"

createLDAPAdapter(adapterName='ldap1', root='dc=com', host='myhost.example.com', port=5566, remoteBase='dc=oid')

4.10.17 createLDAPAdapterWithDefaultPlugins

Creates a new LDAP adapter with default plug-ins based on the directory type for the libOVD configuration associated with an OPSS context.

4.10.17.1 Description

Creates a new LDAP adapter with default plug-ins based on the directory type for the libOVD configuration associated with an OPSS context.

4.10.17.2 Syntax

createLDAPAdapterWithDefaultPlugins(adapterName, directoryType, root, host, port, remoteBase, [isSecure], [bindDN], [bindPasswd], [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the LDAP adapter to be created.</td>
</tr>
<tr>
<td>root</td>
<td>Virtual Namespace of the LDAP adapter.</td>
</tr>
<tr>
<td>host</td>
<td>Remote LDAP host with which the LDAP adapter will communicate.</td>
</tr>
<tr>
<td>port</td>
<td>Remote LDAP host port number.</td>
</tr>
<tr>
<td>remoteBase</td>
<td>Location in the remote DIT to which root corresponds.</td>
</tr>
<tr>
<td>isSecure</td>
<td>Optional. Boolean value that enables secure SSL/TLS connections to the remote hosts when set to true. The default value is false.</td>
</tr>
<tr>
<td>bindDN</td>
<td>Optional. Proxy BindDN used to communicate with remote host. Default value is &quot;&quot;.</td>
</tr>
<tr>
<td>bindPasswd</td>
<td>Optional. Proxy BindPasswd used to communicate with the remote host. Default value is &quot;&quot;.</td>
</tr>
<tr>
<td>passCred</td>
<td>Optional. Controls the credentials, if any, the libOVD configuration will pass to the back-end (remote host) LDAP server. Values can be Always (default), None, or BindOnly.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>
### 4.10.18 createView

Creates a new view.

#### 4.10.18.1 Description

Creates a new view for the libOVD configuration associated with an OPSS context.

#### 4.10.18.2 Syntax

```plaintext
createView(viewName, [contextName])
```

#### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>viewName</code></td>
<td>Name of the new view.</td>
</tr>
</tbody>
</table>

**Examples**

```plaintext
createViewWithDefaultPlugins('testLDAP', 'OID', 'dc=us,dc=oracle,dc=com',
"myhost.example.domain.com", 3060, 'dc=uk,dc=oid', false, "cn=testuser",
"welcome1", "myContext")
```

```plaintext
createViewWithDefaultPlugins('ldap1', 'OID', 'dc=com', 'myhost.example.domain.com',
5566, 'dc=oid', 'cn=testuser', 'welcome1', 'defualt')
```
**4.10.18.3 Example**
```
cREATEVIEW('userView', 'default')
```

### 4.10.19 deleteAdapter

Deletes an existing adapter.

**4.10.19.1 Description**

Deletes an existing adapter for the libOVD configuration associated with an OPSS context.

**4.10.19.2 Syntax**
```
deleteAdapter(adapterName, [contextName])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the Join Adapter to be deleted.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

**4.10.19.3 Examples**
```
deleteAdapter(adapterName='join1')
deleteAdapter('join1', 'default')
```

### 4.10.20 deleteAttributeExclusionRule

Deletes an attribute exclusion rule.

**4.10.20.1 Description**

Deletes an attribute exclusion rule for the libOVD configuration associated with an OPSS context.

**4.10.20.2 Syntax**
```
deleteAttributeExclusionRule(attribute, mappingContextId, [contextName])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>attribute</td>
<td>Name of the attribute to be removed from the exclusion list.</td>
</tr>
<tr>
<td>mappingContextId</td>
<td>Name of the mapping context.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

**4.10.20.3 Example**
```
deleteAttributeExclusionRule('objectsid', 'userrole')
```
4.10.21 deleteAttributeRule

Delete an attribute mapping rule.

4.10.21.1 Description
Delete an attribute mapping rule for the libOVD configuration associated with an OPSS context.

4.10.21.2 Syntax
`deleteAttributeRule(srcAttrs, dstAttr, mappingContextId, [contextName])`

4.10.21.3 Example
`deleteAttributeRule('lastname','sn')`

4.10.22 deleteDomainExclusionRule

Deletes a domain exclusion rule.

4.10.22.1 Description
Deletes a domain exclusion rule for the libOVD configuration associated with an OPSS context.

4.10.22.2 Syntax
`deleteDomainExclusionRule(domain, mappingContextId, [contextName])`

4.10.22.3 Example
`deleteDomainExclusionRule('cn=group,dc=oracle,dc=com','userrole')`

4.10.23 deleteDomainRule

Deletes a domain mapping rule.

4.10.23.1 Argument
Argument Definition
`srcAttrs` Source attributes.
`dstAttr` Destination attribute.
`mappingContextId` Name of the mapping context.
`contextName` Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is `default`.

4.10.23.2 Syntax
`deleteDomainRule(srcAttrs, dstAttr, mappingContextId, [contextName])`

4.10.23.3 Example
`deleteDomainRule('lastname','sn')`
4.10.23.1 Description
Deletes a domain mapping rule for the libOVD configuration associated with an OPSS context.

4.10.23.2 Syntax
deleteDomainRule(srcDomain, destDomain, mappingContextId, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>srcDomain</td>
<td>Source domain.</td>
</tr>
<tr>
<td>destDomain</td>
<td>Destination domain.</td>
</tr>
<tr>
<td>mappingContextId</td>
<td>Name of the mapping context.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.23.3 Example
deleteDomainRule('dc=oracle,dc=com','dc=oracle,dc=com','userrole')

4.10.24 deleteMappingContext
Delete a mapping context.

4.10.24.1 Description
Delete the specified mapping context for the libOVD configuration associated with an OPSS context.

4.10.24.2 Syntax
deleteMappingContext(mappingContextId, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>mappingContextId</td>
<td>Name of the mapping context.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.24.3 Example
deleteMappingContext('defaultContext','context')

4.10.25 deleteView
Deletes a view.

4.10.25.1 Description
Deletes a view for the libOVD configuration associated with an OPSS context.

4.10.25.2 Syntax
createView(viewName, [contextName])
4.10.25 Examples
deleteView('userView','default')

4.10.26 getAdapterDetails
Displays the details of an existing adapter.

4.10.26.1 Description
Displays the details of an existing adapter configured for the libOVD configuration associated with an OPSS context.

4.10.26.2 Syntax
getAdapterDetails(adapterName, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the adapter that contains the details to be displayed.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.26.3 Examples
getAdapterDetails(adapterName='ldap1', contextName='default')
getAdapterDetails(adapterName='join1')

4.10.27 listAdapters
Lists the name and type of all adapters.

4.10.27.1 Description
Lists the name and type of all adapters that are configured for the libOVD configuration associated with an OPSS context.

4.10.27.2 Syntax
listAdapters([contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.27.3 Examples
listAdapters()
listAdapters(contextName='myContext')

4.10.28 listAllMappingContextIds
Lists all mapping contexts.

4.10.28.1 Description
Lists the mapping contexts associated with the specified OPSS context.

4.10.28.2 Syntax
listAllMappingContextIds([contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.28.3 Example
listAllMappingContextIds('default')

4.10.29 listAttributeRules
Lists all the attribute rules.

4.10.29.1 Description
List all the attribute rules in the format SOURCE_ATTRIBUTE:DESTINATION_ATTRIBUTE:DIRECTION.

4.10.29.2 Syntax
listAttributeRules(mappingContextId, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>mappingContextId</td>
<td>Name of the mapping context.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.29.3 Example
listAttributeRules('defaultContext','default')

4.10.30 listDomainRules
Lists all domain rules.

4.10.30.1 Description
Lists all the domain rules in the format of SOURCE_DOMAIN:DESTINATION_DOMAIN.
4.10.30.2 Syntax
listDomainRules(mappingContextId, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>mappingContextId</td>
<td>Name of the mapping context.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.30.3 Example
listDomainRules('defaultContext','default')

4.10.31 listViews
Lists all views

4.10.31.1 Description
Lists all views for a libOVD configuration associated with an OPSS context.

4.10.31.2 Syntax
listViews([contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.31.3 Example
listViews('default')

4.10.32 modifyLDAPAdapter
Modifies parameters in an LDAP adapter.

4.10.32.1 Description
Modifies the following parameters defined in an existing LDAP adapter:

- Remote Base
- Root
- Secure
- BindDN
- BindPassword
- PassCredentials
- MaxPoolSize
- MaxPoolChecks
- MaxPoolWait
- InitialPoolSize
- PoolCleanupInterval
- MaxPoolConnectionIdleTime
- Active
- PingProtocol
- PingBindDN
- PingBindPassword
- PageSize
- HeartBeatInterval
- OperationTimeout
- SearchCountLimit
- Visible
- Critical
- InclusionFilter
- ExclusionFilter
- DNPattern
- RequestControlAllowServerSupported

### 4.10.32.2 Syntax

```sql
modifyLDAPAdapter(adapterName, attribute, value, [contextName])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the LDAP adapter to be modified.</td>
</tr>
<tr>
<td>attribute</td>
<td>Name of the attribute to be modified.</td>
</tr>
<tr>
<td>value</td>
<td>New value for the attribute.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

### 4.10.32.3 Examples

```sql
modifyLDAPAdapter(adapterName='ldap1', attribute='Root', value='dc=us, dc=oracle, dc=com', contextName='mydefault')
```

```sql
modifyLDAPAdapter(adapterName='ldap1', attribute='RemoteBase', value='dc=org', contextName='mydefault')
```

```sql
modifyLDAPAdapter(adapterName='ldap1', attribute='PassCredentials', value='BindOnly', contextName='mydefault')
```

```sql
modifyLDAPAdapter(adapterName='ldap1', attribute='BindDN', value='cn=proxyuser,dc=com', contextName='mydefault')
```

```sql
modifyLDAPAdapter(adapterName='ldap1', attribute='BindPassword', value='testwelcome123', contextName='mydefault')
```

```sql
modifyLDAPAdapter(adapterName='ldap1', attribute='Secure', value=true, contextName='mydefault')
```
modifyLDAPAdapter(adapterName='ldap1', attribute='MaxPoolSize', value=500, contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='MaxPoolChecks', value=10, contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='MaxPoolWait', value=120000, contextName='mydefault') [value is in milliseconds]

modifyLDAPAdapter(adapterName='ldap1', attribute='InitialPoolSize', value=10, contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='PoolCleanupInterval', value=300, contextName='mydefault') [value is in seconds]

modifyLDAPAdapter(adapterName='ldap1', attribute='Active', value=false, contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='PingProtocol', value='LDAP', contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='PingBindDN', value='cn=proxyuser', contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='PingBindPassword', value='welcome1', contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='PageSize', value=500, contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='HeartBeatInterval', value=120, contextName='mydefault') [value is in seconds]

modifyLDAPAdapter(adapterName='ldap1', attribute='OperationTimeout', value=120000, contextName='mydefault') [value is in milliseconds]

modifyLDAPAdapter(adapterName='ldap1', attribute='SearchCountLimit', value=100, contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='Visible', value='Yes', contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='Critical', value='false', contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='InclusionFilter', value='objectclass=inetorgperson#base', contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='ExclusionFilter', value='uniquemember=*#base', contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='DNPattern', value='(\.)cn=[a-z0-9]*$', contextName='mydefault')

modifyLDAPAdapter(adapterName='ldap1', attribute='RequestControlAllowServerSupported', value=false,
4.10.33 modifySocketOptions

Modifies socket options.

4.10.33.1 Description
Modifies socket options for an existing LDAP adapter configuration.

4.10.33.2 Syntax
modifySocketOptions(adapterName, reuseAddress, keepAlive, tcpNoDelay, readTimeout, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the LDAP adapter to be modified.</td>
</tr>
<tr>
<td>reuseAddress</td>
<td>Value of reuseAddress.</td>
</tr>
<tr>
<td>keepAlive</td>
<td>Value of keepAlive.</td>
</tr>
<tr>
<td>tcpNoDelay</td>
<td>Value of tcpNoDelay.</td>
</tr>
<tr>
<td>readTimeout</td>
<td>Value of readTimeout in seconds.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.33.3 Example
modifySocketOptions(adapterName='ldap1', reuseAddress=true, keepAlive=true, tcpNoDelay=true, readTimeout=180000, contextName='default')

4.10.34 removeAllRequestControlExcludeList

Removes all controls from the Request Control Exclude List.

4.10.34.1 Description
Removes all controls from the Request Control Exclude List for an existing LDAP adapter configuration.

4.10.34.2 Syntax
removeAllRequestControlExcludeList(adapterName, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the adapter to be updated.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.34.3 Example
removeAllRequestControlExcludeList(adapterName='ldap1', contextName='default')
4.10.35  **removeAllRequestControlIncludeList**

Removes all controls from the Request Control Include List.

### 4.10.35.1 Description
Removes all controls from the Request Control Include List for an existing LDAP adapter configuration.

### 4.10.35.2 Syntax
`removeAllRequestControlIncludeList(adapterName, [contextName])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the adapter to be updated.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is \texttt{default}.</td>
</tr>
</tbody>
</table>

### 4.10.35.3 Example
`removeAllRequestControlIncludeList(adapterName='ldap1', contextName='default')`

4.10.36  **removeFromRequestControlExcludeList**

Removes a control from the Request Control Exclude List.

### 4.10.36.1 Description
Removes a control from the Request Control Exclude List for an existing LDAP adapter configuration.

### 4.10.36.2 Syntax
`removeFromRequestControlExcludeList(adapterName, control, [contextName])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the LDAP adapter to be modified.</td>
</tr>
<tr>
<td>control</td>
<td>LDAP control object identifier (OID).</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is \texttt{default}.</td>
</tr>
</tbody>
</table>

### 4.10.36.3 Example
`removeFromRequestControlExcludeList(adapterName='ldap1', control='2.16.840.1.113894.1.8.31', contextName='default')`

4.10.37  **removeDNAttribute**

Removes a attribute from the DN Attributes List.

### 4.10.37.1 Description
Removes a attribute from the DN Attributes List for an existing adapter that is configured for the libOVD associated with an OPSS context.
4.10.37.2 Syntax
removeDNAttribute(adapterName, attributeName, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the adapter to be updated.</td>
</tr>
<tr>
<td>attributeName</td>
<td>Name of the new DN attribute to be removed.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.37.3 Example
removeDNAttribute(adapterName='ldap1', attributeName='memberof', contextName='default')

4.10.38 removeFromRequestControlIncludeList
Removes a control from the Request Control Include List.

4.10.38.1 Description
Removes a control from the Request Control Include List for an existing LDAP adapter configuration.

4.10.38.2 Syntax
removeFromRequestControlIncludeList(adapterName, control, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the LDAP adapter to be modified.</td>
</tr>
<tr>
<td>control</td>
<td>LDAP control object identifier (OID).</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.38.3 Example
removeFromRequestControlIncludeList(adapterName='ldap1', control='2.16.840.1.113894.1.8.31', contextName='default')

4.10.39 removeJoinRule
Removes a join rule from a Join Adapter.

4.10.39.1 Description
Removes a join rule from a Join Adapter configured for the libOVD configuration associated with the specified OPSS context.

4.10.39.2 Syntax
removeJoinRule(adapterName, secondary, [contextName])
### removeJoinRule

Removes a join rule from an existing Join Adapter.

#### Description
Removes a join rule from the Join Adapter.

#### Syntax
```
removeJoinRule('join1', 'secondary1dap', 'default')
removeJoinRule(adapterName='join1', secondary='LDAP3')
```

#### Examples
```
removeJoinRule('join1', 'secondary1dap', 'default')
removeJoinRule(adapterName='join1', secondary='LDAP3')
```

### removeLDAPHost

Removes a remote host from an existing LDAP adapter.

#### Description
Removes a remote host (host:port) from an existing LDAP adapter.

#### Syntax
```
removeLDAPHost(adapterName, host, [contextName])
```

#### Argument
<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the LDAP adapter to be modified.</td>
</tr>
<tr>
<td>host</td>
<td>Location of a remote LDAP host with which the</td>
</tr>
<tr>
<td></td>
<td>LDAP adapter will communicate.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which</td>
</tr>
<tr>
<td></td>
<td>the libOVD configuration is associated. Default</td>
</tr>
<tr>
<td></td>
<td>value is default.</td>
</tr>
</tbody>
</table>

#### Examples
```
removeLDAPHost('ldap1', 'myhost.example.com')
removeLDAPHost('ldap1', 'myhost.example.com', 'myContext')
```

### removePlugin

Removes a plug-in from an existing adapter.

#### Description
Removes a plug-in from an existing adapter or at the global level.

#### Syntax
```
removePlugin(pluginName, [adapterName], [contextName])
```

#### Argument
<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pluginName</td>
<td>Name of the plug-in to be removed.</td>
</tr>
</tbody>
</table>
4.10.41  Examples
removePlugin(adapterName='ldap1', pluginName='VirtualAttr')
removePlugin(pluginName='VirtualAttr')

4.10.42  removePluginParam
Removes an existing parameter from a configured adapter level plug-in.

4.10.42.1  Description
Removes an existing parameter from a configured adapter level plug-in or a global plug-in. This command removes all values of the particular parameter from the plug-in.

4.10.42.2  Syntax
removePluginParam(pluginName, paramKey, [adapterName], [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pluginName</td>
<td>Name of the plug-in to be modified.</td>
</tr>
<tr>
<td>paramKey</td>
<td>Parameter to be removed.</td>
</tr>
<tr>
<td>adapterName</td>
<td>Optional. Name of the adapter to be modified. If not specified, the global plug-in is modified.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.42.3  Example
removePluginParam(adapterName='ldap1', pluginName='VirtualAttr', paramKey='ReplaceAttribute')
removePluginParam(pluginName='VirtualAttr', paramKey='ReplaceAttribute')

4.10.43  replacePluginParam
Replaces existing parameter values for a plug-in.

4.10.43.1  Description
Replaces existing parameter values for the specified adapter level plug-in or global plug-in.

4.10.43.2  Syntax
replacePluginParam(pluginName, paramName, paramValues,
4.10.43 Examples

replacePluginParam(adapterName='ldap1', pluginName='VirtualAttr',
paramName='ReplaceAttribute', paramValues='cn=%uid%')

replacePluginParam(adapterName='ldap1', pluginName='UserManagement',
paramName='mapAttribute', paramValues='orclguid=objectGuid | uniquemember=member')

4.10.44 unassignViewFromAdapter

Unassigns a view from an adapter.

4.10.44.1 Description

Unassigns a view from an LDAP adapter configuration.

4.10.44.2 Syntax

unassignViewFromAdapter(viewName, adapterName, [contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>viewName</td>
<td>Name of the view.</td>
</tr>
<tr>
<td>adapterName</td>
<td>Name of the LDAP adapter.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which the libOVD configuration is associated. Default value is default.</td>
</tr>
</tbody>
</table>

4.10.44.3 Example

unassignViewFromAdapter('userView','ldap1', 'default')

4.10.45 listSSLStoreType

Online command that lists the type of SSL store in use.

4.10.45.1 Description

This command lists the type of SSL store in use for libOVD (JKS or KSS).

4.10.45.2 Syntax

listSSLStoreType(contextName=[contextName])
4.10.45.3 Example
listSSLStoreType(contextName='default')

4.10.46 enableKSSForsSL
Online command to enable KSS for libOVD.

4.10.46.1 Description
This command enables KSS for SSL, and disables JKS if it was enabled before. For more information about KSS, see Oracle® Fusion Middleware Securing Applications with Oracle Platform Security Services.

4.10.46.2 Syntax
enableKSSForsSL(contextName=[contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is default.</td>
</tr>
</tbody>
</table>

4.10.46.3 Example
enableKSSForsSL(contextName='default')

4.10.47 enableJKSForsSL
Online command to enable JKS for libOVD.

4.10.47.1 Description
This command enables JKS for SSL, and disables KSS if it was enabled before. The command assumes that the libOVD adapters.jks file exists.

4.10.47.2 Syntax
enableJKSForsSL(contextName=[contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is default.</td>
</tr>
</tbody>
</table>

4.10.47.3 Example
enableJKSForsSL(contextName='default')

4.10.48 createKeyStoreAndEnableJKSForsSL
Online command to enable JKS for SSL.
4.10.48.1 Description
This command enables JKS for SSL, and disables KSS if it was enabled before. The command creates the libOVD adapters.jks file.

4.10.48.2 Syntax
createKeyStoreAndEnableJKSForSSL(keystorePassword=[password],
contextName=[contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>keystorePassword</td>
<td>Password for libOVD adapters.jks file.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is default.</td>
</tr>
</tbody>
</table>

4.10.48.3 Example
createKeyStoreAndEnableJKSForSSL(keystorePassword='welcome1',
contextName='default')

4.10.49 importTrustedCertificateIntoSSLStore
Online command to import trusted certificate into SSL store.

4.10.49.1 Description
This command imports the provided trusted certificate into SSL store.

4.10.49.2 Syntax
importTrustedCertificateIntoSSLStore(certificateFileName=[cert_file],
aliasName=[aliasName],contextName=[contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>certificateFileName</td>
<td>File name that contains the certificate.</td>
</tr>
<tr>
<td>aliasName</td>
<td>Alias name for the certificate.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is default.</td>
</tr>
</tbody>
</table>

4.10.49.3 Example
importTrustedCertificateIntoSSLStore(certificateFileName='/tmp/cert.txt',
aliasName='myCert1',contextName='default')

4.10.50 migrateAllTrustedCertificatesFromJKSToKSS
Online command to migrate all trusted certificates from JKS-based libOVD truststore to KSS store.

4.10.50.1 Description
This command migrates all trusted certificates from JKS-based libOVD truststore to KSS store.
4.10.50.2 Syntax
migrateAllTrustedCertificatesFromJKSToKSS(contextName=[contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is <code>default</code>.</td>
</tr>
</tbody>
</table>

4.10.50.3 Example
migrateAllTrustedCertificatesFromJKSToKSS(contextName='default')

4.10.51 migrateTrustedCertificatesFromJKSToKSS
Online command to migrate given trusted certificates from JKS-based libOVD truststore to KSS store.

4.10.51.1 Description
This command migrates the given trusted certificates from JKS-based libOVD truststore to KSS store.

4.10.51.2 Syntax
migrateTrustedCertificatesFromJKSToKSS(aliasNames=[alias_names], contextName=[contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>aliasNames</td>
<td>List of alias names to migrate separated by a comma.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is <code>default</code>.</td>
</tr>
</tbody>
</table>

4.10.51.3 Example
migrateTrustedCertificatesFromJKSToKSS (aliasNames='alias1,alias2', contextName='default')

4.10.52 changeLDAPHostPort
Online command to change given LDAP host and port in an existing LDAP adapter configuration to a new host and port.

4.10.52.1 Description
This command changes given LDAP host and port in an existing LDAP adapter configuration to a new host and port.

4.10.52.2 Syntax
changeLDAPHostPort(adapterName=[adapterName], oldHost=[oldHost], oldPort=[oldPort], newHost=[newHost], newPort=[newPort], contextName=[contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the LDAP adapter to be modified.</td>
</tr>
</tbody>
</table>
4.10.52.3 Example
changeLDAPHostPort(adapterName='ldap1',
oldHost='oldhost.example.domain.com', oldPort=389,
newHost='newhost.example.domain.com', newPort=389)

4.10.53 removeLDAPHostPort
Online command to remove a remote host and a port from an existing LDAP adapter configuration.

4.10.53.1 Description
This command removes a remote host and a port from an existing LDAP adapter configuration.

4.10.53.2 Syntax
removeLDAPHostPort(adapterName=[adapterName], host=[host], port=[port],
contextName=[contextName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the LDAP adapter to be modified.</td>
</tr>
<tr>
<td>host</td>
<td>Remote LDAP host.</td>
</tr>
<tr>
<td>port</td>
<td>Remote LDAP port.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is default.</td>
</tr>
</tbody>
</table>

4.10.53.3 Example
removeLDAPHostPort(adapterName='ldap1', host='myhost.example.domain.com',
port=389)

4.10.54 setReadOnlyForLDAPHost
Online command to set the given host and port to read-only/writable in an existing LDAP adapter configuration.

4.10.54.1 Description
This command sets the given host and port to read-only/writable in an existing LDAP adapter configuration.
4.10.54.2 Syntax

```
setReadOnlyForLDAPHost(adapterName=[adapterName], host=[host],
port=[port], readOnly=[true/false], contextName=[contextName])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapterName</td>
<td>Name of the LDAP adapter to be modified.</td>
</tr>
<tr>
<td>host</td>
<td>LDAP host.</td>
</tr>
<tr>
<td>port</td>
<td>LDAP port.</td>
</tr>
<tr>
<td>readOnly</td>
<td>It has values: true or false.</td>
</tr>
<tr>
<td>contextName</td>
<td>Optional. Name of the OPSS context with which libOVD configuration is associated. The default value is default.</td>
</tr>
</tbody>
</table>

4.10.54.3 Example

```
setReadOnlyForLDAPHost(adapterName='ldap1',
host='myhost.example.domain.com', port=389, readOnly=true)
```

4.11 Identity Directory Service Commands

Use the WLST commands listed in Table 4–17 to manage Identity Directory Service entity attributes, entity definitions, relationships and default operational configurations.

<table>
<thead>
<tr>
<th>Table 4–17 WLST Identity Directory Service Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use this command...</td>
</tr>
<tr>
<td>activateIDSConfigChanges</td>
</tr>
<tr>
<td>addAttributeInEntityConfig</td>
</tr>
<tr>
<td>addAttributePropsInEntityConfig</td>
</tr>
<tr>
<td>addAttributeRefForEntity</td>
</tr>
<tr>
<td>addAttrrefPropsInEntityConfig</td>
</tr>
<tr>
<td>addCommonPropertyForOperationConfig</td>
</tr>
<tr>
<td>addEntity</td>
</tr>
<tr>
<td>addEntityProps</td>
</tr>
<tr>
<td>addEntityRelation</td>
</tr>
<tr>
<td>addIdentityDirectoryService</td>
</tr>
<tr>
<td>addOperationConfig</td>
</tr>
<tr>
<td>addPropertyForOperationConfig</td>
</tr>
</tbody>
</table>
### Table 4–17  (Cont.) WLST Identity Directory Service Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>deleteAttributeInEntityConfig</td>
<td>Delete an attribute from an entity configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAttrrefPropsInEntityConfig</td>
<td>Delete attribute reference properties in an entity configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteEntity</td>
<td>Delete an entity from an entity configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteEntityProps</td>
<td>Delete entity properties in an entity configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteEntityRelation</td>
<td>Delete the specified entity relation.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteIdentityDirectoryService</td>
<td>Delete the specified Identity Directory Service in the configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteOperationConfig</td>
<td>Delete operation configuration in an entity configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>listAllAttributeInEntityConfig</td>
<td>List all attributes in the entity configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>listAllEntityInEntityConfig</td>
<td>List all entities defined in the specified entity configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>listAllIdentityDirectoryService</td>
<td>List all Identity Directory Services in the configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>removeAttributeRefForEntity</td>
<td>Remove an attribute from the specified entity.</td>
<td>Online</td>
</tr>
<tr>
<td>removeCommonPropertyForOperationConfig</td>
<td>Removes a property for the specified operation configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>removePropertyForOperationConfig</td>
<td>Remove a property for the specified operation configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>updateAttributeInEntityConfig</td>
<td>Update attributes in an entity configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>updateAttributePropsInEntityConfig</td>
<td>Update attribute properties in an entity configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>updateAttrrefPropsInEntityConfig</td>
<td>Update attribute reference properties in an entity configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>updateEntity</td>
<td>Update an entity’s properties in an entity configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>updateEntityAttrs</td>
<td>Update an entity’s attributes in an entity configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>updateEntityProps</td>
<td>Update the entity properties in an entity configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAttributePropsInEntityConfig</td>
<td>Delete attribute properties in an entity configuration.</td>
<td>Online</td>
</tr>
</tbody>
</table>

### 4.11.1 activateIDSConfigChanges

Reloads the Identity Directory Service configuration.

#### 4.11.1.1 Description

This command reloads the Identity Directory Service configuration.

#### 4.11.1.2 Syntax

activateIDSConfigChanges()
This command has no arguments.

**4.11.3 Example**

`activateIDSConfigChanges()`

### 4.11.2 addAttributeInEntityConfig

Adds a new attribute to the entity configuration.

#### 4.11.2.1 Description

This command adds a new attribute to the entity configuration.

#### 4.11.2.2 Syntax

`addAttributeInEntityConfig(name, datatype, description, readOnly, pwdAttr, appName)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the attribute to be added.</td>
</tr>
<tr>
<td>datatype</td>
<td>The attribute’s type is defined as one of the following:</td>
</tr>
<tr>
<td></td>
<td>■ binary</td>
</tr>
<tr>
<td></td>
<td>■ boolean</td>
</tr>
<tr>
<td></td>
<td>■ datetime</td>
</tr>
<tr>
<td></td>
<td>■ double</td>
</tr>
<tr>
<td></td>
<td>■ integer</td>
</tr>
<tr>
<td></td>
<td>■ rfc822name</td>
</tr>
<tr>
<td></td>
<td>■ string</td>
</tr>
<tr>
<td></td>
<td>■ x500name</td>
</tr>
<tr>
<td>description</td>
<td>Description of the attribute to be added.</td>
</tr>
<tr>
<td>readOnly</td>
<td>Flag to specify whether the attribute is read only or can be modified.</td>
</tr>
<tr>
<td>pwdAttr</td>
<td>Flag to specify whether the attribute defines a password or not.</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

#### 4.11.3 Example

`addAttributeInEntityConfig('commonname','string','common name',false,false,'userrole')`

### 4.11.3 addAttributePropsInEntityConfig

Adds new properties for an attribute in an entity configuration.

#### 4.11.3.1 Description

This command adds new properties for an attribute in an entity configuration.

#### 4.11.3.2 Syntax

`addAttributePropsInEntityConfig(name, propNames, propVals, appName)`
### 4.11.3.3 Example

```plaintext
addAttributePropsInEntityConfig('orgunit','labelname|multivalued','common
tname|true','userrole')
```

### 4.11.4 addAttributeRefForEntity

Adds a new attribute to the specified entity.

#### 4.11.4.1 Description

This command adds a new attribute to the specified entity.

#### 4.11.4.2 Syntax

```plaintext
addAttributeRefForEntity(name, attrRefName, attrRefFilter,
attrRefDefaultFetch, appName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the entity to which the attribute will be added.</td>
</tr>
<tr>
<td>attrRefName</td>
<td>Name of the attribute to be added to the entity.</td>
</tr>
<tr>
<td>attrRefFilter</td>
<td>Type of filter to be used with the attribute, defined as one of the following:</td>
</tr>
<tr>
<td></td>
<td>- beginswith</td>
</tr>
<tr>
<td></td>
<td>- contains</td>
</tr>
<tr>
<td></td>
<td>- doesnotcontain</td>
</tr>
<tr>
<td></td>
<td>- dynamic</td>
</tr>
<tr>
<td></td>
<td>- endswith</td>
</tr>
<tr>
<td></td>
<td>- equals</td>
</tr>
<tr>
<td></td>
<td>- greaterequal</td>
</tr>
<tr>
<td></td>
<td>- greaterthan</td>
</tr>
<tr>
<td></td>
<td>- lessequal</td>
</tr>
<tr>
<td></td>
<td>- less-than</td>
</tr>
<tr>
<td></td>
<td>- none</td>
</tr>
<tr>
<td></td>
<td>- notequals</td>
</tr>
</tbody>
</table>

The properties (`propNames` and `propVals`) are free key/value pairs. Applications can store any required metadata at the attribute level in these properties. The Identity Directory Service does not perform any validation for these property names and does not interpret or use these properties internally.

For configuration attributes, however, the Identity Directory Service performs a schema check and interprets the configuration names and their values.
**4.11.4.3 Example**

addAttributeRefForEntity('User','givenname','none','true','userrole')

### 4.11.5 `addAttrrefPropsInEntityConfig`

Adds new properties for an attribute reference in an entity configuration.

#### 4.11.5.1 Description

This command adds new properties for an attribute reference in an entity configuration.

#### 4.11.5.2 Syntax

```plaintext
addAttrrefPropsInEntityConfig(entityName, attrName, propNames, propVals, appName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>entityName</td>
<td>Name of the entity.</td>
</tr>
<tr>
<td>attrName</td>
<td>Name of the attribute reference.</td>
</tr>
<tr>
<td>propNames</td>
<td>List of property names separated by &quot;</td>
</tr>
<tr>
<td>propVals</td>
<td>List of corresponding property values separated by &quot;</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

The properties (propNames and propVals) are free key/value pairs. Applications can store any required metadata at the attribute level in these properties. The Identity Directory Service does not perform any validation for these property names and does not interpret or use these properties internally.

For configuration attributes, however, the Identity Directory Service performs a schema check and interprets the configuration names and their values.

#### 4.11.5.3 Example

addAttrrefPropsInEntityConfig('org', 'orgunit','labelname|multivalued','common name|true','userrole')

### 4.11.6 `addCommonPropertyForOperationConfig`

Adds a new property for a specified operation configuration.

#### 4.11.6.1 Description

This command adds a new property for a specified operation configuration.

#### 4.11.6.2 Syntax

```plaintext
addCommonPropertyForOperationConfig(entityName, propName, propValue, appName)
```
4.11.6.3 Example
addCommonPropertyForOperationConfig('groupmember.attr', 'member', 'userrole')

4.11.7 addEntity
Adds a new entity to the entity configuration.

4.11.7.1 Description
This command adds a new entity to the entity configuration.

4.11.7.2 Syntax
addEntity(name, type, idAttr, create, modify, delete, search, attrRefNames, attrRefFilters, attrRefDefaultFetches, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the entity to which the attribute will be added.</td>
</tr>
<tr>
<td>type</td>
<td>Name of the attribute to be added to the entity.</td>
</tr>
<tr>
<td>idAttr</td>
<td>Identity attribute of the entity to be added.</td>
</tr>
<tr>
<td>create</td>
<td>Flag to specify the create is allowed.</td>
</tr>
<tr>
<td>modify</td>
<td>Flag to specify the modify is allowed.</td>
</tr>
<tr>
<td>delete</td>
<td>Flag to specify the delete is allowed.</td>
</tr>
<tr>
<td>search</td>
<td>Flag to specify the search is allowed.</td>
</tr>
<tr>
<td>attrRefNames</td>
<td>Array of attribute names.</td>
</tr>
<tr>
<td>attrRefFilters</td>
<td>An array of filter type values, defined as one of the following:</td>
</tr>
<tr>
<td></td>
<td>- beginswith</td>
</tr>
<tr>
<td></td>
<td>- contains</td>
</tr>
<tr>
<td></td>
<td>- doesnotcontain</td>
</tr>
<tr>
<td></td>
<td>- dynamic</td>
</tr>
<tr>
<td></td>
<td>- endswith</td>
</tr>
<tr>
<td></td>
<td>- equals</td>
</tr>
<tr>
<td></td>
<td>- greaterequal</td>
</tr>
<tr>
<td></td>
<td>- greaterthan</td>
</tr>
<tr>
<td></td>
<td>- lessequal</td>
</tr>
<tr>
<td></td>
<td>- lessthan</td>
</tr>
<tr>
<td></td>
<td>- none</td>
</tr>
<tr>
<td></td>
<td>- notequals</td>
</tr>
</tbody>
</table>
4.11.7.3 Example
addEntity('Group','group','commonname',true,true,true,true,'name|commonname','none|none','true|false','userrole')

4.11.8 addEntityProps
Adds new properties for an entity in an entity configuration.

4.11.8.1 Description
This command adds new properties for an entity in an entity configuration.

4.11.8.2 Syntax
addEntityProps(name, propNames, propVals, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the entity.</td>
</tr>
<tr>
<td>propNames</td>
<td>List of property names separated by &quot;</td>
</tr>
<tr>
<td>propValues</td>
<td>List of corresponding property values separated by &quot;</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

4.11.8.3 Example
addEntityProps('User','inclobjclasses|exclobjclasses','inetorgperson|orclidxperson','userrole')

4.11.9 addEntityRelation
Add a new entity relation to the entity configuration.

4.11.9.1 Description
This command adds a new entity relation to the entity configuration.

4.11.9.2 Syntax
addEntityRelation(name, type, fromEntity, fromAttr, toEntity, toAttr, recursive, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the relation between the entities for the given attributes.</td>
</tr>
<tr>
<td>type</td>
<td>Type of the entity relation (&quot;ManyToMany&quot;, &quot;ManyToOne&quot;, &quot;OneToMany&quot;, &quot;OneToOne&quot;).</td>
</tr>
<tr>
<td>fromEntity</td>
<td>Name of the from entity.</td>
</tr>
<tr>
<td>fromAttr</td>
<td>Name of the from attribute.</td>
</tr>
<tr>
<td>toEntity</td>
<td>Name of the to entity.</td>
</tr>
</tbody>
</table>
4.11.9.3 Example
addEntityRelation('manager', 'ManyToOne', 'User', 'manager', 'User',
'principal', false, 'userrole')

4.11.10 addIdentityDirectoryService
Adds a new IdentityStoreService to the Identity Directory Service configuration.

4.11.10.1 Description
This command adds a new IdentityStoreService to the Identity Directory Service configuration.

4.11.10.2 Syntax
addIdentityDirectoryService(name, description, propNames, propValues)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the IdentityStoreService to be added.</td>
</tr>
<tr>
<td>description</td>
<td>Description of the IdentityStoreService.</td>
</tr>
<tr>
<td>propNames</td>
<td>An array of property names to be added to the IdentityStoreService configuration.</td>
</tr>
<tr>
<td>propValues</td>
<td>An array of values to be defined for the property names added to the IdentityStoreService configuration.</td>
</tr>
</tbody>
</table>

4.11.10.3 Example
addIdentityDirectoryService('userrole', 'user role',
'ovd.context|entity.config', 'default|userrole')

4.11.11 addOperationConfig
Adds a new operation configuration to the entity configuration.

4.11.11.1 Description
This command adds a new operation configuration to the entity configuration.

4.11.11.2 Syntax
addOperationConfig(entityName, propNames, propValues, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>entityName</td>
<td>Name of the entity to which the operation configuration will be added.</td>
</tr>
<tr>
<td>propNames</td>
<td>An array of property names to be added to the operation configuration.</td>
</tr>
</tbody>
</table>
4.11.3 Example
addOperationConfig('User', 'entity.searchbase', 'cn=users,dc=oracle,dc=com', 'userrole')

4.11.12 addPropertyForOperationConfig

Add a new property to a specified operation configuration.

4.11.12.1 Description
This command adds a new property to a specified operation configuration.

4.11.12.2 Syntax
addPropertyForOperationConfig(entityName, propName, propValue, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>entityType</td>
<td>Name of the entity to which the operation configuration will be added.</td>
</tr>
<tr>
<td>propName</td>
<td>A property name to be added to the operation configuration.</td>
</tr>
<tr>
<td>propValue</td>
<td>A value for the property added to the operation configuration.</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

4.11.12.3 Example
addPropertyForOperationConfig('User', 'entity.searchbase', 'cn=users,dc=oracle,dc=com', 'userrole')

4.11.13 deleteAttributeInEntityConfig

Deletes an attribute from an entity configuration.

4.11.13.1 Description
This command deletes an attribute from an entity configuration.

4.11.13.2 Syntax
deleteAttributeInEntityConfig(name, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the attribute to be deleted.</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

4.11.13.3 Example
deleteAttributeInEntityConfig('commonname', 'userrole')
4.11.14 deleteAttributePropsInEntityConfig

Deletes attribute properties in an entity configuration.

4.11.14.1 Description
This command deletes attribute properties in an entity configuration.

4.11.14.2 Syntax
deleteAttributePropsInEntityConfig(name, propNames, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the attribute.</td>
</tr>
<tr>
<td>propNames</td>
<td>List of property names separated by &quot;</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

4.11.14.3 Example
deleteAttributePropsInEntityConfig('orgunit','labelname|multivalued','user role')

4.11.15 deleteAttrrefPropsInEntityConfig

Deletes attribute reference properties in an entity configuration.

4.11.15.1 Description
This command deletes attribute reference properties in an entity configuration.

4.11.15.2 Syntax
deleteAttrrefPropsInEntityConfig(entityName, attrName, propNames, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>entityName</td>
<td>Name of the entity.</td>
</tr>
<tr>
<td>attrName</td>
<td>Name of the attribute reference.</td>
</tr>
<tr>
<td>propNames</td>
<td>List of property names to be deleted. If multiple properties are to be deleted, they should be separated by &quot;</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

4.11.15.3 Example
deleteAttrrefPropsInEntityConfig('org', 'orgunit','labelname|multivalued','userrole')

4.11.16 deleteEntity

Deletes an entity from an entity configuration.

4.11.16.1 Description
This command deletes an entity from an entity configuration.
4.11.16.2 Syntax
deleteEntity(name, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the entity to be deleted.</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

4.11.16.3 Example
deleteEntity('User', 'userrole')

4.11.17 deleteEntityProps
Deletes entity properties in an entity configuration.

4.11.17.1 Description
This command deletes entity properties in an entity configuration.

4.11.17.2 Syntax
deleteEntityProps(name, propNames, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the entity.</td>
</tr>
<tr>
<td>propNames</td>
<td>List of property names separated by &quot;</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

4.11.17.3 Example
deleteEntityProps('User','inclobjclasses|exclobjclasses','userrole')

4.11.18 deleteEntityRelation
Deletes the specified entity relation.

4.11.18.1 Description
This command deletes the specified entity relation.

4.11.18.2 Syntax
deleteEntityRelation(name, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the relation between the entities for the given attributes.</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

4.11.18.3 Example
deleteEntityRelation('manager', 'userrole')
4.11.19 deleteIdentityDirectoryService

Deletes the specified IdentityStoreService in the Identity Directory Service configuration.

4.11.19.1 Description
This command deletes the specified IdentityStoreService in the Identity Directory Service configuration.

4.11.19.2 Syntax
deleteIdentityDirectoryService(name)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the IdentityStoreService configuration to be deleted.</td>
</tr>
</tbody>
</table>

4.11.19.3 Example
deleteIdentityDirectoryService('ids1')

4.11.20 deleteOperationConfig

Deletes an operation configuration in an entity configuration.

4.11.20.1 Description
This command deletes an operation configuration in an entity configuration.

4.11.20.2 Syntax
deleteOperationConfig(entityName, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>entityName</td>
<td>Name of the entity from which the operation configuration will be removed.</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

4.11.20.3 Example
deleteOperationConfig('User','userrole')

4.11.21 listAllAttributeInEntityConfig

Lists all attributes in the entity configuration.

4.11.21.1 Description
This command lists all attributes in the entity configuration.

4.11.21.2 Syntax
listAllAttributeInEntityConfig(appName)
### 4.11.21.3 Example

```
listAllAttributeInEntityConfig('userrole')
```

### 4.11.22 listAllEntityInEntityConfig

Lists all entities defined in the specified entity configuration.

#### 4.11.22.1 Description

This command lists all entities defined in the specified entity configuration.

#### 4.11.22.2 Syntax

```
listAllEntityInEntityConfig(appName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service that contains the entity configuration from which the list of attributes is retrieved.</td>
</tr>
</tbody>
</table>

#### 4.11.22.3 Example

```
listAllEntityInEntityConfig('userrole')
```

### 4.11.23 listAllIdentityDirectoryService

Lists all IdentityStoreService in Identity Directory Service configuration.

#### 4.11.23.1 Description

This command lists all IdentityStoreService in Identity Directory Service configuration.

#### 4.11.23.2 Syntax

```
listAllIdentityDirectoryService()
```

This command has no arguments.

#### 4.11.23.3 Example

```
listAllIdentityDirectoryService()
```

### 4.11.24 removeAttributeRefForEntity

Removes an attribute from the specified entity.

#### 4.11.24.1 Description

This command removes an attribute from the specified entity.

#### 4.11.24.2 Syntax

```
removeAttributeRefForEntity(name, attrRefName, appName)
```
4.11.24.3 Example
removeAttributeRefForEntity('User','givenname','userrole')

4.11.25 removeCommonPropertyForOperationConfig
Removes a property for the specified operation configuration.

4.11.25.1 Description
This command removes a property for the specified operation configuration.

4.11.25.2 Syntax
removeCommonPropertyForOperationConfig(entityName, propName, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>entityName</td>
<td>Name of the entity.</td>
</tr>
<tr>
<td>propName</td>
<td>Name of property to be removed for this operation configuration.</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

4.11.25.3 Example
removeCommonPropertyForOperationConfig('groupmember.attr','userrole')

4.11.26 removePropertyForOperationConfig
Removes a property for the specified operation configuration.

4.11.26.1 Description
This command removes a property for the specified operation configuration.

4.11.26.2 Syntax
removePropertyForOperationConfig(entityName, propName, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>entityName</td>
<td>Name of the entity to which the operation configuration will be added.</td>
</tr>
<tr>
<td>propName</td>
<td>A property name to be added to the operation configuration.</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

4.11.26.3 Example
removePropertyForOperationConfig('User','entity.searchbase','userrole')
4.11.27 **updateAttributeInEntityConfig**

Updates attributes in an entity configuration.

**4.11.27.1 Description**
This command updates attributes in an entity configuration.

**4.11.27.2 Syntax**

updateAttributeInEntityConfig(name, attrNames, attrVals, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the entity attribute to be updated.</td>
</tr>
<tr>
<td>attrNames</td>
<td>List of configuration attribute names separated by &quot;</td>
</tr>
<tr>
<td></td>
<td>■ dataType</td>
</tr>
<tr>
<td></td>
<td>■ description</td>
</tr>
<tr>
<td></td>
<td>■ readOnly</td>
</tr>
<tr>
<td></td>
<td>■ pwdAttr</td>
</tr>
<tr>
<td></td>
<td>■ attrInUse</td>
</tr>
<tr>
<td>attrVals</td>
<td>List of corresponding attribute values separated by &quot;</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

**4.11.27.3 Example**

updateAttributeInEntityConfig('commonname','readOnly|pwdAttr|attrInUse','true|false|false','userrole')

4.11.28 **updateAttributePropsInEntityConfig**

Updates attribute properties in an entity configuration.

**4.11.28.1 Description**
This command updates attribute properties in an entity configuration.

**4.11.28.2 Syntax**

updateAttributePropsInEntityConfig(name, propNames, propVals, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the attribute to be updated.</td>
</tr>
<tr>
<td>propNames</td>
<td>List of property names separated by &quot;</td>
</tr>
<tr>
<td>propVals</td>
<td>List of corresponding property values separated by &quot;</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

**4.11.28.3 Example**

updateAttributePropsInEntityConfig('orgunit','multivalued','multivalued','userrole')
4.11.29  **updateAttrrefPropsInEntityConfig**

Updates attribute reference properties in an entity configuration.

4.11.29.1  **Description**

This command updates attribute reference properties in an entity configuration.

4.11.29.2  **Syntax**

updateAttrrefPropsInEntityConfig(entityName, attrName, propNames, propVals, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>entityName</td>
<td>Name of the entity.</td>
</tr>
<tr>
<td>attrName</td>
<td>Name of the attribute reference.</td>
</tr>
<tr>
<td>propNames</td>
<td>List of property names separated by &quot;</td>
</tr>
<tr>
<td>propVals</td>
<td>List of corresponding property values separated by &quot;</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

4.11.29.3  **Example**

updateAttrrefPropsInEntityConfig('org', 'orgunit', 'multivalued', 'multivalued', 'userrole')

4.11.30  **updateEntity**

Updates an entity's properties in an entity configuration.

4.11.30.1  **Description**

This command updates the properties of an entity in an entity configuration.

4.11.30.2  **Syntax**

updateEntity(name, type, idAttr, create, modify, delete, search, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the entity to be updated.</td>
</tr>
<tr>
<td>type</td>
<td>Type of the entity.</td>
</tr>
<tr>
<td>idAttr</td>
<td>Identity attribute of the entity.</td>
</tr>
<tr>
<td>create</td>
<td>Flag to specify the create is allowed.</td>
</tr>
<tr>
<td>modify</td>
<td>Flag to specify the modify is allowed.</td>
</tr>
<tr>
<td>delete</td>
<td>Flag to specify the delete is allowed.</td>
</tr>
<tr>
<td>search</td>
<td>Flag to specify the search is allowed.</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

4.11.30.3  **Example**

updateEntity('Group', 'group', 'commonname', true, true, true, true, 'userrole')
### 4.11.31 updateEntityAttrs

Updates the configuration attributes for an entity attribute.

#### 4.11.31.1 Description

This command updates the configuration attributes for an entity attribute.

#### 4.11.31.2 Syntax

updateEntityAttrs(name, attrNames, attrVals, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the entity attribute.</td>
</tr>
<tr>
<td>attrNames</td>
<td>List of configuration attribute names. If multiple configuration attributes are to be updated, they should be separated by &quot;</td>
</tr>
<tr>
<td></td>
<td>idAttr</td>
</tr>
<tr>
<td></td>
<td>pwdAttr</td>
</tr>
<tr>
<td></td>
<td>firstnameAttr</td>
</tr>
<tr>
<td></td>
<td>lastnameAttr</td>
</tr>
<tr>
<td></td>
<td>mailAttr</td>
</tr>
<tr>
<td></td>
<td>displaynameAttr</td>
</tr>
<tr>
<td></td>
<td>descriptionAttr</td>
</tr>
<tr>
<td></td>
<td>challengeQnAttr</td>
</tr>
<tr>
<td></td>
<td>challengeAnsAttr</td>
</tr>
<tr>
<td></td>
<td>commonIdAttr</td>
</tr>
<tr>
<td>attrVals</td>
<td>List of corresponding configuration attribute values separated by &quot;</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

#### 4.11.31.3 Example

updateEntityAttrs('User','idAttr|firstnameAttr','uid|givenname','userrole')

### 4.11.32 updateEntityProps

Updates the entity properties in an entity configuration.

#### 4.11.32.1 Description

This command updates the entity properties in an entity configuration.

#### 4.11.32.2 Syntax

updateEntityProps(name, propNames, propVals, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the attribute to be added.</td>
</tr>
<tr>
<td>propNames</td>
<td>List of property names separated by &quot;</td>
</tr>
</tbody>
</table>
4.11.32.3 Example
updateEntityProps('User','inclobjclasses|exclobjclasses','inetorgperson|or
cldxperson','userrole')

4.11.33 deleteAttributePropsInEntityConfig

Deletes the attribute properties in an entity configuration.

4.11.33.1 Description
This command deletes the attribute properties in an entity configuration.

4.11.33.2 Syntax
deleteAttributePropsInEntityConfig(name, propNames, appName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the attribute to be deleted.</td>
</tr>
<tr>
<td>propNames</td>
<td>List of property names separated by &quot;</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the Identity Directory Service.</td>
</tr>
</tbody>
</table>

4.11.33.3 Example
deleteAttributePropsInEntityConfig('orgunit','labelname|validvalues','userrule')
This chapter provides detailed descriptions of custom WLST commands for UMS, including command syntax, arguments and command examples.

Use the User Messaging Service commands, listed in Table 5–1, to download user messaging preferences from your backend database.

Note: To use these commands, you must invoke WLST from the Oracle home in which the component has been installed. See "Using Custom WLST Commands" in the Oracle Fusion Middleware Administrator’s Guide.

<table>
<thead>
<tr>
<th>Table 5–1 User Messaging Service for WLST Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command category</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>Section 5.1, &quot;UMS WLST Command Group&quot;</td>
</tr>
</tbody>
</table>

5.1 UMS WLST Command Group

The UMS WLST commands are listed under the command group "ums".

5.1.1 manageUserMessagingPrefs

Command Category: UMS
Use with WLST: Offline

5.1.1.1 Description

manageUserMessagingPrefs is used to download the user messaging preferences from a backend database to the specified XML file, or to upload the user messaging preferences from an XML file into the backend database.

5.1.1.2 Syntax

manageUserMessagingPrefs {operation=, filename, url, username, password, [encoding], [guid], [merge] }
5.1.1.3 Examples

To download the user messaging preferences of all users to the specified file.

```
  wls:offline> manageUserMessagingPrefs(operation='download',
                filename='download.xml', url='t3://localhost:8001', username='weblogic',
                password='<password>')
```

To download the user messaging preferences of all users to the specified file using UTF-8 character encoding.

```
  wls:offline> manageUserMessagingPrefs(operation='download',
                filename='download.xml', url='t3://localhost:8001', username='weblogic',
                password='<password>', encoding='UTF-8')
```

To download the user messaging preferences of the user with guid 'john.doe' to the specified file.

```
  wls:offline> manageUserMessagingPrefs(operation='download',
                filename='download.xml', url='t3://localhost:8001', username='weblogic',
                password='<password>', guid='john.doe')
```

To download the user messaging preferences of the users with guid 'john.doe' and 'jane.doe' to the specified file using UTF-8 character encoding.

```
  wls:offline> manageUserMessagingPrefs(operation='download',
                filename='download.xml', url='t3://localhost:8001', username='weblogic',
                password='<password>', guid='john.doe,jane.doe', encoding='UTF-8')
```

To upload the user messaging preferences from the specified file to the backend database.

```
  wls:offline> manageUserMessagingPrefs(operation='upload',
                filename='upload.xml', url='t3://localhost:8001', username='weblogic',
                password='<password>', merge='create_new')
```

### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>operation</td>
<td>specifies the upload or download operation to be performed.</td>
</tr>
<tr>
<td>filename</td>
<td>For download, a unique file name (path) to download the user preferences to. For example, /tmp/download.xml (Linux) or C:\temp\download.xml (Windows). For upload, the file name (path) from which to upload the user preferences.</td>
</tr>
<tr>
<td>url</td>
<td>The JNDI URL to access the User Messaging Server. For example: t3://&lt;hostname&gt;:&lt;port&gt;</td>
</tr>
<tr>
<td>username</td>
<td>The username with login permission to access the User Messaging Server.</td>
</tr>
<tr>
<td>password</td>
<td>The password of the username.</td>
</tr>
<tr>
<td>encoding</td>
<td>Character encoding to use to download the user preferences.</td>
</tr>
<tr>
<td>guid</td>
<td>The globally unique identifier (guid) of a list of users to use to download their preferences. If no guid is specified, the preferences for all users are downloaded.</td>
</tr>
<tr>
<td>merge</td>
<td>This option is for upload only. Valid values are: create_new (default): Create new user device, device addresses and/or ruleset entities. An exception will be thrown if an entity with the same primary key already exists and processing will terminate. overwrite: Remove all existing entities of a user and then create new entities. append: Only upload entities that do not already exist.</td>
</tr>
</tbody>
</table>
manageUserMessagingPrefs(operation='upload', filename='upload.xml', url='t3://localhost:8001', username='weblogic', password='<password>')

To upload the user messaging preferences from the specified file to the backend database and overwrite existing preferences.

manageUserMessagingPrefs(operation='upload', filename='upload.xml', url='t3://localhost:8001', username='weblogic', password='<password>', merge='overwrite')

5.1.2 deployUserMessagingDriver

Command Category: UMS

Use with WLST: Online

5.1.2.1 Description

deployUserMessagingDriver is used to deploy additional instances of user messaging drivers.

Specify a base driver type (for example: email, xmpp, voicexml, and others) and a short name for the new driver deployment. The string usermessagingdriver- will be prepended to the specified application name. Any valid parameters for the deploy command can be specified, and will be passed through when the driver is deployed.

5.1.2.2 Syntax

deployUserMessagingDriver(baseDriver, appName, [targets], [stageMode], [options])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseDriver</td>
<td>Specifies the base messaging driver type. Must be a known driver type, such as 'email', 'proxy', 'smpp', 'voicexml', or 'xmpp'.</td>
</tr>
<tr>
<td>appName</td>
<td>A short descriptive name for the new deployment. The specified value will be prepended with the string usermessagingdriver-</td>
</tr>
<tr>
<td>targets</td>
<td>Optional. Additional arguments that are valid for the deploy command can be specified and will be passed through when the new driver is deployed.</td>
</tr>
<tr>
<td>stageMode</td>
<td></td>
</tr>
<tr>
<td>options</td>
<td></td>
</tr>
</tbody>
</table>

5.1.2.3 Examples

To deploy a second instance of an email driver with name myEmail.

wls:base_domain/servereConfig> deployUserMessagingDriver(baseDriver='email', appName='myEmail')

To deploy a second instance of an email driver, specifying deployment targets.

wls:base_domain/servereConfig> deployUserMessagingDriver(baseDriver='email', appName='email2', targets='server1,server2')
This chapter provides detailed descriptions of custom WLST commands for Dynamic Monitoring Service (DMS), which you can use to view performance metrics and to configure Event Tracing. This chapter includes command syntax, arguments and command examples.

**Note:** To use these DMS custom WLST commands, you must invoke the WLST script from the Oracle Common home. See "Using Custom WLST Commands" in the *Oracle Fusion Middleware Administrator’s Guide*.

Use the DMS commands in the categories in Table 6–1 to view performance metrics and to configure Event Tracing.

<table>
<thead>
<tr>
<th>Command category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMS Metric Commands</td>
<td>View information about performance metrics.</td>
</tr>
<tr>
<td>DMS Event Tracing Commands</td>
<td>Configure Event Tracing</td>
</tr>
</tbody>
</table>

### 6.1 DMS Metric Commands

Use the commands in Table 6–2 to view information about a specific performance metric, a set of performance metrics, or all performance metrics for a particular server or component.

For additional details about metrics, see the chapter "Monitoring Oracle Fusion Middleware" in the *Oracle Fusion Middleware Administrator’s Guide* and the appendix "Instrumenting Applications with DMS" in the *Oracle Fusion Middleware Performance and Tuning Guide*.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>displayMetricTableNames</td>
<td>Displays the names of the available DMS metric tables.</td>
<td>Online</td>
</tr>
<tr>
<td>displayMetricTables</td>
<td>Displays the content of the DMS metric tables.</td>
<td>Online</td>
</tr>
<tr>
<td>dumpMetrics</td>
<td>Displays available metrics.</td>
<td>Online</td>
</tr>
<tr>
<td>reloadMetricRules</td>
<td>Reloads the metric rules.</td>
<td>Online</td>
</tr>
</tbody>
</table>
6.1.1 displayMetricTableNames

Command Category: DMS Metrics

Use with WLST: Online

6.1.1.1 Description
Displays the names of the available DMS metric tables. The returned value is a list of metric table names.

6.1.1.2 Syntax
displayMetricTableNames([servers])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>servers</td>
<td>Optional. Specifies the servers from which to retrieve metrics. Valid values are a list of WebLogic Server instance names and system component names. To specify one server, use the following syntax: servers=‘servername’ To specify multiple servers, use one of the following syntax options: servers=['servername1', 'servername2', ...] servers=('servername1', 'servername2', ...) If this argument is not specified, the command returns the list of metric table names for all WebLogic servers and system components. Note that the name of a system component instance is not a fixed string. It is the UID of the instance. The UID changes at each instance restart. The UID uses the format: /install_type/component_type/UID To find the UID, use the following OPMN command: opmnctl status -l</td>
</tr>
</tbody>
</table>

6.1.1.3 Examples
The following example displays metric table names for all WebLogic servers and system components:

    displayMetricTableNames()
    ADF
    ADFc
    ADFc_Metadata_Service
    ADFcRegion
    ADFc_Taskflow
    ADFc_Viewport
    BAM_common_connectionpool
    BAM_common_connectionpool_main
    BAM_common_messaging
    BAM_common_messaging_consumers
    .
    .
    .

The following example displays metric table names for the WebLogic Managed Server soa_server1:

    displayMetricTableNames(servers='soa_server1')
    ADF
The following example displays metric table names for two WebLogic Managed Servers:

```python
displayMetricTableNames(servers=['soa_server1', 'bam-server1'])
```

ADF
ADFc
ADFc_Metadata_Service
ADFc_Region
ADFc_Taskflow
ADFc_Viewport
BAM_common_connectionpool
BAM_common_connectionpool_main
BAM_common_messaging
BAM_common_messaging_consumers
...
...
...

The following example displays the metric table names for Oracle Web Cache. Note that the name of a system component instance is not a fixed string. It includes the UID of the instance, which changes at each instance restart. To find the UID, use the following OPMN command:

```bash
opmnctl status -l
```

Then, use the display MetricTableNames command. For example:

```python
displayMetricTableNames(servers=('/web_tier/webcache1/123456789'))
```

### 6.1.2 displayMetricTables

Command Category: DMS Metrics

Use with WLST: Online

#### 6.1.2.1 Description

Displays the content of the DMS metric tables.

The returned value is list of DMS metric tables, with the following information about each table:

- The metric table name.
- The metric table schema information.
- The metric table Rows.

The metric table schema information contains the following:

- The name of the column.
The type of the column value.

- The unit of the column.
- The description of the column.

### 6.1.2.2 Syntax

displayMetricTables([metricTable_1] [, metricTable_2], [..] [, servers] [, variables])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>metricTable_n</td>
<td>Optional. Specifies a list of metric tables. By default, this argument displays all available metrics. The metric table name can contain special characters for simple pattern matching. The character '?' matches any single character. The character '*' matches zero or more characters. You specify the metric table name. You can specify multiple metric table names in a comma-separated list. These are the same names output by the WLST command displayMetricTableNames.</td>
</tr>
<tr>
<td>servers</td>
<td>Optional. Specifies the servers from which to retrieve metrics. Valid values are a list of WebLogic Server instance names and system component names. To specify one server, use the following syntax: servers='servername' To specify multiple servers, use one of the following syntax options: servers=['servername1', 'servername2', ...] servers=('servername1', 'servername2', ...) If this argument is not specified, the command returns the list of metric tables for all WebLogic servers and system components. Note that the name of a system component instance is not a fixed string. It is the UID of the instance. The UID changes at each instance restart. The UID uses the format: /install_type/component_type/UID To find the UID, use the following OPMN command: opmnctl status -l</td>
</tr>
<tr>
<td>variables</td>
<td>Optional. Defines the metric aggregation parameters. Valid values are a set of name-value pairs. It uses the following syntax: variables={name1:value1, name2:value2, ...} The specific name-value pairs depend on the aggregated metric tables. Each aggregated metric table has its specific set of variable names.</td>
</tr>
</tbody>
</table>

### 6.1.2.3 Examples

The following example displays the data from the JVM and the weblogic.management.runtime.WebAppComponentRuntimeMBean metric tables, and limits it to data retrieved from soa_server1 and bam_server1:

displayMetricTables('JVM', 'weblogic.management.runtime.WebAppComponentRuntimeMBean', servers=['soa_server1', 'bam_server1'])

```
ApplicationRuntime:  soa-infra
ComponentName:     /integration/services/IdentityService
ContextRoot:        /integration/services/IdentityService
DeploymentState:    2
```
FilterDispatchedRequestsEnabled: false
IndexDirectoryEnabled: false
JSPDebug: false
JSPKeepGenerated: false
JSPPageCheckSecs: 1
JSPVerbose: true
ModuleId: /integration/services/IdentityService
ModuleURI: IdentityService.war
Name: soa_server1_/integration/services/IdentityService
ObjectName: com.bea:ApplicationRuntime=soa-infra,Name=soa_server1_/integration/services/IdentityService,
ServerRuntime=soa_server1,Type=WebAppComponentRuntime
OpenSessionsCurrentCount: 0
OpenSessionsHighCount: 0

The following example displays the aggregated metric tables with the specified metric aggregation parameters:

displayMetricTables('j2ee_application:webservices_port_rollup',
    servers=['soa_server1','bam_server1'],
    variables={'host': 'hostname', 'servletName': 'dms'})
----------------------------------------
j2ee_application:webservices_port_rollup
----------------------------------------
Faults: 0
Requests: 0
Requests.averageTime: 0.0
Requests.totalTime: 0.0
ServerName: soa_server1
moduleName: RuntimeConfigService
moduleType: WEBs
portName: RuntimeConfigServicePortSAML
processRequest.active: 0
service.throughput: 0.0
service.time: 0.0
startTime: 1238182359291
webserviceName: RuntimeConfigService

Faults: 0
Requests: 0
Requests.averageTime: 0.0
Requests.totalTime: 0.0
ServerName: soa_server1
moduleName: TaskMetadataService
moduleType: WEBs
portName: TaskMetadataServicePort
processRequest.active: 0
service.throughput: 0.0
service.time: 0.0
startTime: 1238182358096
webserviceName: TaskMetadataService
The following example displays the metric tables which names match the specified patterns:

displayMetricTables('J??', 'JVM_*')
.
.
-----------------------
JVM_ThreadStats
-----------------------

Host:   hostname.com  
JVM:    JVM            
Name:   threads        
Parent: /JVM/MxBeans  
Process:        AdminServer:9001  
ServerName:     AdminServer
contention.value:       enabled in JVM  
daemon.value:   60      threads  
deadlock.value: 0       threads  
live.value:     61      threads  
peak.value:     66      threads  
started.value:  241     threads  

Host:   hostname.com  
JVM:    JVM            
Name:   threads        
Parent: /JVM/MxBeans  
Process:        soa_server1:9001  
ServerName:     soa_server1
contention.value:       enabled in JVM  
daemon.value:   68      threads  
deadlock.value: 0       threads  
live.value:     74      threads  
peak.value:     74      threads  
started.value:  105     threads  
.
.

6.1.3  dumpMetrics

Command Category: DMS Metrics

Use with WLST: Online

6.1.3.1 Description

Displays available metrics in the internal format or in XML. The returned value is a text document.

6.1.3.2 Syntax

dumpMetrics([servers] [, format])
### 6.1.3.3 Examples

The following example outputs all available metrics, including native WebLogic Server metrics and internal DMS metrics, in the XML format:

```python
dumpMetrics(format='xml')
```

```xml
<table name='weblogic_j2eeserver:jvm' keys='ServerName serverName'
    componentId='bam_server1' cacheable='false'>
  <row cacheable='false'>
    <column name='serverName'><![CDATA[bam_server1]]></column>
    <column name='nurserySize.value' type='DOUBLE'>0.0</column>
    <column name='jdkVersion.value'><![CDATA[1.6.0_05]]></column>
    <column name='jdkVendor.value'><![CDATA[BEA Systems, Inc.]]></column>
    <column name='daemonThreads.active' type='LONG'>68</column>
    <column name='cpuUsage.percentage' type='DOUBLE'>100.0</column>
    <column name='threads.active' type='LONG'>71</column>
    <column name='heapUsed.value' type='DOUBLE'>0.0</column>
  </row>
</table>
```

The following example outputs metrics from Server-0 in the default raw format:

```python
dumpMetrics(servers='Server-0')
```

```text
/JVM/MxBeans/threads/Thread-44 [type=JVM_Thread]
ECID.value:        null
RID.value: null
blocked.value:     0       msec
blockedCount.value:        1       times
cpu.value: 40      msecs
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>servers</code></td>
<td>Optional. Specifies the servers from which to retrieve metrics. Valid values are a list of WebLogic Server instance names and system component names. To specify one server, use the following syntax: <code>servers='servername'</code> To specify multiple servers, use one of the following syntax options: <code>servers=['servername1', 'servername2', ...]</code> <code>servers=('servername1', 'servername2', ...)</code> If this argument is not specified, the command returns the list of metric tables for all WebLogic servers and system components. Note that the name of a system component instance is not a fixed string. It is the UID of the instance. The UID changes at each instance restart. The UID uses the format: <code>//install_type/component_type/UID</code> To find the UID, use the following OPMN command: <code>opmnctl status -l</code></td>
</tr>
<tr>
<td><code>format</code></td>
<td>Optional. Specifies the command output format. Valid values are <code>raw</code> (the default), <code>xml</code>, and <code>pdml</code>. For example: <code>format='raw'</code> <code>format='xml'</code> <code>format='pdml'</code> DMS raw format is a simple metric display format; it displays one metric per line.</td>
</tr>
</tbody>
</table>

DMS Metric Commands

DMS Custom WLST Commands
The following example outputs metrics from soa_server1 and bam_server1 in XML format:

dumpMetrics(servers=['soa_server1', 'bam_server1'], format='xml')

```
<table name='oracle_soainfra:high_latency_sync_composites' keys='ServerName soainfra_composite soainfra_composite_revision soainfra_domain' componentId='bam_server1' cacheable='false'>
</table>
<table name='weblogic_j2eeserver:ejb_transaction' keys='ServerName appName ejbModuleName name serverName' componentId='bam_server1' cacheable='false'>
  <row cacheable='false'>
    <column name='serverName'><![CDATA[bam_server1]]></column>
    <column name='name'><![CDATA[MessagingClientParlayX]]></column>
    <column name='ejbTransactionCommit.percentage' type='DOUBLE'>0.0</column>
    <column name='ejbTransactionRollback.completed' type='LONG'>0</column>
    <column name='ejbTransactionTimeout.throughput' type='DOUBLE'>0.0</column>
    <column name='ejbTransactionCommit.completed' type='LONG'>0</column>
    <column name='ejbTransactionTimeout.completed' type='LONG'>0</column>
    <column name='appName'><![CDATA[ usermessagingserver]]></column>
    <column name='ejbTransactionRollback.throughput' type='DOUBLE'>0.0</column>
    <column name='ServerName'><![CDATA[bam_server1]]></column>
    <column name='ejbModuleName'><![CDATA[sdpmessagingclient-ejb-parlayx.jar]]></column>
  </row>
```

### 6.1.4 reloadMetricRules

**Command Category:** DMS Metrics

**Use with WLST:** Online

#### 6.1.4.1 Description

Reloads the metric rules. You must run this command after you deploy system components or after you modify metric rules. Generally, Oracle does not recommend that you modify metric rules.

#### 6.1.4.2 Syntax

```
reloadMetricRules()
```
6.1.4.3 Example

The following example reloa ds metric rules for all servers running in the domain:

```
reloadMetricRules()
```

Location changed to domainRuntime tree. This is a read-only tree with DomainMBean as the root.

For more help, use help(domainRuntime)

```
loaded 'server-oracle_eps_server-11.0.xml'
loaded 'server-weblogic_j2eeserver-11.0.xml'
loaded 'server-oracle_bamweb-11.0.xml'
loaded 'server-oracle_federation-11.0.xml'
loaded 'server-portal-11.0.xml'
loaded 'server-weblogic_j2ee_application_webcenter-11.0.xml
```

6.2 DMS Event Tracing Commands

Use the commands in Table 6–3 to configure Event Tracing. Event Tracing configures live tracing with no restarts. DMS metrics that were updated using Oracle Fusion Middleware products may be traced using the DMS Event Tracing feature.

For information about using DMS Event Tracing, see "DMS Tracing and Events" in the Oracle Fusion Middleware Performance and Tuning Guide.

Table 6–3  DMS Tracing Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>addDMSEventDestination</td>
<td>Add a new destination to the Event Tracing configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>addDMSEventFilter</td>
<td>Add a filter to the Event Tracing configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>addDMSEventRoute</td>
<td>Adds the specified event route to the Event Tracing configuration</td>
<td>Online</td>
</tr>
<tr>
<td>enableDMSEventTrace</td>
<td>Enable an event trace and create a filter with a specified condition and destination and an enabled event-route.</td>
<td>Online</td>
</tr>
<tr>
<td>listDMSEventConfiguration</td>
<td>Display an overview of the event tracing configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>listDMSEventDestination</td>
<td>Display the full configuration for a destination or a list of all destinations.</td>
<td>Online</td>
</tr>
<tr>
<td>listDMSEventFilter</td>
<td>Displays the configuration of a filter or a list of all filters.</td>
<td>Online</td>
</tr>
<tr>
<td>listDMSEventRoutes</td>
<td>Displays event routes and their status (enabled or disabled).</td>
<td>Online</td>
</tr>
<tr>
<td>removeDMSEventDestination</td>
<td>Removes the specified destination.</td>
<td>Online</td>
</tr>
<tr>
<td>removeDMSEventFilter</td>
<td>Removes the specified filter.</td>
<td>Online</td>
</tr>
<tr>
<td>removeDMSEventRoute</td>
<td>Removes the specified event route.</td>
<td>Online</td>
</tr>
<tr>
<td>updateDMSEventDestination</td>
<td>Updates configuration of an event destination.</td>
<td>Online</td>
</tr>
<tr>
<td>updateDMSEventFilter</td>
<td>Updates the configuration of an event filter.</td>
<td>Online</td>
</tr>
<tr>
<td>updateDMSEventRoute</td>
<td>Updates the configuration of an event route.</td>
<td>Online</td>
</tr>
</tbody>
</table>
6.2.1 addDMSEventDestination

Command Category: DMS Event Tracing
Use with WLST: Online

6.2.1.1 Description

Adds a new destination to the Event Tracing configuration. If a destination with the same ID already exists, the command reports this and does not add the destination. You must be connected to the Administration Server to add a destination. If you are not, an error is returned.

6.2.1.2 Syntax

```
addDMSEventDestination(id=id [, name=dest_name] ,class=class_name
 [, props= {'name': 'value'...}] [,server=server_name])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The unique identifier for the specified destination.</td>
</tr>
<tr>
<td>name</td>
<td>Optional. A name for the destination.</td>
</tr>
<tr>
<td>class</td>
<td>The full class name of the destination. See Table 6-4 for a list of available classes.</td>
</tr>
<tr>
<td>props</td>
<td>Optional. The name/value properties to use for the destination. Some destinations require properties. For example, the LoggerDestination class requires the property loggerName. See addDMSEventFilter for information about the syntax and allowed values.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. The server on which to perform this operation. The default is the server to which you are connected.</td>
</tr>
</tbody>
</table>

Table 6-4 shows the built-in destinations, with the full runtime class name.

<table>
<thead>
<tr>
<th>Runtime Destination Class Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>oracle.dms.trace2.runtime.LoggerDestination</td>
<td>Uses ODL to send the log messages to a file.</td>
</tr>
<tr>
<td>oracle.dms.event.HTTPRequestTrackerDestination</td>
<td>Dumps the set of active HTTP requests, allowing an administrator to get a snapshot of activity.</td>
</tr>
<tr>
<td>oracle.dms.jrockit.jfr.JFRDestination</td>
<td>Passes events to the JRockit Flight Recorder so that they can be viewed in the context of other data coming from the JRockit JVM and WLDF using JRockit Mission Control.</td>
</tr>
<tr>
<td>oracle.dms.jmx.MetricMBeanFactory</td>
<td>Exposes Nouns as MBeans.</td>
</tr>
<tr>
<td>oracle.dms.util.StackTraceCollatorDestination</td>
<td>Collates the stack traces that are in play whenever the events of interest occur. This is primarily a debugging tool. The collated data is written out on shutdown, and also when an event being handled has not been reported for a certain period of time (defaults to one minute).</td>
</tr>
</tbody>
</table>
6.2.1.3 Examples
The following example adds a destination with the ID jfr, the name Flight-Recorder, and the class oracle.dms.event.JRockitFlightRecorder:

```
addDMSEventDestination(id='jfr', name='Flight-Recorder',
                      class='oracle.dms.event.JRockitFlightRecorder')
```

Destination 'jfr' added.

The following example adds a destination with the ID destination1, the name File-system, the class oracle.dms.trace2.runtime.LoggerDestination. Because the LoggerDestination requires the property loggerName, it sets the value to trace2-logger:

```
addDMSEventDestination(id='destination1', name='File-system',
                      class='oracle.dms.trace2.runtime.LoggerDestination',
                      props={'loggerName': 'trace2-logger'})
```

Destination 'destination1' added.

The following example attempts to add a destination with an ID that already exists:

```
addDMSEventDestination(id='destination1', name='File-system',
                      class='oracle.dms.trace2.runtime.LoggerDestination',
                      props={'loggerName': 'trace2-logger'})
```

Destination 'destination1' already exists. Unable to add this.

6.2.2 addDMSEventFilter
Command Category: DMS Event Tracing
Use with WLST: Online

6.2.2.1 Description
Adds a filter to the Event Tracing configuration. If a filter with the same ID already exists, the command returns an error and does not add the filter.

You must be connected to the Administration Server to add an event filter. If you are not, an error message is reported.

6.2.2.2 Syntax
```
addDMSEventFilter{[id=id [, name=name] [, etypes]
                  , props= {'prop-name': 'value'...}
                  [, server=server_name]}
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The unique identifier for specified filter.</td>
</tr>
<tr>
<td>name</td>
<td>Optional. The name of the filter.</td>
</tr>
<tr>
<td>etypes</td>
<td>Optional. A string containing a comma-separated list of event/action pairs. This argument allows you to create a filter with a broader granularity when used with a condition. It also allows you to create a filter with a broader range of metrics. For example, all nouns or all nouns with the action create.</td>
</tr>
<tr>
<td>props</td>
<td><code>prop-name</code>: The name of the filter property. <code>&lt;condition&gt;</code> is the only valid property, and only one condition may be specified. <code>value</code>: The value of the property of the filter.</td>
</tr>
</tbody>
</table>
The following shows the syntax for etypes:

ettes::=
<type>::=[<action>]

The following lists the valid etypes:

EXECUTION_CONTEXT
EXECUTION_CONTEXT:START
EXECUTION_CONTEXT:STOP
HTTP_REQUEST
HTTP_REQUEST:START
HTTP_REQUEST:STOP
NOUN
NOUN:CREATE
NOUN:DELETE
STATE_SENSOR
STATE_SENSOR:CREATE
STATE_SENSOR:DELETE

The following shows an etype with two event/action pairs, separated by a comma:

ettes='NOUN:DELETE, STATE_SENSOR:DELETE'

The following shows the syntax for the <condition> property of the argument props. The arguments are described in the tables following the syntax:

<condition>::=
<type>  [<operator> <condition>]  

<type>::=
<nountype> | <context>

<nountype>::=
NOUNTYPE <nountype-operator> value

<nountype-operator>::=
"equals" | "starts_with" | "contains" | "not_equals"

<context>::=
CONTEXT <name> <context-operator> [value] [IGNORECASE=true|false] [DATATYPE="string|long|double"
]

<context-operator>::=
"equals" | "starts_with" | "contains" | "not_equals" | "is_null" | "gt" | "le" | "ge"

<operator>::=
AND | OR

The following table describes the arguments for <type>:
The following table describes the arguments for `<nountype>`:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOUNTYPE</td>
<td>A keyword.</td>
</tr>
<tr>
<td><code>&lt;nountype-operator&gt;</code></td>
<td>The following are valid operators:</td>
</tr>
<tr>
<td>equals</td>
<td>Filters only if the Noun type name equals the value.</td>
</tr>
<tr>
<td>starts_with</td>
<td>Filters only if the Noun type name starts with the value.</td>
</tr>
<tr>
<td>contains</td>
<td>Filters only if the Noun type name equals the value.</td>
</tr>
<tr>
<td>not_equal</td>
<td>Filters only if the Noun type name does not equal the value.</td>
</tr>
<tr>
<td>value</td>
<td>The name of the Noun type on which to operate.</td>
</tr>
</tbody>
</table>

The following table describes `<context>`

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTEXT</td>
<td>A keyword.</td>
</tr>
<tr>
<td>name</td>
<td>The name of the context to filter.</td>
</tr>
<tr>
<td>value</td>
<td>The name of the context on which to operate.</td>
</tr>
<tr>
<td><code>&lt;context-operator&gt;</code></td>
<td>The following are valid operators:</td>
</tr>
<tr>
<td>equals</td>
<td>Filters only if the context name equals the value.</td>
</tr>
<tr>
<td>starts_with</td>
<td>Filters only if the context name starts with the value.</td>
</tr>
<tr>
<td>contains</td>
<td>Filters only if the context name equals the value.</td>
</tr>
<tr>
<td>not_equal</td>
<td>Filters only if the context name does not equal the value.</td>
</tr>
<tr>
<td>is_null</td>
<td>Filters only if the context name is null.</td>
</tr>
<tr>
<td>lt</td>
<td>Filters only if the context name is less than the value.</td>
</tr>
<tr>
<td>gt</td>
<td>Filters only if the context name is greater than the value.</td>
</tr>
<tr>
<td>le</td>
<td>Filters only if the context name is less than or equal to the value.</td>
</tr>
<tr>
<td>ge</td>
<td>Filters only if the context name is greater than or equal to the value.</td>
</tr>
<tr>
<td>IGNORECASE</td>
<td>Optional. If specified, the case of a string datatype is ignored. The</td>
</tr>
<tr>
<td></td>
<td>default is that the case of a string datatype is checked.</td>
</tr>
</tbody>
</table>

Each Sensor, with its associated metric, is organized in a hierarchy according to Nouns. A Noun type is a name that reflects the set of metrics being collected. For example, JDBC could be a Noun type. For information about Sensors and Nouns, see "Understanding Common DMS Terms and Concepts" in the Oracle Fusion Middleware Performance and Tuning Guide.

An Execution Context is an association of the Execution Context ID (ECID), Relationship ID (RID), and Maps of Values. This argument allows the data stored in the map of values to be inspected and used by the filter. For example, if the map contains the key "user", you can create a filter that returns requests with "user" equal to "bruce".
### 6.2.2.3 Examples

The following example adds a filter with the name MyFilter, specifying a Noun type and context:

```java
addDMSEventFilter(id='mds1', name='MyFilter',
                   props={'condition': 'NOUNTYPE equals MDS_Connections AND CONTEXT user equals bruce IGNORECASE'})
```

Filter "mds1" added.

The following example attempts to add a filter with the same id. The command returns an error:

```java
addDMSEventFilter(id='mds1', name='MyFilter',
                   props={'condition': 'NOUNTYPE equals MDS_Connections AND CONTEXT user equals bruce'})
```

Filter "mds1" already exists. Unable to add this.

The following example adds a filter with two event/action pairs:

```java
addDMSEventFilter(id='mds2', name='MyFilter',
                   etypes='NOUN:CREATE,HTTP_REQUEST:START',
                   props={'condition': 'NOUNTYPE equals MDS_Connections AND CONTEXT user equals bruce IGNORECASE=true'})
```

Filter "mds2" added.

### 6.2.3 addDMSEventRoute

**Command Category:** DMS Event Tracing  
**Use with WLST:** Online

#### 6.2.3.1 Description

Adds the specified event route to the Event Tracing configuration. If an event route with the same ID already exists, the command returns an error and does not add the event route.

You must be connected to the Administration Server to add an event route. If you are not, an error is returned.

#### 6.2.3.2 Syntax

```java
addDMSEventRoute([filterid=filter_id], destinationid=destination_id,
                 [enable=true|false] [,server=server_name])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>filterid</td>
<td>Optional. The unique identifier for the filter.</td>
</tr>
</tbody>
</table>
6.2.3.3 Examples

The following example adds an event route with the filter id of mds1 and the destination id of jfr:

```java
addDMSEventRoute(filterid='mds1', destinationid='jfr', enable='false')
```

Event-route for filter "mds1", destination "jfr" added.

The following example attempts to add an event route that already exists:

```java
addDMSEventRoute(filterid='mds1', destinationid='jfr', enable='false')
```

Event-route for filter "mds1", destination "jfr" already exists. Unable to add this.

6.2.4 enableDMSEventTrace

Command Category: DMS Event Tracing
Use with WLST: Online

6.2.4.1 Description

Enables an event trace and creates a filter with a specified condition and destination and an enabled event-route. This is a simple way to start filtering, without having to explicitly create a filter, destination and event-route, but with less configuration options. The specified destination must exist.

You must be connected to the Administration Server to enable a DMS event trace. If you are not, an error is returned.

If you require a more complex configuration, use the `addDMSEventDestination`, `addDMSEventFilter`, and `addDMSEventRoute`.

6.2.4.2 Syntax

```java
enableDMSEventTrace(destinationid=destinationid [, etypes=etype] [, condition=condition] [, server=server_name])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>destinationid</td>
<td>The unique identifier for the specific destination. Any existing destination is valid.</td>
</tr>
<tr>
<td>etypes</td>
<td>Optional. A string containing a comma-separated list of event/action pairs. See <code>addDMSEventFilter</code> for a list of available etypes.</td>
</tr>
<tr>
<td>condition</td>
<td>Optional. A condition on which to filter. See <code>addDMSEventFilter</code> for the syntax for a condition. If no condition is specified, all DMS events will be passed</td>
</tr>
<tr>
<td>server</td>
<td>Optional. The server on which to perform this operation. The default is the server to which you are connected.</td>
</tr>
</tbody>
</table>
6.2.4.3 Example
The following example enables an event trace with a specified condition:

```
enableDMSEventTrace(condition='CONTEXT username EQUALS Joe AND CONTEXT ip EQUALS 192.168.1.5')
```

Filter 'EventTrace9', using Destination "LoggerDestination" added, and event-route enabled.

6.2.5 listDMSEventConfiguration
Command Category: DMS Event Tracing
Use with WLST: Online

6.2.5.1 Description
Displays an overview of the Event Tracing configuration.

6.2.5.2 Syntax
```
listDMSEventConfiguration([server=server_name])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>server</td>
<td>Optional. The server on which to perform this operation. The default is the server to which you are connected.</td>
</tr>
</tbody>
</table>

6.2.5.3 Example
The following example lists the configuration for the Managed Server to which you are connected:

```
listDMSEventConfiguration()
```

Event routes:
- FILTER DESTINATION
- MyFilter des1
- MyFilter des2
- null des3

Filters with no event route:
- Fred

Destinations with no event route:
- des4

6.2.6 listDMSEventDestination
Command Category: DMS Event Tracing
Use with WLST: Online

6.2.6.1 Description
For a specific destination, display the full configuration. If no destination ID is specified, list the destination ID and name for all the destinations in the Event Tracing configuration.
6.2.6.2 Syntax

listDMSEventDestination([id=id] [, server=server_name])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Optional. The unique identifier for the specific destination.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. The server on which to perform this operation. The default is the server to which you are connected.</td>
</tr>
</tbody>
</table>

6.2.6.3 Examples

The following example displays information about the destinations for the Managed Server to which you are connected:

listDMSEventDestination()

ID : destination1
NAME: File-system
ID : jrf
NAME: Flight-Recorder

The following example displays information about the destinations for the Managed Server, MS1:

listDMSEventDestination(server='MS1')

ID       NAME
Network1  Send file over network
desman1   File-system

The following example displays information about the destination destination1:

listDMSEventDestination(id='destination1')

ID: destination1
NAME: File-system
CLASS: oracle.dms.trace2.runtime.LoggerDestination
PROPERTIES:
NAME       VALUE
LoggerName trace2-logger

6.2.7 listDMSEventFilter

Command Category: DMS Event Tracing
Use with WLST: Online

6.2.7.1 Description

For a specific filter, displays the full configuration. If you do not specify a filter ID, the command displays the filter ID and name for all the filters in the Event Tracing configuration.

6.2.7.2 Syntax

listDMSEventFilter([id=id] [, server=server_name])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Optional. The unique identifier for specified filter.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. The server on which to perform this operation. The default is the server to which you are connected.</td>
</tr>
</tbody>
</table>
6.2.7.3 Example

The following example displays the list of all the filters in the Event Tracing configuration:

```
listDMSEventFilter()
```

<table>
<thead>
<tr>
<th>ID</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>mds1</td>
<td>MyFilter</td>
</tr>
<tr>
<td>mds2</td>
<td>MDS2Filter</td>
</tr>
</tbody>
</table>

The following example displays the configuration of the filter mds1:

```
listDMSEventFilter(id='mds1')
```

<table>
<thead>
<tr>
<th>ID</th>
<th>NAME</th>
<th>PROPERTIES</th>
</tr>
</thead>
<tbody>
<tr>
<td>mds1</td>
<td>MyFilter</td>
<td>CONDITION: NOUNTYPE equals MDS_Connections AND CONTEXT user equals bruce IGNORECASE=false</td>
</tr>
</tbody>
</table>

6.2.8 listDMSEventRoutes

Command Category: DMS Event Tracing

Use with WLST: Online

6.2.8.1 Description

List the events routes and their status (enabled or disabled) that are associated with the specified filter or destination. If you do not specify a filterid or destinationid, this command lists all the event routes in the Event Tracing configuration.

6.2.8.2 Syntax

```
listDMSEventRoutes([filterid=filter_id] [, destinationid=destination_id] [, server=server_name])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>filterid</td>
<td>Optional. The unique identifier for the filter.</td>
</tr>
<tr>
<td>destinationid</td>
<td>Optional. The unique identifier for the specific destination. The destination must exist.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. The server on which to perform this operation. The default is the server to which you are connected.</td>
</tr>
</tbody>
</table>

6.2.8.3 Examples

The following example lists all event routes:

```
listDMSEventRoutes()
```

```
FILTER : mdsbruce
DESTINATION: jfr
ENABLED : false
FILTER : null
DESTINATION: destination1
ENABLED : true
```

The following example lists the event routes with the filter id of filter1:

```
listDMSEventRoutes(filterid='filter1')
```
The following example lists the event routes with the destination id of destination1:

```java
listDMSEventRoutes(destinationid='destination1')
FILTER : filter1
DESTINATION: destination1
ENABLED : true
```

### 6.2.9 removeDMSEventDestination

**Command Category:** DMS Event Tracing  
**Use with WLST:** Online

#### 6.2.9.1 Description

Removes an existing destination from the Event Tracing configuration. You can remove a destination only if no event route depends on the destination. If an event route that depends on the destination exists, a warning is returned.

You must be connected to the Administration Server to remove a destination. If you are not, an error is returned.

#### 6.2.9.2 Syntax

```
removeDMSEventDestination(id=id [, server=server_name])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The unique identifier for the destination to be removed.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. The server on which to perform this operation. The default is the server to which you are connected.</td>
</tr>
</tbody>
</table>

#### 6.2.9.3 Examples

The following example removes the destination jfr:

```
removeDMSEventDestination(id='jfr')
```

Destination 'jfr' removed.

The following example attempts to remove the destination styx.inpass.db1. However, because an event route exists for the destination, the command returns an error.

```
removeDMSEventDestination(id='styx.inpass.db1')
```

Destination '"styx.inpass.db1"' cannot be removed. An event-route currently exists for that destination. Remove the event-route first using the command removeDMSEventRoute().

### 6.2.10 removeDMSEventFilter

**Command Category:** DMS Event Tracing  
**Use with WLST:** Online
6.2.10.1 Description
Removes an existing filter from the Event Tracing configuration. You can remove a filter only if no event route depends on the filter. If an event route that depends on the filter exists, a warning is returned.

You must be connected to the Administration Server to remove an event filter. If you are not, an error is returned.

6.2.10.2 Syntax
removeDMSEventFilter(id=id [, server=server_name])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The unique identifier for the filter to be removed.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. The server on which to perform this operation. The default is the server to which you are connected.</td>
</tr>
</tbody>
</table>

6.2.10.3 Example
The following example removes the filter mds1:
removeDMSEventFilter(id='mds1')

Filter "mds1" removed.

The following example attempts to remove a filter for which and event-route currently exists:
removeDMSEventFilter(id='allaccounts')

Filter "allaccounts" cannot be removed. An event-route currently exists for that filter. Remove the event-route first using the command removeDMSEventRoute().

6.2.11 removeDMSEventRoute
Command Category: DMS Event Tracing
Use with WLST: Online

6.2.11.1 Description
Removes the specified event route. You must be connected to the Administration Server to add an event route. If you are not, an error is returned.

6.2.11.2 Syntax
removeDMSEventRoute([filterid=filter_id] [, destinationid=destination_id] [, server=server_name])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>filterid</td>
<td>Optional. The unique identifier for the filter.</td>
</tr>
<tr>
<td>destinationid</td>
<td>Optional. The unique identifier for the specific destination. The destination must exist.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. The server on which to perform this operation. The default is the server to which you are connected.</td>
</tr>
</tbody>
</table>
6.2.11.3 Example
The following example removes the event route with the filterid mds1 and the destination jfr:
```python
removeDMSEventRoute(filterid='mds1', destinationid='jfr')
```
Event-route for filter "mds1", destination "jfr" removed

The following example removes the event route with the destination destination1:
```python
removeDMSEventRoute(destinationid='destination1')
```
Event-route for filter "None", destination "destination1" removed

6.2.12 updateDMSEventDestination
Command Category: DMS Event Tracing
Use with WLST: Online

6.2.12.1 Description
Updates an existing destination, allowing a specified argument to be updated. You must be connected to the Administration Server to update a destination. If you are not, an error is returned.

6.2.12.2 Syntax
```python
updateDMSEventDestination(id=id [, name=dest_name], class=class_name
[,props= {'name': 'value'...}] [, server=server_name])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The unique identifier for the destination to be updated.</td>
</tr>
<tr>
<td>name</td>
<td>Optional. A name for the destination.</td>
</tr>
<tr>
<td>class</td>
<td>The full classname of the destination.</td>
</tr>
<tr>
<td>props</td>
<td>Optional. The name/value properties to use for the destination. You can add a new property, or update or remove an existing one. If you update properties, you must specify all properties. If you omit a property, it is removed. For example, if a destination contains the properties LoggerName and severity, and you omit severity, it will be removed from the destination.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. The server on which to perform this operation. The default is the server to which you are connected.</td>
</tr>
</tbody>
</table>

6.2.12.3 Examples
The following example updates the name of the destination jfr:
```python
updateDMSEventDestination(id='jfr', name='Alternative Flight-Recorder')
```
Destination 'jfr' updated.

The following example attempts to update a destination that does not exist. The command returns an error:
```python
updateDMSEventDestination(id='destination1',
props={'loggerName': 'MyNewTrace2-logger'})
```
Destination 'destination1' does not yet exist. Unable to update this.
6.2.13 updateDMSEventFilter

Command Category: DMS Event Tracing
Use with WLST: Online

6.2.13.1 Description
Updates an existing filter in the Event Tracing configuration.
You must be connected to the Administration Server to update an event filter. If you
are not, an error is returned.

6.2.13.2 Syntax
updateDMSEventFilter(id=id [, name=name] [,etypes=etypes],
props= {'prop-name': 'value'...}
[,server=server_name])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The unique identifier for the filter to be updated.</td>
</tr>
<tr>
<td>name</td>
<td>Optional. The name of the filter to be updated.</td>
</tr>
<tr>
<td>etypes</td>
<td>Optional. A string containing a comma-separated list of event/action pairs. See addDMSEventFilter for a list of valid values.</td>
</tr>
<tr>
<td>props</td>
<td>prop-name: The name of the filter property. &lt;condition&gt; is the only valid property, and only one condition may be specified. See addDMSEventFilter for information on the syntax of prop-name. value: The value of the property of the filter.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. The server on which to perform this operation. The default is the server to which you are connected.</td>
</tr>
</tbody>
</table>

6.2.13.3 Examples
The following example updates the filter properties for the filter with the id mds1:

updateDMSEventFilter(id='mds1',
props= {'condition': 'NOUNTYPE equals XYZ_Total_Connections AND CONTEXT user equals bruce'})

Filter "mds1" updated.

The following example attempts to update a filter that does not exist:

updateDMSEventFilter(id='Filter2')

Filter "Filter2" does not yet exist. Unable to update this.

6.2.14 updateDMSEventRoute

Command Category: DMS Event Tracing
Use with WLST: Online

6.2.14.1 Description
Enables or disables the specified event route. You must be connected to the
Administration Server to update an event route. If you are not, an error is returned.
6.2.14.2 Syntax

updateDMSEventRoute([filterid=filter_id], destinationid=destination_id
[, enable=true|false] [, server=server_name])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>filterid</td>
<td>Optional. The unique identifier for the filter.</td>
</tr>
<tr>
<td>destinationid</td>
<td>Optional. The unique identifier for the specific destination. The destination must exist.</td>
</tr>
<tr>
<td>enable</td>
<td>Optional. Enables the filter. Valid values are true and false.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. The server on which to perform this operation. The default is the server to which you are connected.</td>
</tr>
</tbody>
</table>

6.2.14.3 Example

The following example disables the event route with the filterid mds1 and the destinationid jfr:

updateDMSEventRoute(filterid='mds1', destinationid='jfr', enable='false')

Event-route for filter "mds1", destination "jfr" disabled.
This chapter provides detailed descriptions of custom WLST commands for logging, including command syntax, arguments and command examples. Use the logging commands to configure settings for log files and to view and search log files.

For additional details about configuring and searching log files, see "Managing Log Files and Diagnostic Data" in the Oracle Fusion Middleware Administrator’s Guide.

Note: To use these logging custom WLST commands, you must invoke the WLST script from the Oracle Common home. See "Using Custom WLST Commands" in the Oracle Fusion Middleware Administrator’s Guide.

Table 7–1 describes the different categories of logging commands.

<table>
<thead>
<tr>
<th>Command category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Log Configuration Commands</strong></td>
<td>Configure settings for log files, such as the level of information written to the file or the maximum file size.</td>
</tr>
<tr>
<td><strong>Search and Display Commands</strong></td>
<td>View Oracle Fusion Middleware log files and search log files for particular messages.</td>
</tr>
<tr>
<td><strong>Selective Tracing Commands</strong></td>
<td>Configure and use selective tracing, which specifies that messages are traced for specific server, loggers, or users.</td>
</tr>
</tbody>
</table>

### 7.1 Log Configuration Commands

Use the commands in Table 7–2 to configure settings for log files, such as the level of information written to the file or the maximum file size. In the Use with WLST column, online means the command can only be used when connected to a running server. Offline means the command can only be used when not connected to a running server. Online or offline means the command can be used in both situations.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>configureLogHandler</td>
<td>Configure an existing log handler, add a new handler, or remove existing handlers.</td>
<td>Online</td>
</tr>
<tr>
<td>getLogLevel</td>
<td>Get the level for a given logger.</td>
<td>Online</td>
</tr>
<tr>
<td>listLoggers</td>
<td>Get the list of loggers and the level of each logger.</td>
<td>Online</td>
</tr>
</tbody>
</table>
7.1.1 configureLogHandler

Command Category: Log Configuration

Use with WLST: Online

7.1.1.1 Description

Configures an existing Java logging handler, adds a new handler, or removes an existing handler. It returns a java.util.List with one entry for each handler. Each entry is a javax.management.openmbean.CompositeData object describing the handler.

With this command, you can change the location of the log files, the frequency of the rotation of log files, and other log file properties.

7.1.1.2 Syntax

configureLogHandler(options)
<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Comma-separated list of options, specified as name-value pairs. Valid options include:</td>
</tr>
<tr>
<td></td>
<td>■ <strong>target</strong>—The name of a WebLogic Server instance, or a string describing a system component. For system components, refer to the component’s documentation for details. The default value is the server to which WLST is connected.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>name</strong>—The name of a log handler. This option is required.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>maxFileSize</strong>—The value of the maximum file size for an ODL handler. The value is a string representing a numeric value, optionally followed by a suffix indicating a size unit (k for kilobytes, m for megabytes, g for gigabytes). If you do not specify a suffix, the value is returned in bytes. Note that this option does not apply to the QuickTrace handler.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>maxLogSize</strong>—The value of the maximum size of the log files for an ODL handler. The value is a string representing a numeric value, optionally followed by a suffix indicating a size unit (k for kilobytes, m for megabytes, g for gigabytes). Note that this option does not apply to the QuickTrace handler.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>rotationFrequency</strong>—The value of the rotation frequency for an ODL handler. The value is a string representing a numeric value, optionally followed by a suffix indicating a time unit (m for minutes, h for hours, d for days). The default unit is minutes. The following special values are also accepted and are converted to a numeric value in minutes: HOUR, HOURLY, DAY, DAILY, WEEK, WEEKLY, MONTH, MONTHLY. Note that this option does not apply to the QuickTrace handler.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>baseRotationTime</strong>—The base rotation time, to be used with the rotationFrequency option. The value must be a string representing a date/time value. It can be a full date/time in ISO 8601 date/time format, or a short form including only hours and minutes. The default baseRotationTime is 00:00. Note that this option does not apply to the QuickTrace handler.</td>
</tr>
</tbody>
</table>
The following table lists the properties for the quicktrace-handler. This handler allows you to trace messages from specific loggers and store the messages in memory. For more information, see "Configuring QuickTrace" in the Oracle Fusion Middleware Administrator’s Guide.

<table>
<thead>
<tr>
<th>QuickTrace Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bufferSize</td>
<td>The approximate size of the circular QuickTrace buffer, in which log records are stored in memory. Note that actual memory consumption may be less than, but not more than this value.</td>
</tr>
</tbody>
</table>
### 7.1.1.3 Examples

The following example specifies the maximum file size for the odl-handler:

```java
configureLogHandler(name="odl-handler", maxFileSize="5M")
```

The following example specifies the rotation frequency for the odl-handler:

```java
configureLogHandler(name="odl-handler", rotationFrequency="1M")
```
configureLogHandler(name="odl-handler", rotationFrequency="daily")

The following example specifies the rotation frequency and the retention period for the odl-handler. It also removes the properties maxFileSize and maxLogSize:

configureLogHandler(name="odl-handler", rotationFrequency="daily",
                   retentionPeriod="week", removeProperty=["maxFileSize", "maxLogSize"])

The following example configures the quicktrace-handler, adding the logger oracle.adf.faces, and enabling user buffers for user1 and user2:

configureLogHandler(name="quicktrace-handler", addToLogger="oracle.adf.faces",
                   propertyName="enableUserBuffer", propertyName="reserveBufferUserID",
                   propertyValue="true", propertyValue="user1, user2")

The oracle.adf logger is associated with the handlers odl-handler, wls-domain, and console-handler. When you set the level of the logger, these handlers will use the same level (TRACE:1) for the logger oracle.adf. As a result, much information will be written to the log files, consuming resources. To avoid consuming resources, set the level of the handlers to a lower level, such as WARNING or INFORMATION. For example:

configureLogHandler(name="odl-handler", level="WARNING:1")
configureLogHandler(name="wls-domain", level="WARNING:1")
configureLogHandler(name="console-handler", level="WARNING:1")

7.1.2 getLogLevel

Command Category: Log Configuration
Use with WLST: Online

7.1.2.1 Description
Returns the level of a given Java logger.
The returned value is a string with the logger’s level, or None if the logger does not exist. An empty string indicates that the logger level is null.

7.1.2.2 Syntax
getLogLevel(options)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>Comma-separated list of options, specified as name-value pairs. Valid options include:</td>
</tr>
<tr>
<td></td>
<td>■ target—The name of a WebLogic Server instance, or a string describing a system component. For system components, refer to the component’s documentation for details. The default value is the server to which WLST is connected.</td>
</tr>
<tr>
<td></td>
<td>■ logger—A logger name. An empty string denotes the root logger. This option is required and has no default.</td>
</tr>
<tr>
<td></td>
<td>■ runtime—A Jython boolean value (0 or 1) that determines if the operation is to list runtime loggers or config loggers. The default value is 1 (runtime).</td>
</tr>
</tbody>
</table>

7.1.2.3 Examples
The following example returns the level for the logger oracle:
The following example returns the level for the logger oracle, specifying only config loggers, not runtime loggers:

```python
getLogLevel(logger='oracle')
```

The following example returns the level for the logger oracle on the Oracle WebLogic Server server2:

```python
getLogLevel(logger='oracle', target='server2')
```

### 7.1.3 listLoggers

**Command Category:** Log Configuration

**Use with WLST:** Online

#### 7.1.3.1 Description

Lists Java loggers and their levels. The command returns a PyDictionary object where the keys are logger names and the associated values are the logger levels. An empty level is used to indicate that the logger does not have the level set.

#### 7.1.3.2 Syntax

```python
listLoggers([options])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>An optional comma-separated list of options, specified as name-value pairs. Valid options include:</td>
</tr>
<tr>
<td></td>
<td>■ <strong>target</strong>—The name of a WebLogic Server instance, or a string describing a system component. For system components, refer to the component's documentation for details. The default value is the server to which WLST is connected.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>pattern</strong>—A regular expression pattern that is used to filter logger names. The default value returns all logger names.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>runtime</strong>—A Jython boolean value (0 or 1) that determines if the operation is to list runtime loggers or config loggers. The default value is 1 (runtime).</td>
</tr>
</tbody>
</table>

#### 7.1.3.3 Examples

The following example lists all of the loggers:

```python
listLoggers()
```

The following example lists all of the loggers that start with the name oracle.*.

```python
listLoggers(pattern="oracle.*")
```

The following example list all config loggers:

```python
listLoggers(runtime=0)
```

The following example list all loggers for the WebLogic Server server1:

```python
listLoggers(target="server1")
```
7.1.4 listLogHandlers

Command Category: Log Configuration
Use with WLST: Online

7.1.4.1 Description
Lists Java log handlers configuration. This command returns a java.util.List with one entry for each handler. Each entry is a javax.management.openmbean.CompositeData object describing the handler.

7.1.4.2 Syntax
listLogHandlers([options])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>An optional comma-separated list of options, specified as name-value pairs. Valid options include:</td>
</tr>
<tr>
<td></td>
<td>■ target—The name of a WebLogic Server instance, or a string describing a system component. For system components, refer to the component's documentation for details. The default value is the server to which WLST is connected.</td>
</tr>
<tr>
<td></td>
<td>■ name—The name of a log handler. If the name is not provided, then all handlers are listed.</td>
</tr>
</tbody>
</table>

7.1.4.3 Examples
The following example lists all log handlers:
listLogHandlers()

The following example lists all log handlers named odl-handler:
listLogHandlers(name="odl-handler")

The following example lists all log handlers for the WebLogic Server server1:
listLogHandlers(target="server1")

7.1.5 setLogLevel

Command Category: Log Configuration
Use with WLST: Online

7.1.5.1 Description
Sets the level of information written by a given Java logger to a log file.

7.1.5.2 Syntax
setLogLevel(options)
### 7.1.5.3 Examples

The following example sets the log level to NOTIFICATION:1 for the logger oracle.my.logger:

```java
setLogLevel(logger="oracle.my.logger", level="NOTIFICATION:1")
```

The following example sets the log level to TRACE:1 for the logger oracle.my.logger and specifies that the level should be saved to the configuration file:

```java
setLogLevel(logger="oracle.my.logger", level="TRACE:1", persist=0)
```

The following example sets the log level to WARNING for the config logger oracle.my.logger on the WebLogic Server server1:

```java
setLogLevel(target="server1", logger="oracle.my.logger", level="WARNING", runtime=0)
```
7.2 Search and Display Commands

Use the commands in Table 7–3 to view Oracle Fusion Middleware log files and to search log files for particular messages.

Table 7–3  Search and Display Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>displayLogs</td>
<td>List the logs for one or more components.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>listLogs</td>
<td>Search and display the contents of log files.</td>
<td>Online or Offline</td>
</tr>
</tbody>
</table>

7.2.1 displayLogs

Command Category: Search and Display

Use with WLST: Online or Offline

7.2.1.1 Description

Search and display the contents of diagnostic log files. The command returns a value only when the returnData option is set to true. By default it will not return any data. The return value depends on the option used.

7.2.1.2 Syntax

displayLogs([searchString,][options])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>searchString</td>
<td>An optional search string. Only messages that contain the given string (case-insensitive) will be returned. Note that the displayLogs command can read logs in multiple formats and it converts the messages to ODL format. The search will be performed in the native format, if possible. Otherwise, it may be performed in the message contents, and it may exclude mark-up. Therefore you should avoid using mark-up characters in the search string.</td>
</tr>
<tr>
<td>options</td>
<td>An optional comma-separated list of options, specified as name-value pairs. Valid options include:</td>
</tr>
</tbody>
</table>

- **target**—The name of a WebLogic Server instance, or a system component.
  
  For a system component, the syntax for the target is:
  
  `opmn:instance-name/component-name`
  
  In connected mode, the default target is the WebLogic domain. In disconnected mode, there is no default; the target option is required.

- **oracleInstance**—Defines the path to the ORACLE_INSTANCE or WebLogic domain home. The command is executed in disconnected mode when you use this parameter.

- **log**—A log file path. The command will read messages from the given log file. If the log file path is not given, the command will read all logs associated with the given target.
options (continued)

- **last**—An integer value. Restricts the search to messages logged within the last minutes. The value can have a suffix s (second), m (minute), h (hour), or d (day) to specify a different time unit. (For example, last=’2h’ will be interpreted as the last 2 hours).
- **tail**—An integer value. Restricts the search to the last n messages from each log file and limits the number of messages displayed to n.
- **pattern**—A regular expression pattern. Only messages that contain the given pattern are returned. Using the pattern option is similar to using the searchString argument, except that you can use a regular expression.
  
  The regular expression pattern search is case sensitive (unless you explicitly turn on case-insensitive flags in the pattern). The pattern must follow java.util.regex syntax.
- **ecid**—A string or string sequence containing one or more Execution Context ID (ECID) values to be used as a filter for log messages.
- **component**—A string or string sequence containing one or more component ID values to be used as a filter for log messages.
- **module**—A string or string sequence containing one or more module ID values to be used as a filter for log messages.
- **type**—A string or string sequence containing one or more message type values to be used as a filter for log messages.
- **app**—A string or string sequence containing one or more application values to be used as a filter for log messages.
- **query**—A string that specifies an expression used to filter the contents of log messages.
  
  A simple expression has the form:

  field-name operator value

  where field-name is a log record field name and operator is an appropriate operator for the field type (for example, you can specify equals, startsWith, contains or matches for string fields).

  A field name is either one of the standard ODL attribute names (such as COMPONENT_ID, MSG_TYPE, MSG_TEXT, and SUPPL_DETAIL), or the name of a supplemental attribute (application specific), prefixed by SUPPL_ATTR. (For example, SUPPL_ATTR.myAttribute).

  A few common supplemental attributes can be used without the prefix. For example, you can use APP to filter by application name.

  You can combine multiple simple expressions using the boolean operators and, or and not to create complex expressions, and you can use parenthesis for grouping expressions.

  See the Oracle Fusion Middleware Administrator’s Guide for a detailed description of the query syntax.
- **groupBy**—A string list. When the groupBy option is used, the output is a count of log messages, grouped by the attributes defined in the string list.
- **orderBy**—A string list that defines the sort order for the result. The values are log message attribute names. The name may be extended with an optional suffix :asc or :desc to specify ascending or descending sorting. The default sort order is ascending.
  
  By default, the result is sorted by time.
- **returnData**—A Jython boolean value (0 or 1). If the value is true the command will return data (for example, to be used in a script). The default value is false, which means that the command only displays the data but does not return any data.
7.2.3 Examples

The following example displays the last 100 messages from all log files in the domain:

displayLogs(tail=100)

The following example displays all messages logged in the last 15 minutes:

displayLogs(last='15m')

The following example displays log messages that contain a given string:

displayLogs('Exception')

The following example displays log messages that contain a given ECID:

displayLogs(ecid='0000H19TwKUCs1T6uBi8UH181bWX000002')

The following example displays log messages of type ERROR or INCIDENT_ERROR:

displayLogs(type=['ERROR','INCIDENT_ERROR'])

The following example displays log messages for a given Java EE application:

displayLogs(app="myApplication")

The following example displays messages for a system component, ohs1:

displayLogs(target="opmn:instance1/ohs1")

The following example displays a message summary by component and type:

displayLogs(groupBy=['COMPONENT_ID', 'MSG_TYPE'])

The following example displays messages for a particular time interval:

displayLogs(query="TIME from 11:15 and TIME to 11:20")

The following example shows an advanced query:

displayLogs(query="TIME from 11:15 and TIME to 11:20 and ( MSG_TEXT contains exception or SUPPL_DETAIL contains exception )")

A similar query could be written as:

displayLogs("exception", query="TIME from 11:15 and TIME to 11:20")

7.2.2 listLogs

Command Category: Search and Display

Use with WLST: Online or Offline
7.2.2.1 Description
Lists log files for Oracle Fusion Middleware components. This command returns a
PyArray with one element for each log. The elements of the array are
javax.management.openmbean.CompositeData objects describing each log.

7.2.2.2 Syntax
listLogs([options]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>An optional comma-separated list of options, specified as name-value pairs. Valid options include:</td>
</tr>
<tr>
<td>target</td>
<td>The name of a WebLogic Server instance, or an Oracle Fusion Middleware system component. For a system component, the syntax for the target is: <code>opmn:instance-name/component-name</code> In connected mode, the default target is the WebLogic domain. In disconnected mode, there is no default; the target option is required.</td>
</tr>
<tr>
<td>oracleInstance</td>
<td>Defines the path to the ORACLE_INSTANCE or WebLogic domain home. The command is executed in disconnected mode when you use this parameter.</td>
</tr>
<tr>
<td>unit</td>
<td>defines the unit to use for reporting file size. Valid values are B (bytes), K (kilobytes), M (megabytes), G (gigabytes), or H (display size in a human-readable form, similar to the UNIX <code>ls -h</code> option). The default value is H.</td>
</tr>
<tr>
<td>fullTime</td>
<td>A Jython Boolean value. If true, reports the full time for the log file last modified time. Otherwise, it displays a short version of the time. The default value is false.</td>
</tr>
</tbody>
</table>

7.2.2.3 Examples
The following example lists all of the log files for the WebLogic domain:

`listLogs()`

The following example lists the log files for the WebLogic Server server1:

`listLogs(target="server1")`

The following example lists the log files for the Oracle HTTP Server ohs1:

`listLogs(target="opmn:instance1/ohs1")`

The following example, used in disconnected mode, lists the log files for the WebLogic Server server1:

`listLogs(oracleInstance="/middleware/user_projects/domains/base_domain", target="server1")`

7.3 Selective Tracing Commands
Use the commands in Table 7–4 to configure and use selective tracing. Selective tracing provides fine-grained logging for specified users or other properties of a request. In the Use with WLST column, online means the command can only be used when connected to a running server.
7.3.1 configureTracingLoggers

Command Category: Tracing
Use with WLST: Online

7.3.1.1 Description
Configures one or more loggers for selective tracing. This command also enables or disables a logger for selective tracing.

7.3.1.2 Syntax
configureTracingLoggers([options])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>A comma-separated list of options, specified as name-value pairs. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>■ target—Optional. The name of a WebLogic Server instance, or an array of strings containing one or more target names. By default, loggers on all running server instances in the domain that are JRF-enabled will be configured for tracing.</td>
</tr>
<tr>
<td></td>
<td>■ pattern—A regular expression pattern that is used to filter logger names. The default value matches all tracing logger names.</td>
</tr>
<tr>
<td></td>
<td>■ action—Enables or disables all loggers for tracing. Valid values are enable and disable. This option is required; there is no default value.</td>
</tr>
</tbody>
</table>

7.3.1.3 Examples
The following example configures selective tracing for all loggers beginning with oracle.security:

configureTracingLoggers(pattern='oracle.security.*', action='enable')
Configured 62 loggers

The following example disables selective tracing for all loggers:

configureTracingLoggers(action='disable')
Configured 1244 loggers

7.3.2 listActiveTraces
Command Category: Tracing
Use with WLST: Online
7.3.2.1 Description
Lists the active traces.

7.3.2.2 Syntax
listActiveTraces([options])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>A comma-separated list of options, specified as name-value pairs. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>■ target—Optional. The name of a WebLogic Server instance, or an array of strings containing one or more target names. By default, loggers on all running server instances in the domain that are JRF-enabled are listed.</td>
</tr>
</tbody>
</table>

7.3.2.3 Example
The following example lists the active traces:

listActiveTraces()

<table>
<thead>
<tr>
<th>Trace ID</th>
<th>Attr. Name</th>
<th>Attr. Value</th>
<th>Level</th>
<th>Exp. Time</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a9580e65-13c4-420b-977e-5ba7d88ca7f</td>
<td>USER_ID</td>
<td>user1</td>
<td>FINE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a04b47f7-2830-4d80-92ee-ba1e0cdac6f6</td>
<td>USER_ID</td>
<td>user2</td>
<td>FINE</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

7.3.3 listTracingLoggers

Command Category: Tracing
Use with WLST: Online or Offline

7.3.3.1 Description
Lists the loggers that support selective tracing. This command displays a table of logger names and their tracing status. The status enabled means that the logger is enabled for tracing on all servers. The status disabled means that the logger is disabled for tracing on all servers. The status mixed means that the logger is enabled for tracing on some servers, but disabled on others.

7.3.3.2 Syntax
listTracingLoggers([options])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>A comma-separated list of options, specified as name-value pairs. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>■ target—Optional. The name of a WebLogic Server instance, or an array of strings containing one or more target names. By default, loggers on all running server instances in the domain that are JRF-enabled are listed.</td>
</tr>
<tr>
<td></td>
<td>■ pattern—A regular expression pattern that is used to filter logger names. The default value matches all tracing logger names.</td>
</tr>
</tbody>
</table>

7.3.3.3 Example
The following example lists all tracing loggers beginning with oracle.security:

listTracingLoggers(pattern="oracle.security.*")

<table>
<thead>
<tr>
<th>Logger</th>
<th>Status</th>
</tr>
</thead>
</table>
### 7.3.4 startTracing

Command Category: Tracing

Use with WLST: Online

#### 7.3.4.1 Description

Starts a new selective tracing session for a specified user or DMS context attribute at a specified level of tracing.

#### 7.3.4.2 Syntax

```
startTracing([options])
```

#### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>options</td>
<td>A comma-separated list of options, specified as name-value pairs. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>■ <code>target</code>—Optional. The name of a WebLogic Server instance, or an array of strings containing one or more target names. By default, loggers on all running server instances in the domain that are JRF-enabled are included in the trace.</td>
</tr>
<tr>
<td></td>
<td>■ <code>traceId</code>—Optional. An identifier for the tracing session. If a traceId is not provided, the command generates a unique traceId.</td>
</tr>
<tr>
<td></td>
<td>■ <code>attrName</code>—Optional, unless the user argument is not specified. Valid values are USER_ID, APP, CLIENT_HOST, CLIENT_ADDR, composite_name, WEBSERVICE.name, WEBSERVICE_PORT.name.</td>
</tr>
<tr>
<td></td>
<td>■ <code>attrValue</code>—Required if attrName is specified. The value of the attribute.</td>
</tr>
<tr>
<td></td>
<td>■ <code>user</code>—The user name. Messages associated with the user are returned. This is equivalent to passing the USER_ID with the attrName and AttrValue options.</td>
</tr>
<tr>
<td></td>
<td>■ <code>level</code>—Required. The tracing level. The level must be a valid Java or ODL level. See the table “Mapping of Log Levels Among ODL, Oracle WebLogic Server, and Java” in the Oracle Fusion Middleware Administrator’s Guide.</td>
</tr>
<tr>
<td></td>
<td>■ <code>desc</code>—Optional. A description of the tracing session.</td>
</tr>
</tbody>
</table>

#### 7.3.4.3 Example

The following example starts a trace for messages associated with user1 and sets the level of information to FINE:

```
startTracing(user="user1",level="FINE")
```

Started tracing with ID: 885649f7-8efd-4a7a-9898-acbbfc0bba3

---

### 7.3.5 stopTracing

Command Category: Tracing

Use with WLST: Online
### 7.3.5.1 Description
Stops one or more selective tracing sessions.

### 7.3.5.2 Syntax
```
stopTracing([options])
```

### Argument | Definition
---|---
**options** | A comma-separated list of options, specified as name-value pairs. Valid options are:
- **target**—Optional. The name of a WebLogic Server instance, or an array of strings containing one or more target names. By default, loggers on all running server instances in the domain that are JRF-enabled are included in the operation.
- **stopAll**—A Jython boolean value (0 or 1) that determines if all of the active traces are stopped. Required if the traceId, user, or attrName and attrValue arguments are not specified. The default value is 0 (false).
- **traceId**—An identifier for the tracing session to be stopped. Required if the stopAll, user, or attrName and attrValue arguments are not specified.
- **attrName**—Valid values are USER_ID, APP, CLIENT_HOST, CLIENT_ADDR, composite_name, WEBSERVICE.name, WEBSERVICE_PORT.name. Required if the traceId, user, stopAll arguments are not specified.
- **attrValue**—Required if attrName is specified. The value of the attribute.
- **user**—The user name. All tracing sessions associated with the user are stopped. Required if the stopAll, traceId, or attrName and attrValue arguments are not specified.

### 7.3.5.3 Examples
The following example stops a tracing session with a specified traceId:
```
stopTracing(traceId="a04b47f7-2830-4d80-92ee-ba160cdacf6b")
```
Stopped 1 traces

The following example stops all tracing sessions:
```
stopTracing(stopAll=1)
```
Stopped 1 traces
This chapter provides detailed descriptions of WLST commands for Oracle Metadata Services (MDS), including command syntax, arguments and command examples.

For additional details about creating and managing an MDS repository, see the chapter "Managing the Oracle Metadata Repository" in the Oracle Fusion Middleware Administrator’s Guide. For information about the roles needed to perform each operation, see "Understanding MDS Operations" in the Oracle Fusion Middleware Administrator’s Guide.

Note: To use these MDS custom WLST commands, you must invoke the WLST script from the Oracle Common home. See "Using Custom WLST Commands" in the Oracle Fusion Middleware Administrator’s Guide.

Use the Oracle Metadata Services (MDS) commands in the categories listed in Table 8–1 to manage MDS.

**Table 8–1  MDS Command Categories**

<table>
<thead>
<tr>
<th>Command category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Repository Management Commands</td>
<td>Manage the MDS repository.</td>
</tr>
<tr>
<td>Application Metadata Management Commands</td>
<td>Manage the application metadata in the MDS repository.</td>
</tr>
<tr>
<td>Sandbox Metadata Management Commands</td>
<td>Manage the metadata in a sandbox in the MDS repository.</td>
</tr>
<tr>
<td>Application Label Management Commands</td>
<td>Manage the labels for the application.</td>
</tr>
<tr>
<td>Application Management Deployment Commands</td>
<td>Manage the application deployment.</td>
</tr>
<tr>
<td>Multitenancy Management Commands</td>
<td>Manage tenants.</td>
</tr>
</tbody>
</table>

### 8.1 Repository Management Commands

Use the MDS commands listed in Table 8–2 to manage the MDS repository. In the Use with WLST column, online means the command can only be used when connected to a running Administration Server. Offline means the command can only be used when
not connected to a running server. Online or offline means the command can be used in both situations.

### Table 8–2  Repository Management Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>createMetadataPartition</td>
<td>Create a metadata repository partition.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteMetadataPartition</td>
<td>Delete a metadata repository partition.</td>
<td>Online</td>
</tr>
<tr>
<td>deregisterMetadataDBRepository</td>
<td>Deregister a database-based MDS repository.</td>
<td>Online</td>
</tr>
<tr>
<td>registerMetadataDBRepository</td>
<td>Register a database-based MDS repository.</td>
<td>Online</td>
</tr>
</tbody>
</table>

#### 8.1.1 createMetadataPartition

**Command Category:** Repository Management

**Use with WLST:** Online

**Description**
A metadata repository is used as a common repository for managing metadata of different applications. Many applications use the MDS repository to manage their metadata. Each deployed application uses a logical partition in metadata repository. This logical partition also helps in maintaining the metadata lifecycle. Before deploying an application, you create a partition for it in MDS repository. This command creates a partition with the given name in the specified repository.

**Syntax**
createMetadataPartition(repository, partition)

**Argument**
- **repository**
  - Definition: The name of the repository where the partition will be created.
- **partition**
  - Definition: The name of the partition to create in the repository.

**Example**
The following example creates the metadata partition `partition1` in the repository `mds-myrepos`:

```
  wls:/weblogic/serverConfig> createMetadataPartition(repository='mds-myrepos', partition='partition1')
  Executing operation: createMetadataPartition
  Metadata partition created: partition1
```

#### 8.1.2 deleteMetadataPartition

**Command Category:** Repository Management

**Use with WLST:** Online

**Description**
Deletes a metadata partition in the specified repository. When you delete a repository partition, all of the metadata in that partition is lost.

**Example**
The following example deletes the metadata partition `partition1` in the repository `mds-myrepos`:

```
  wls:/weblogic/serverConfig> deleteMetadataPartition(repository='mds-myrepos', partition='partition1')
  Executing operation: deleteMetadataPartition
  Metadata partition deleted: partition1
```
8.1.2.2 Syntax

\texttt{deleteMetadataPartition(repository, partition)}

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>repository</td>
<td>The name of the repository that contains the partition.</td>
</tr>
<tr>
<td>partition</td>
<td>The name of the partition to delete in the repository.</td>
</tr>
</tbody>
</table>

8.1.2.3 Example

The following example deletes the metadata partition \texttt{partition1} from the repository \texttt{mds-myrepos}:

\begin{verbatim}
wls:/weblogic/serverConfig> deleteMetadataPartition(repository='mds-myrepos', partition='partition1')
Executing operation: deleteMetadataPartition
Metadata partition deleted: partition1
\end{verbatim}

8.1.3 deregisterMetadataDBRepository

Command Category: Repository Management
Use with WLST: Online

8.1.3.1 Description

Removes the database metadata repository registration as a System JDBC data source in the domain. After this command completes successfully, applications can no longer use this repository.

8.1.3.2 Syntax

deregisterMetadataDBRepository(name)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the repository to deregister.</td>
</tr>
</tbody>
</table>

8.1.3.3 Example

The following example deregisters the metadata repository \texttt{mds-myrepos}:

\begin{verbatim}
wls:/weblogic/serverConfig> deregisterMetadataDBRepository('mds-myrepos')
Executing operation: deregisterMetadataDBRepository.
Metadata DB repository "mds-myrepos" was deregistered successfully.
\end{verbatim}

8.1.4 registerMetadataDBRepository

Command Category: Repository Management
Use with WLST: Online

8.1.4.1 Description

A database metadata repository should be registered with WebLogic Server instances before the application can use it. This command registers a System JDBC data source with the domain for use as database-based metadata repository.
8.1.4.2 Syntax

registerMetadataDBRepository(name, dbVendor, host, port, dbName, user, password [, targetServers])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the repository to register. If the name you supply does not begin with mds-, the commands adds the prefix mds-.</td>
</tr>
<tr>
<td>dbVendor</td>
<td>The database vendor. The acceptable values are ORACLE, MSSQL, IBMDB2, and MYSQL.</td>
</tr>
<tr>
<td>host</td>
<td>The host name or the IP address of the database.</td>
</tr>
<tr>
<td>port</td>
<td>The port number used by the database.</td>
</tr>
<tr>
<td>dbName</td>
<td>The service name of the database. For example, orcl.hostname.com</td>
</tr>
<tr>
<td>user</td>
<td>The database user name.</td>
</tr>
<tr>
<td>password</td>
<td>The password for the database user.</td>
</tr>
<tr>
<td>targetServers</td>
<td>Optional. The WebLogic Server instances to which this repository will be registered. If this argument is not specified, then the repository will be registered only to the Administration Server. To specify multiple servers, separate the names with a comma. Register the repository with all Managed Servers to which the application will be deployed.</td>
</tr>
</tbody>
</table>

8.1.4.3 Example

The following example registers the metadata repository myrepos to two servers, and specifies the database parameters:

```
wls:/weblogic/serverConfig> registerMetadataDBRepository('myrepos','ORACLE',
'test.oracle.com','1521','mds', 'user1','x','server1, server2')
Executing operation: registerMetadataDBRepository.
Metadata DB repository 'mds-myrepos' was registered successfully.
'mds-myrepos'
```

8.2 Application Metadata Management Commands

Use the commands in Table 8–3 to manage application metadata.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>deleteMetadata</td>
<td>Deletes the metadata in the application repository.</td>
<td>Online</td>
</tr>
<tr>
<td>exportMetadata</td>
<td>Exports metadata for an application.</td>
<td>Online</td>
</tr>
<tr>
<td>importMetadata</td>
<td>Imports metadata for an application.</td>
<td>Online</td>
</tr>
<tr>
<td>purgeMetadata</td>
<td>Purge metadata.</td>
<td>Online</td>
</tr>
</tbody>
</table>

8.2.1 deleteMetadata

Command Category: Application Metadata
Use with WLST: Online
8.2.1.1 Description

Deletes the selected documents from the application repository. When this command is run against repositories that support versioning (that is, database-based repositories), delete is logical and marks the tip version (the latest version) of the selected documents as "deleted" in the MDS repository partition.

You may want to delete metadata when the metadata is moved from one repository to another. In such a case, after you have exported the metadata, you can delete the metadata in the original repository.

8.2.1.2 Syntax

deleteMetadata(application, server, docs [, restrictCustTo] [, excludeAllCust] [, excludeBaseDocs] [, excludeExtendedMetadata] [, cancelOnException] [, applicationVersion] [, tenantName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>The name of the application for which the metadata is to be deleted.</td>
</tr>
<tr>
<td>server</td>
<td>The target server on which this application is deployed.</td>
</tr>
<tr>
<td>docs</td>
<td>A list of comma-separated, fully qualified document names or document name patterns, or both. The patterns can have the following wildcard characters: * and <strong>. The asterisk (*) represents all documents under the current namespace. The double asterisk (</strong>) represents all documents under the current namespace and also recursively includes all documents in subnamespaces. For example, &quot;/oracle/*&quot; will include all documents under &quot;/oracle/&quot; but not include documents under &quot;/oracle/mds/&quot;. As another example, &quot;/oracle/**&quot; will include all documents under &quot;/oracle/&quot; and also under &quot;/oracle/mds/&quot; and any other documents further in the namespace chain.</td>
</tr>
</tbody>
</table>
| restrictCustTo      | Optional. Valid values are percent (%) or a list of comma-separated customization layer names used to restrict the delete operation to delete only customization documents that match the specified customization layers. Each customization layer name can contain, within a pair of brackets, optional customization layer values and value patterns separated by commas. For example:
  restrictCustTo="user[scott]"
  restrictCustTo="site[sitel],user[scott]"
  restrictCustTo="site[sitel, %_2],user[scott, m%]"
  If you do not specify this argument, only customization classes declared in the cust-config element of adf-config.xml are deleted. If there is no cust-config element declared in adf-config.xml, all customization classes are deleted. If you specify percent (%) as the value of this argument, all customizations are deleted, whether or not they are declared in the cust-config element of adf-config.xml. Use this option to delete all customizations or a subset of declared customizations. You can also use this option to delete customizations from customization classes that are not declared in the cust-config element of adf-config.xml. |
### 8.2.1.3 Examples

The following example deletes metadata files under the package `mypackage` from `mdsApp` deployed in the server `server1`:

```
wls:/weblogic/serverConfig> deleteMetadata(application='mdsapp',
    server='server1', docs='/mypackage/**')
Executing operation: deleteMetadata.
"deleteMetadata" operation completed. Summary of "deleteMetadata" operation is:
List of documents successfully deleted:
/mypackageljobs.xml
/mypackagemo.xml
/mypackagemdssys/cust/site/site1/jobs.xml
/mypackagemdssys/cust/site/site1/mo.xml
4 documents successfully deleted.
```

The following example deletes metadata files under the package `mypackage` from `mdsApp` deployed in the server `server1` and excludes extended metadata and all customizations:

```
wls:/weblogic/serverConfig> deleteMetadata(application='mdsapp',
    server='server1', docs='/mypackage/**',
    excludeExtendedMetadata='true',
    excludeAllCust='true',
    cancelOnException='false')
Executing operation: deleteMetadata.
"deleteMetadata" operation completed. Summary of "deleteMetadata" operation is:
List of documents successfully deleted:
/mypackageljobs.xml
/mypackagemo.xml
2 documents successfully deleted.
```

The following example deletes metadata files belonging to tenant `tenant1` under the package `mypackage` from the application `appl` deployed in the server `server1`:

```
wls:/weblogic/serverConfig> deleteMetadata(application='appl',
    server='server1',
    docs='/mypackage/**',
    tenantName='tenant1')
Executing operation: deleteMetadata.
```
8.2.2 exportMetadata

Command Category: Application Metadata

Use with WLST: Online

8.2.2.1 Description

Exports application metadata. Use this command and the importMetadata command to transfer application metadata from one server location (for example, testing) to another server location (for example, production).

This command exports application metadata including customizations. However, by default, only those customizations from customization classes that are defined in the cust-config element of adf.config.xml are exported. To export customizations from customization classes not declared, use the restrictCustTo option.

8.2.2.2 Syntax

```plaintext
exportMetadata(application, server, toLocation [, docs]
[, restrictCustTo] [, excludeCustFor] [, excludeAllCust] [, excludeBaseDocs]
[, excludeExtendedMetadata] [, excludeSeededDocs]
[, fromLabel][, toLabel] [, applicationVersion] [, remote] [, tenantName])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>The name of the application from which the metadata is to be exported.</td>
</tr>
<tr>
<td>server</td>
<td>The target server on which this application is deployed.</td>
</tr>
<tr>
<td>toLocation</td>
<td>The target directory or archive file (.jar, .JAR, .zip or .ZIP) to which documents selected from the source partition will be transferred. If you export to a directory, the directory must be a local or network directory or file where the application is physically deployed. If you export to an archive, the archive can be located on a local or network directory or file where the application is physically deployed, or on the system on which you are executing the command. If the location does not exist in the file system, a directory will be created except that when the names ends with .jar, .JAR, .zip or .ZIP, an archive file will be created. If the archive file already exists, the exportMetadata operation will overwrite the file. This argument can be used as temporary file system for transferring metadata from one server to another. For more information, see “Moving Metadata from a Test System to a Production System” in the Oracle Fusion Middleware Administrator’s Guide.</td>
</tr>
</tbody>
</table>
### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>docs</td>
<td>Optional. A list of comma-separated, fully qualified document names or document name patterns, or both. The patterns can have the following wildcard characters: * and <strong>. This argument defaults to &quot;/</strong>&quot;, which exports all the metadata in the repository. The asterisk (<em>) represents all documents under the current namespace. The double asterisk (**) represents all documents under the current namespace and also recursively includes all documents in subnamespaces. For example, &quot;/oracle/</em>&quot; will include all documents under &quot;/oracle/&quot; but not include documents under &quot;/oracle/mds/&quot;. &quot;/oracle/**&quot; will include all documents under &quot;/oracle/&quot; and also under &quot;/oracle/mds/&quot; and any other documents further in the namespace chain.</td>
</tr>
</tbody>
</table>
| restrictCustTo     | Optional. Valid values are percent (%) or a list of comma-separated customization layer names used to restrict the export operation to export only customization documents that match the specified customization layers. Each customization layer name can contain, within a pair of brackets, optional customization layer values and value patterns separated by commas. For example:  

```
restrictCustTo="user[scott]"
restrictCustTo="site[sit1],user[scott]"
restrictCustTo="site[sit1, %_2],user[scott, m%]"
```

If you do not specify this argument, only customization classes declared in the cust-config element of adf-config.xml are exported. If there is no cust-config element declared in adf-config.xml, all customization classes are exported.  

If you specify percent (%) as the value of this argument, all customizations are exported, whether or not they are declared in the cust-config element of adf-config.xml. Use this option to export all customizations or a subset of declared customizations. You can also use this option to export customizations from customization classes that are not declared in the cust-config element of adf-config.xml. This argument is ignored if the excludeAllCust argument is also specified. |
| excludeCustFor     | Optional. A list of comma-separated customization layer names used to restrict the export operation to exclude customization documents that match the specified customization layers from being exported. This argument is ignored if the excludeAllCust argument is also specified. |
| excludeAllCust     | Optional. A Boolean value (true or false) that specifies whether or not to export all customization documents. This argument defaults to false. This argument overrides the restrictCustTo and excludeCustFor arguments. |
| excludeBaseDocs    | Optional. A Boolean value (true or false) that specifies whether or not to export base documents. This argument defaults to false. |
| excludeExtendedMetadata | Optional. A Boolean value (true or false) that specifies whether or not to export the Extended Metadata documents. This argument defaults to false. |
8.2.2.3 Examples

The following example exports all metadata files from the application mdsapp deployed in the server server1.

```
wlst:/weblogic/serverConfig> exportMetadata(application='mdsapp',
server='server1',toLocation='/tmp/myrepos',docs='/**')
```

Location changed to domainRuntime tree. This is a read-only tree with DomainMBean as the root.
For more help, use help(domainRuntime)
Executing operation: exportMetadata.
'exportMetadata' operation completed. Summary of "exportMetadata" operation is:
List of documents successfully transferred:
/mypackage/write.xml
/mypackage/write1.xml
/sample1.jspx

The following example exports only the customization documents under the layer user without any base documents from label label1 to label label2:

```
wlst:/weblogic/serverConfig> exportMetadata(application='mdsapp',
server='server1',toLocation='/tmp/myrepos',
restrictCustTo='user',
excludeBaseDocs='true',
fromLabel='label1',
toLabel='label2',
applicationVersion='11.1.1')
```

List of documents successfully transferred:
/mypackage/mdssys/cust/user/user1/write1.xml
/mypackage/mdssys/cust/user/user2/write2.xml
2 documents successfully transferred.

8.2.3 importMetadata

Command Category: Application Metadata
Use with WLST: Online

8.2.3.1 Description
Imports application metadata. Use the exportMetadata command and this command to transfer application metadata from one server location (for example, testing) to another server location (for example, production).

8.2.3.2 Syntax

```java
importMetadata(application, server, fromLocation [, docs]
[, restrictCustTo] [, excludeAllCust] [, excludeBaseDocs]
[, excludeExtendedMetadata] [, excludeUnmodifiedDocs]
[, cancelOnException] [, applicationVersion] [, remote] [, tenantName])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>The name of the application for which the metadata is to be imported.</td>
</tr>
<tr>
<td>server</td>
<td>The target server on which this application is deployed.</td>
</tr>
<tr>
<td>fromLocation</td>
<td>The source directory or archive file from which documents will be selected for transfer. If you exported to a directory, the directory must be a local or network directory or file where the application is physically deployed. If you exported to an archive, the archive can be located on a local or network directory or file where the application is physically deployed, or on the system on which you are executing the command. This argument can be used as a temporary file system location for transferring metadata from one server to another. For more information, see &quot;Moving Metadata from a Test System to a Production System&quot; in the Oracle Fusion Middleware Administrator’s Guide</td>
</tr>
<tr>
<td>docs</td>
<td>Optional. A list of comma-separated, fully qualified document names or document name patterns, or both. The patterns can have the following wildcard characters: * and <strong>. This argument defaults to &quot;/**/&quot;, which imports all of the documents in the repository. The asterisk (*) represents all documents under the current namespace. The double asterisk (</strong>) represents all documents under the current namespace and also recursively includes all documents in subnamespaces. For example, &quot;/oracle/*&quot; will include all documents under &quot;/oracle/&quot; but not include documents under &quot;/oracle/mds/&quot;. &quot;/oracle/**&quot; will include all documents under &quot;/oracle/&quot; and also under &quot;/oracle/mds/&quot; and any other documents further in the namespace chain.</td>
</tr>
<tr>
<td>Argument</td>
<td>Definition</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>restrictCustTo</td>
<td>Optional. Valid values are percent (%) or a list of comma-separated customization layer names used to restrict the import operation to import only customization documents that match the specified customization layers, including customization classes that are not declared in the cust-config element of adf-config.xml. Each customization layer name can contain, within a pair of brackets, optional customization layer values and value patterns separated by commas. For example: restrictCustTo=&quot;user[scott]&quot; restrictCustTo=&quot;site[site1],user[scott]&quot; restrictCustTo=&quot;site[site1, %_2],user[scott, m%]&quot; If you do not specify this argument, only customization classes declared in the cust-config element of adf-config.xml are imported. If there is no cust-config element declared in adf-config.xml, all customization classes are imported. If you specify percent (%) as the value of this argument, all customizations are imported, whether or not they are declared in the cust-config element of adf-config.xml. Use this option to import all customizations or a subset of declared customizations. You can also use this option to export customizations from customization classes that are not declared in the cust-config element of adf-config.xml. This argument is ignored if the excludeAllCust argument is also specified.</td>
</tr>
<tr>
<td>excludeAllCust</td>
<td>Optional. A Boolean value (true or false) that specifies whether or not to import all customization documents. This argument defaults to false. This argument overrides the restrictCustTo argument.</td>
</tr>
<tr>
<td>excludeBaseDocs</td>
<td>Optional. A Boolean value (true or false) that specifies whether or not to import base documents. This argument defaults to false.</td>
</tr>
<tr>
<td>excludeExtendedMetadata</td>
<td>Optional. A Boolean value (true or false) that specifies whether or not to import the Extended Metadata documents. This argument defaults to false.</td>
</tr>
<tr>
<td>excludeUnmodifiedDocs</td>
<td>Optional. A Boolean value (true or false) that specifies whether only changed documents are imported. If you specify true, only changed documents are imported. The default is false.</td>
</tr>
<tr>
<td>cancelOnException</td>
<td>Optional. A Boolean value (true or false) that specifies whether or not to abort the import operation when an exception is encountered. The default is true.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. The application version, if multiple versions of the same application are deployed.</td>
</tr>
<tr>
<td>remote</td>
<td>Optional. A Boolean value (true or false) that specifies whether the archive file is in a location where the application is deployed (false) or on the system on which you are executing the command (true). The default is false.</td>
</tr>
<tr>
<td>tenantName</td>
<td>A unique name identifying the tenant to use for this operation. This argument is required for a multitenant application and is not applicable for a non-multitenant application. For a non-multitenant application, any specified value will be ignored.</td>
</tr>
</tbody>
</table>
8.2.3.3 Example
The following example imports all metadata available in /tmp/myrepos to the application mdsapp deployed in the server server1:

```
wlstalk/weblogic/serverConfig> importMetadata(application='mdsapp', server='server1',
fromLocation='/tmp/myrepos', docs='/**')
```

Executing operation: importMetadata.
"importMetadata" operation completed. Summary of "importMetadata" operation is:
List of documents successfully transferred:
/app1/jobs.xml
/app1/mo.xml
2 documents successfully transferred.

8.2.4 purgeMetadata

Command Category: Application Metadata
Use with WLST: Online

8.2.4.1 Description
Purges the older (non-tip) versions of unlabeled documents from the application’s repository. All unlabeled documents will be purged if they are expired, based on Time-To-Live (the olderThan argument). This command is applicable only for repositories that support versioning, that is, database-based repositories.

8.2.4.2 Syntax
```
purgeMetadata(application, server, olderThan [, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>The name of the application, used to identify the partition in the repository on which the purge operation will be run.</td>
</tr>
<tr>
<td>server</td>
<td>The target server on which this application is deployed.</td>
</tr>
<tr>
<td>olderThan</td>
<td>Document versions that are older than this value (in seconds) will be purged. The maximum value is 2147483647 seconds.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. The application version, if multiple versions of the same application are deployed.</td>
</tr>
</tbody>
</table>

8.2.4.3 Example
The following example purges the document version history for the application mdsapp deployed in the server server1, if the version is older than 10 seconds:

```
wlstalk/weblogic/serverConfig> purgeMetadata('mdsapp', 'server1', 10)
```

Executing operation: purgeMetadata.
Metadata purged: Total number of versions: 10.
Number of versions purged: 0.

8.3 Sandbox Metadata Management Commands

Use the commands in Table 8-4 to manage metadata in a sandbox. A sandbox is a temporary location for testing changes before moving them to a production system. Sandboxes are not visible to most users until they are applied.
8.3.1 exportSandboxMetadata

Command Category: Sandbox Metadata Management

Use with WLST: Online

8.3.1.1 Description

Exports the changes to the metadata from a sandbox on a test system.

You can only use this command with a database-based MDS repository.

8.3.1.2 Syntax

```plaintext
exportSandboxMetadata(application, server, toArchive, sandboxName
[, restrictCustTo] [, applicationVersion] [, remote] [, tenantName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>The name of the application from which the metadata is to be exported.</td>
</tr>
<tr>
<td>server</td>
<td>The target server on which this application is deployed.</td>
</tr>
<tr>
<td>toArchive</td>
<td>The target archive file (.jar, .JAR, .zip or .ZIP) to which the sandbox contents will be transferred. The archive can be located on a local or network directory where the application is physically deployed. If you specify the -remote argument, the archive can be located on the system on which you are executing the command.</td>
</tr>
<tr>
<td>sandboxName</td>
<td>The name of the sandbox to export.</td>
</tr>
</tbody>
</table>
| restrictCustTo | Optional. Valid values are percent (%) or a list of comma-separated customization layer names used to restrict the export operation to export only customization documents that match the specified customization layers. Each customization layer name can contain, within a pair of brackets, optional customization layer values and value patterns separated by commas. For example:
```
restrictCustTo="user[scott]"
restrictCustTo="site[sit1],user[scott]"
restrictCustTo="site[sit1, %_2],user[scott, m%]"
```

If you do not specify this argument or if you specify percent (%) as the value of this argument, all customizations are exported, whether or not they are declared in the cust-config element of adf-config.xml.

Use this option to export all customizations or a subset of declared customizations. You can also use this option to export customizations from customization classes that are not declared in the cust-config element of adf-config.xml.

This argument is ignored if the excludeAllCust argument is also specified.
## Sandbox Metadata Management Commands

8.3.1.3 Example

The following example exports a sandbox from the MDS repository for the application `myapp`:

```shell
wls:/weblogic/serverConfig> exportSandboxMetadata('myapp', 'server1', '/tmp/sandbox1.jar', 'sandbox1')
```

### importSandboxMetadata

#### Command Category: Sandbox Metadata Management

Use with WLST: Online

#### 8.3.2.1 Description

Imports the contents of a sandbox archive to another sandbox in the MDS repository partition of the specified application. It can also update the contents of a given archive to a sandbox in the MDS repository partition of a given application. All customizations are imported, whether or not they are declared in the cust-config element of `adf-config.xml`.

You can only use this command with a database-based MDS repository.

#### 8.3.2.2 Syntax

```shell
cmd = importSandboxMetadata[
    application, server, fromArchive [, forceSBCreation]
    [, useExistingSandbox] [, sandboxName] [, applicationVersion]
    [, remote] [, tenantName])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>application</code></td>
<td>The name of the application for which the metadata is to be imported.</td>
</tr>
<tr>
<td><code>server</code></td>
<td>The target server on which this application is deployed.</td>
</tr>
<tr>
<td><code>fromArchive</code></td>
<td>The source archive file from which documents will be selected for transfer. The archive can be located on a local or network directory where the application is physically deployed. If you specify the <code>-remote</code> argument, the archive can be located on the system on which you are executing the command.</td>
</tr>
</tbody>
</table>
8.3.2.3 Examples

The following example imports the contents of the sandbox sandbox1.jar:

```bash
wls:/weblogic/serverConfig> importSandboxMetadata(application='myapp', 'server1', '/tmp/sandbox1.jar')
```

The following example updates the sandbox sandbox1.jar:

```bash
wls:/weblogic/serverConfig> importSandboxMetadata('myapp', 'server1', '/tmp/sandbox1.jar', useExistingSandbox='true', sandboxName='sandbox1')
```

8.4 Application Label Management Commands

Use the commands in Table 8–5 to manage labels for applications.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>forceSBCreation</td>
<td>Optional. A Boolean value (true or false) that specifies whether the operation will overwrite an existing sandbox with the same name. When the argument is set to true, if the fromArchive argument specifies a sandbox with the same name as one that already exists in the application's partition, the original sandbox is deleted and a new sandbox is created. When the argument is set to false, if a sandbox with the same name exists, an exception is thrown. The default is false.</td>
</tr>
<tr>
<td>useExistingSandbox</td>
<td>Optional. When set to true, the contents of the archive are imported to the sandbox specified with the sandboxName argument. This argument is ignored if there is no value specified for sandboxName. The default is false.</td>
</tr>
<tr>
<td>sandboxName</td>
<td>Optional. The name of the sandbox to update. This argument is ignored if useExistingSandbox is false.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. The application version, if multiple versions of the same application are deployed.</td>
</tr>
<tr>
<td>remote</td>
<td>Optional. A Boolean value (true or false) that specifies whether the archive file is in a location where the application is deployed (false) or on the system on which you are executing the command (true). The default is false.</td>
</tr>
<tr>
<td>tenantName</td>
<td>A unique name identifying the tenant to use for this operation. This argument is required for a multitenant application and is not applicable for a non-multitenant application. For a non-multitenant application, any specified value will be ignored.</td>
</tr>
</tbody>
</table>

**Table 8–5 Application Label Management Commands**

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>createMetadataLabel</td>
<td>Creates a metadata label.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteMetadataLabel</td>
<td>Deletes a metadata label from the repository partition.</td>
<td>Online</td>
</tr>
<tr>
<td>listMetadataLabels</td>
<td>Lists metadata labels in the repository partition.</td>
<td>Online</td>
</tr>
<tr>
<td>promoteMetadataLabel</td>
<td>Promotes the metadata associated with a label to tip.</td>
<td>Online</td>
</tr>
<tr>
<td>purgeMetadataLabels</td>
<td>Deletes the labels matching the specified criteria.</td>
<td>Online</td>
</tr>
</tbody>
</table>
8.4.1 createMetadataLabel

Command Category: Application Label Management
Use with WLST: Online

8.4.1.1 Description
Creates a new label for the documents in the application's repository partition. This command is applicable only for repositories that support versioning.

8.4.1.2 Syntax
createMetadataLabel(application, server, name [, applicationVersion] [, tenantName])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>The name of the application for which a label will be created in the partition configured for this application.</td>
</tr>
<tr>
<td>server</td>
<td>The target server on which this application is deployed. If the application is deployed to multiple Managed Servers in a cluster, you can use the name of any of the server names. You cannot specify multiple server names.</td>
</tr>
<tr>
<td>name</td>
<td>The name of the label to create in the repository partition.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. The application version, if multiple versions of the same application are deployed.</td>
</tr>
<tr>
<td>tenantName</td>
<td>A unique name identifying the tenant to use for this operation. This argument is required for a multitenant application and is not applicable for a non-multitenant application. For a non-multitenant application, any specified value will be ignored.</td>
</tr>
</tbody>
</table>

8.4.1.3 Example
The following example creates the label label1 for the application mdsapp deployed in the server server1:

```
wls:/weblogic/serverConfig> createMetadataLabel('mdsapp','server1','label1')
Executing operation: createMetadataLabel('mdsapp','server1','label1')
Created metadata label 'label1'.
```

8.4.2 deleteMetadataLabel

Command Category: Application Label Management
Use with WLST: Online

8.4.2.1 Description
Deletes a label for the documents in the application's repository partition. This command is applicable only for repositories that support versioning.

8.4.2.2 Syntax
deleteMetadataLabel(application, server, name [, applicationVersion] [, tenantName])
8.4.2.3 Example

The following example deletes the metadata label label1 from the application mdsapp deployed in the server server1:

```
wlst:/weblogic/serverConfig> deleteMetadataLabel('mdsapp','server1','label1')
Executing operation: deleteMetadataLabel.
Deleted metadata label "label1".
```

8.4.3 listMetadataLabels

Command Category:
Use with WLST: Online

8.4.3.1 Description

Lists all of the metadata labels in the application's repository partition. This command is applicable only for repositories that support versioning.

8.4.3.2 Syntax

```
listMetadataLabels(application, server [, applicationVersion] [, tenantName])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>The name of the application for which all of the labels in the repository partition should be listed.</td>
</tr>
<tr>
<td>server</td>
<td>The target server on which this application is deployed. If the application is deployed to multiple Managed Servers in a cluster, you can use the name of any of the server names. You cannot specify multiple server names.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. The application version, if multiple versions of the same application are deployed.</td>
</tr>
<tr>
<td>tenantName</td>
<td>A unique name identifying the tenant to use for this operation. This argument is required for a multitenant application and is not applicable for a non-multitenant application. For a non-multitenant application, any specified value will be ignored.</td>
</tr>
</tbody>
</table>
8.4.3 Example
The following example lists the metadata labels available for the application mdsapp deployed in the server server1:

```
  wls:weblogic/serverConfig> listMetadataLabels('mdsapp', 'server1')
  Executing operation: listMetadataLabels.
  Database Repository partition contains the following labels:
  label2
  label3
```

8.4.4 promoteMetadataLabel
Command Category: Application Label Management
Use with WLST: Online

8.4.4.1 Description
Promotes documents associated with a label to the tip version in the repository. This command is useful to achieve rollback capability. This command is applicable only for repositories that support versioning.

8.4.4.2 Syntax
`promoteMetadataLabel(application, server, name[, applicationVersion][, tenantName])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>The name of the application in whose associated repository the metadata is to be promoted to tip.</td>
</tr>
<tr>
<td>server</td>
<td>The target server on which this application is deployed. If the application is deployed to multiple Managed Servers in a cluster, you can use the name of any of the server names. You cannot specify multiple server names.</td>
</tr>
<tr>
<td>name</td>
<td>The name of the label to promote in the repository partition.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. The application version, if multiple versions of the same application are deployed.</td>
</tr>
<tr>
<td>tenantName</td>
<td>A unique name identifying the tenant to use for this operation. This argument is required for a multitenant application and is not applicable for a non-multitenant application. For a non-multitenant application, any specified value will be ignored.</td>
</tr>
</tbody>
</table>

8.4.4.3 Example
The following example promotes the metadata label label1 to tip in the application mdsapp deployed in the server server1:

```
  wls:weblogic/serverConfig> promoteMetadataLabel('mdsapp', 'server1','label1')
  Executing operation: promoteMetadataLabel.
  Promoted metadata label 'label1' to tip.
```

8.4.5 purgeMetadataLabels
Command Category: Application Label Management
Use with WLST: Online
8.4.5.1 Description
Purges or lists the metadata labels that match the given pattern or age, but does not delete the metadata documents that were part of the label. You can delete the documents by executing the `purgeMetadata` command.

8.4.5.2 Syntax
`purgeMetadataLabels(repository, partition [, namePattern] [, olderThanInMin] [, infoOnly] [, tenantName])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>repository</td>
<td>The name of the MDS repository that contains the partition whose metadata labels will be purged or listed.</td>
</tr>
<tr>
<td>partition</td>
<td>The name of the partition whose metadata labels will be purged or listed.</td>
</tr>
<tr>
<td>namePattern</td>
<td>Optional. A pattern that matches the names of labels. The pattern can contain the following special characters:</td>
</tr>
<tr>
<td></td>
<td>The percent (%) character, which matches any number of characters.</td>
</tr>
<tr>
<td></td>
<td>The underscore (_) character, which matches exactly one arbitrary character.</td>
</tr>
<tr>
<td></td>
<td>The backslash character (), which can be used to escape the percent, the underscore, and the backslash (itself) characters, so they match only %, _, or .</td>
</tr>
<tr>
<td>olderThanInMin</td>
<td>Optional. The age of the labels, in minutes. The default is 525600 (one year).</td>
</tr>
<tr>
<td>infoOnly</td>
<td>Optional. Valid values are true or false. If you set it to true, it does not purge the labels, but lists the labels that match the specified pattern. The default is false.</td>
</tr>
<tr>
<td>tenantName</td>
<td>A unique name identifying the tenant to use for this operation. This argument is required for a multitenant application and is not applicable for a non-multitenant application. For a non-multitenant application, any specified value will be ignored.</td>
</tr>
</tbody>
</table>

8.4.5.3 Examples
The following example lists the labels that match the specified namePattern, but does not delete them:

```sql
wls:/weblogic/serverConfig> purgeMetadataLabels(repository='mds-myRepos', partition='partition1', namePattern='mylabel*', infoOnly='true' )
```

The following example purges the labels that match the specified namePattern and that are older than a year:

```sql
wls:/weblogic/serverConfig> purgeMetadataLabels(repository='mds-myRepos', partition='partition1', namePattern='mylabel*')
```

The following example deletes labels that match the specified namePattern and that are older than 30 minutes:

```sql
wls:/weblogic/serverConfig> purgeMetadataLabels(repository='mds-myRepos', partition='partition1', namePattern='mylabel*', olderThanInMin='30')
```
### 8.5 Application Management Deployment Commands

Use the commands in Table 8–6 to manage deployment.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>getMDSArchiveConfig</td>
<td>Returns an MDSArchiveConfig object.</td>
<td>Offline</td>
</tr>
<tr>
<td>importMAR</td>
<td>Imports an MAR.</td>
<td>Online</td>
</tr>
</tbody>
</table>

#### 8.5.1 getMDSArchiveConfig

Command Category: Application Management Deployment

Use with WLST: Offline

**8.5.1.1 Description**

Returns a handle to the MDSArchiveConfig object for the specified archive. The returned MDSArchiveConfig object's methods can be used to change application and shared repository configuration in an archive.

The MDSArchiveConfig object provides the following methods:

- **setAppMetadataRepository**—This method sets the connection details for the application metadata repository.
  
  If the archive's existing adf-config.xml file does not contain any configuration for the application's metadata repository, then you must provide all necessary arguments to define the target repository. To define a database-based repository, provide the repository, partition, type, and jndi arguments. For a file-based repository, provide the path argument instead of jndi.

  If the adf-config.xml file already contains some configuration for the application’s metadata repository, you can provide only a subset of arguments that you want to change. You do not need to provide all arguments in such a case. However, if the store type is changed, then the corresponding jndi or path argument is required.

- **setAppSharedMetadataRepository**—This method sets the connection details for the shared repository in the application archive that is mapped to specified namespace.

  If the archive's existing adf-config.xml file does not contain any configuration for a shared metadata repository mapped to the specified namespace, you must provide all required arguments (in this case, repository, partition, type, and jndi or path). For a database-based repository, provide the jndi argument. For a file-based repository, path is a required argument.

  If the adf-config.xml file already contains some configuration for a shared metadata repository mapped to the specified namespace and you want to change some specific arguments, you can provide only a subset of those arguments; all others are not needed.

- **save**—If you specify the toLocation argument, then the changes will be stored in the target archive file and the original file will remain unchanged. Otherwise, the changes will be saved in the original file itself.

**8.5.1.2 Syntax**

```
archiveConfigObject = getMDSArchiveConfig(fromLocation)
```
The syntax for `setAppMetadataRepository` is:

```java
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>fromLocation</code></td>
<td>The name of the ear file, including its complete path.</td>
</tr>
</tbody>
</table>

The syntax for `setAppSharedMetadataRepository` is:

```java
archiveConfigObject.setAppSharedMetadataRepository(namespace [, repository] [, partition] [, type] [, jndi] [, path])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>namespace</code></td>
<td>The namespace used for looking up the shared repository to set connection details.</td>
</tr>
<tr>
<td><code>repository</code></td>
<td>Optional. The name of the application's shared repository.</td>
</tr>
<tr>
<td><code>partition</code></td>
<td>Optional. The name of the partition for the application's shared metadata.</td>
</tr>
<tr>
<td><code>type</code></td>
<td>Optional. The type of connection, file or database, to the repository. Valid values are 'File' or 'DB' (case insensitive).</td>
</tr>
<tr>
<td><code>jndi</code></td>
<td>Optional. The JNDI location for the database connection. This argument is required if the type is set to DB. This argument will not be considered if the type is set to File.</td>
</tr>
<tr>
<td><code>path</code></td>
<td>Optional. The location of the file metadata store. This argument is required if the type is set to File. This argument will not be considered if the type is set to DB.</td>
</tr>
</tbody>
</table>

The syntax for `save` is:

```java
archiveConfigObject.save([toLocation])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>toLocation</code></td>
<td>Optional. The file name, including the absolute path to store the changes. If this option is not provided, the changes are written to the archive represented by this configuration object.</td>
</tr>
</tbody>
</table>
8.5.3 Examples
In the following example, if the adf-config.xml file in the archive does not have the application and shared metadata repositories defined, then you should provide the complete connection information.

```java
wls:/offline> archive = getMDSArchiveConfig(fromLocation='/tmp/testArchive.ear')
wls:/offline> archive.setAppMetadataRepository(repository='AppRepos1',
   partition='partition1', type='DB', jndi='mds-jndi1')
wls:/offline> archive.setAppSharedMetadataRepository(namespace='/a',
   repository='SharedRepos1', partition='partition2', type='File',
   path='/temp/dir')
wls:/offline> archive.save()
```

In the following example, if the adf-config.xml file in the archive already has the application and shared metadata repositories defined, all arguments are optional. You can set only the arguments you want to change.

```java
wls:/offline> archive = getMDSArchiveConfig(fromLocation='/tmp/testArchive.ear')
wls:/offline> archive.setAppMetadataRepository(partition='MDS-partition2')
wls:/offline> archive.setAppSharedMetadataRepository(namespace='/a',
   repository='SharedRepos2')
wls:/offline> archive.save(toLocation='/tmp/targetArchive.ear')
```

8.5.2 importMAR
Command Category: Application Management Deployment
Use with WLST: Online

8.5.2.1 Description
Imports the metadata from the MAR that is packaged with the application’s EAR file. If the MAR had already been imported into the partition, the command deletes the previous version and imports the new version.

8.5.2.2 Syntax
importMAR(application, server [, force] [, applicationVersion] )

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>The name of the application for which the metadata is to be imported.</td>
</tr>
<tr>
<td>server</td>
<td>The target server on which this application is deployed.</td>
</tr>
<tr>
<td>force</td>
<td>Optional. A Boolean value (true or false) that specifies whether only changed documents and MARs are imported. For a database-based repository, if you set this argument to false, only new or changed documents from changed MARs are imported. The command creates a label for each MAR for which documents are imported. The label has the following format: <code>postDeploy_application_name_MAR_name_MAR_checksum</code> For a file-based repository, if you set this argument to false, only changed MARs are imported. The command does not compare individual documents. The command creates a file in the repository for each imported MAR. The default is true.</td>
</tr>
</tbody>
</table>
8.5.2.3 Example
The following example imports metadata from the MAR to the application mdsapp:

```
$ wls:/weblogic/serverConfig> importMAR('mdsapp','server1')
Executing operation: importMAR.
'importMAR' operation completed. Summary of "importMAR" operation is:
/appl/jobs.xml
/appl/mo.xml
2 documents successfully transferred.
```

8.6 Multitenancy Management Commands
Use the commands in Table 8–7 to manage tenants.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>deprovisionTenant</td>
<td>Deprovisions a tenant from the metadata store.</td>
<td>Online</td>
</tr>
<tr>
<td>listTenants</td>
<td>Lists the tenants.</td>
<td>Online</td>
</tr>
</tbody>
</table>

8.6.1 deprovisionTenant
Deprovisions a tenant from the metadata store. All metadata associated with the tenant will be removed from the store.

8.6.1.1 Syntax
deprovisionTenant(repository, partition, tenantName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>repository</td>
<td>The name of the repository that contains the tenant.</td>
</tr>
<tr>
<td>partition</td>
<td>The name of the partition that contains the tenant.</td>
</tr>
<tr>
<td>tenantName</td>
<td>A unique name identifying the tenant to use for this operation.</td>
</tr>
</tbody>
</table>

8.6.1.2 Example
The following example deprovisions the tenant with tenantName tenant1:

```
$ wls:/weblogic/serverConfig> deprovisionTenant("mds-myrepos", "part1", "tenant1")
Executing operation: deprovisionTenant.
Tenant 'tenant1' has been deprovisioned.
```

8.6.2 listTenants
Lists all tenants in an MDS Repository partition.

8.6.2.1 Syntax
listTenants(repository, partition)
### 8.6.2.2 Example

The following example lists all tenants in the specified repository and partition:

```bash
wls:/weblogic/serverConfig> listTenants("mds-myrepos", "part1")
Executing operation: listTenants.
0  GLOBAL
1  tenant1
2  tenant2
3  tenant3
```
This chapter describes how to use custom WLST commands in Oracle SOA Suite to deploy, manage, compile, package, test, export, import, partition, and diagnosis SOA composite applications; manage configuration plans; and validate human workflow tasks. Command line syntax, arguments, and command examples are also provided.

**Note:** To use these commands, you must invoke WLST from the Oracle home in which the component has been installed. See "Using Custom WLST Commands" in the Oracle Fusion Middleware Administrator's Guide.

This chapter includes the following sections:

- Section 9.1, "Overview of WSLT Command Categories"
- Section 9.2, "Deployment Commands"
- Section 9.3, "SOA Composite Application Management Commands"
- Section 9.4, "SOA Composite Application Compilation Commands"
- Section 9.5, "SOA Composite Application Packaging Commands"
- Section 9.6, "SOA Composite Application Test Commands"
- Section 9.7, "SOA Composite Application HTTP Client-Based Export and Import Commands"
- Section 9.8, "SOA Composite Application MBean-Based Export and Import Commands"
- Section 9.9, "SOA Composite Application Partition Management Commands"
- Section 9.10, "SOA Composite Application Preconfigured WLDF Watches Command"
- Section 9.11, "Configuration Plan Management Commands"
- Section 9.12, "Task Validation Commands"
- Section 9.13, "SOA Health Check Commands"

For additional details about deployment, configuration plans, and test suites, see Oracle Fusion Middleware Developer’s Guide for Oracle SOA Suite.

### 9.1 Overview of WSLT Command Categories

WLST commands are divided into the categories shown in Table 9-1.
Use the deployment commands, listed in Table 9–2, to deploy and undeploy SOA composite applications.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>sca_deployComposite</td>
<td>Deploy a SOA composite application.</td>
<td>Offline</td>
</tr>
<tr>
<td>sca_undeployComposite</td>
<td>Undeploy a SOA composite application.</td>
<td>Offline</td>
</tr>
</tbody>
</table>

### 9.2 Deployment Commands

9.2.1 **sca_deployComposite**

Command Category: Deployment Commands

Use with WLST: Offline
9.2.1.1 Description
Deploys a SOA composite application to the Oracle WebLogic Server. This command does not package the artifact files of the application for deployment. See Section 9.5, "SOA Composite Application Packaging Commands" for instructions on packaging a SOA composite application.

9.2.1.2 Syntax
sca_deployComposite(serverURL, sarLocation, [overwrite], [user], [password], [forceDefault], [configplan], [partition])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverURL</td>
<td>URL of the server that hosts the SOA Infrastructure application (for example, <a href="http://myhost10:7001">http://myhost10:7001</a>).</td>
</tr>
<tr>
<td>sarLocation</td>
<td>Absolute path to one the following:</td>
</tr>
<tr>
<td></td>
<td>■ SOA archive (SAR) file.</td>
</tr>
<tr>
<td></td>
<td>■ A SAR file is a special JAR file that requires a prefix of sca_ (for example, sca_HelloWorld_rev1.0.jar). The SAR file can be deployed with the deployment commands (such as sca_deployComposite()), but a regular .jar file is not treated as a special SAR file.</td>
</tr>
<tr>
<td></td>
<td>■ ZIP file that includes multiple SARs, metadata archives (MARs), or both.</td>
</tr>
<tr>
<td></td>
<td>■ Enterprise archive (EAR) file that contains a SAR file.</td>
</tr>
<tr>
<td>overwrite</td>
<td>Optional. Indicates whether to overwrite an existing SOA composite application file.</td>
</tr>
<tr>
<td></td>
<td>■ false (default): Does not overwrite the file.</td>
</tr>
<tr>
<td></td>
<td>■ true: Overwrites the file.</td>
</tr>
<tr>
<td>user</td>
<td>Optional. User name to access the composite deployer servlet when basic authentication is configured.</td>
</tr>
<tr>
<td>password</td>
<td>Optional. Password to access the composite deployer servlet when basic authentication is configured.</td>
</tr>
<tr>
<td>forceDefault</td>
<td>Optional. Indicates whether to set the new composite as the default.</td>
</tr>
<tr>
<td></td>
<td>■ true (default): Makes it the default composite.</td>
</tr>
<tr>
<td></td>
<td>■ false: Does not make it the default composite.</td>
</tr>
<tr>
<td>configplan</td>
<td>Optional. Absolute path of a configuration plan to be applied to a specified SAR file or to all SAR files included in the ZIP file.</td>
</tr>
<tr>
<td>partition</td>
<td>Optional. The name of the partition in which to deploy the SOA composite application. The default value is default. If you do not specify a partition, the composite is automatically deployed into the default partition.</td>
</tr>
</tbody>
</table>
**9.2.1.3 Examples**

The following example deploys the **HelloWorld** application.

```
  wls:/mydomain/ServerConfig> sca_deployComposite("http://myhost10:7001",
                                         "/tmp/sca_HelloWorld_rev1.0.jar")
```

The following example deploys the **HelloWorld** application as the default version.

```
  wls:/mydomain/ServerConfig> sca_deployComposite("http://myhost10:7001",
                                         "/tmp/sca_HelloWorld_rev1.0.jar", true)
```

The following example deploys the **HelloWorld** application with a required user name when basic authentication is configured. You are then prompted to provide the password for this user name.

```
  wls:/mydomain/ServerConfig> sca_deployComposite("http://myhost10:7001",
                                         "/tmp/sca_HelloWorld_rev1.0.jar", user="weblogic")
  Password:
```

The following example deploys the **HelloWorld** application and applies the configuration plan named **deployplan.xml**.

```
  wls:/mydomain/ServerConfig> sca_deployComposite("http://myhost10:7001",
                                         "/tmp/sca_HelloWorld_rev1.0.jar", forceDefault=false,
                                         configplan="/tmp/deployplan.xml")
```

The following example deploys the **HelloWorld** ZIP file, which can include multiple SARs, MARs, or both.

```
  wls:/mydomain/ServerConfig> sca_deployComposite("http://myhost:7001",
                                         "/tmp/HelloWorld.zip")
```

The following example deploys the **HelloWorld** application to the **myPartition** partition.

```
  wls:/mydomain/ServerConfig> sca_deployComposite("http://stadp10:7001",
                                         "/tmp/sca_HelloWorld_rev1.0.jar", partition="myPartition")
```

### 9.2.2 sca_undeployComposite

**Command Category:** Deployment Commands

**Use with WLST:** Offline

#### 9.2.2.1 Description

Undeploys a currently deployed SOA composite application.
9.2.2.2 Syntax

`sca_undeployComposite(serverURL, compositeName, revision, [user], [password], [partition])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverURL</td>
<td>URL of the server that hosts the SOA Infrastructure application (for example, <code>http://myhost10:7001</code>).</td>
</tr>
<tr>
<td>compositeName</td>
<td>Name of the SOA composite application.</td>
</tr>
<tr>
<td>revision</td>
<td>Revision ID of the SOA composite application.</td>
</tr>
<tr>
<td>user</td>
<td>Optional. User name to access the composite deployer servlet when basic authentication is configured.</td>
</tr>
<tr>
<td>password</td>
<td>Optional. Password to access the composite deployer servlet when basic authentication is configured.</td>
</tr>
<tr>
<td>partition</td>
<td>Optional. The name of the partition in which the SOA composite application is located. The default value is <code>default</code>. If you do not specify a partition, the <code>default</code> partition is searched for the SOA composite application. However, no other partitions are searched.</td>
</tr>
</tbody>
</table>

9.2.2.3 Examples

The following example undeploys the `HelloWorld` application.

```
wls:/mydomain/ServerConfig> sca_undeployComposite("http://myhost10:7001", "HelloWorld", "1.0")
```

The following example undeploys the `HelloWorld` application with a required user name when basic authentication is configured. You are then prompted to provide the password for this user name.

```
wls:/mydomain/ServerConfig> sca_undeployComposite("http://myhost10:7001", "HelloWorld", "1.0", user="weblogic")
Password:
```

The following example undeploys the `HelloWorld` application in the `myPartition` partition.

```
wls:/mydomain/ServerConfig> sca_undeployComposite("http://stadp10:7001", "HelloWorld", "1.0", partition='myPartition')
```

9.3 SOA Composite Application Management Commands

Use the management commands, listed in Table 9–3, to start, stop, activate, retire, assign a default revision version, and list deployed SOA composite applications.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sca_startComposite</code></td>
<td>Start a previously stopped SOA composite application.</td>
<td>Offline</td>
</tr>
<tr>
<td><code>sca_stopComposite</code></td>
<td>Stop a SOA composite application.</td>
<td>Offline</td>
</tr>
<tr>
<td><code>sca_activateComposite</code></td>
<td>Activate a previously retired SOA composite application.</td>
<td>Offline</td>
</tr>
<tr>
<td><code>sca_retireComposite</code></td>
<td>Retire a SOA composite application.</td>
<td>Offline</td>
</tr>
</tbody>
</table>
**Table 9–3 (Cont.) SOA Composite Application Management Commands for WLST**

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>sca_startComposite</td>
<td>Assign the default revision version to a SOA composite application.</td>
<td>Offline</td>
</tr>
<tr>
<td>sca_getDefaultCompositeRevision</td>
<td>List the revision of the default composite of the given composite series.</td>
<td>Offline</td>
</tr>
<tr>
<td>sca_listDeployedComposites</td>
<td>List the deployed SOA composite applications.</td>
<td>Offline</td>
</tr>
<tr>
<td>sca_listCompositesFailedInServerStart</td>
<td>List the SOA composite applications that failed during server startup.</td>
<td>Offline</td>
</tr>
</tbody>
</table>

### 9.3.1 sca_startComposite

Command Category: Application Management Commands

Use with WLST: Offline

#### 9.3.1.1 Description

Starts a previously stopped SOA composite application.

#### 9.3.1.2 Syntax

`sca_startComposite(host, port, user, password, compositeName, revision, [label], [partition])`

#### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>Hostname of the Oracle WebLogic Server (for example, myhost).</td>
</tr>
<tr>
<td>port</td>
<td>Port of the Oracle WebLogic Server (for example, 7001).</td>
</tr>
<tr>
<td>user</td>
<td>User name for connecting to the running server to get MBean information (for example, weblogic).</td>
</tr>
<tr>
<td>password</td>
<td>Password for the user name.</td>
</tr>
<tr>
<td>compositeName</td>
<td>Name of the SOA composite application.</td>
</tr>
<tr>
<td>revision</td>
<td>Revision of the SOA composite application.</td>
</tr>
<tr>
<td>label</td>
<td>Optional. Label of the SOA composite application. The label identifies the metadata service (MDS) artifacts associated with the application. If the label is not specified, the system finds the latest one.</td>
</tr>
<tr>
<td>partition</td>
<td>Optional. The name of the partition in which the SOA composite application is located. The default value is default. If you do not specify a partition, the default partition is searched for the SOA composite application. However, no other partitions are searched.</td>
</tr>
</tbody>
</table>

#### 9.3.1.3 Example

The following example starts revision 1.0 of the HelloWorld application.

```
wls:/mydomain/ServerConfig> sca_startComposite("myhost", "7001", "weblogic", "welcome1", "HelloWorld", "1.0")
```

The following example starts revision 1.0 of the HelloWorld application in the partition myPartition.
9.3.2 sca_stopComposite

Command Category: Application Management Commands

Use with WLST: Offline

9.3.2.1 Description

Stops a currently running SOA composite application.

9.3.2.2 Syntax

sca_stopComposite(host, port, user, password, compositeName, revision, [label], [partition])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>Hostname of the Oracle WebLogic Server (for example, myhost).</td>
</tr>
<tr>
<td>port</td>
<td>Port of the Oracle WebLogic Server (for example, 7001).</td>
</tr>
<tr>
<td>user</td>
<td>User name for connecting to the running server to get MBean information (for example, weblogic).</td>
</tr>
<tr>
<td>password</td>
<td>Password for the user name.</td>
</tr>
<tr>
<td>compositeName</td>
<td>Name of the SOA composite application.</td>
</tr>
<tr>
<td>revision</td>
<td>Revision of the SOA composite application.</td>
</tr>
<tr>
<td>label</td>
<td>Optional. Label of the SOA composite application. The label identifies the MDS artifacts associated with the application. If the label is not specified, the system finds the latest one.</td>
</tr>
<tr>
<td>partition</td>
<td>Optional. The name of the partition in which the SOA composite application is located. The default value is default. If you do not specify a partition, the default partition is searched for the SOA composite application. However, no other partitions are searched.</td>
</tr>
</tbody>
</table>

9.3.2.3 Example

The following example stops revision 1.0 of the HelloWorld application.

wls:/mydomain/ServerConfig> sca_stopComposite("myhost", "7001", "weblogic", "welcome1", "HelloWorld", "1.0")

The following example stops revision 1.0 of the HelloWorld application in the partition myPartition.

wls:/mydomain/ServerConfig> sca_stopComposite("stadp10", "7001", "weblogic", "weblogic", "HelloWorld", "1.0", partition="myPartition")

9.3.3 sca_activateComposite

Command Category: Application Management Commands

Use with WLST: Offline

9.3.3.1 Description

Activates a retired SOA composite application and its instances. You can then create new instances.
9.3.3.2 Syntax
sca_activateComposite(host, port, user, password, compositeName, revision, [label], [partition])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>Hostname of the Oracle WebLogic Server (for example, myhost).</td>
</tr>
<tr>
<td>port</td>
<td>Port of the Oracle WebLogic Server (for example, 7001).</td>
</tr>
<tr>
<td>user</td>
<td>User name for connecting to the running server to get MBean information (for example, weblogic).</td>
</tr>
<tr>
<td>password</td>
<td>Password for the user name.</td>
</tr>
<tr>
<td>compositeName</td>
<td>Name of the SOA composite application.</td>
</tr>
<tr>
<td>revision</td>
<td>Revision of the SOA composite application.</td>
</tr>
<tr>
<td>label</td>
<td>Optional. Label of the SOA composite application. The label identifies the MDS artifacts associated with the application. If the label is not specified, the system finds the latest one.</td>
</tr>
<tr>
<td>partition</td>
<td>Optional. The name of the partition in which the SOA composite application is located. The default value is default. If you do not specify a partition, the default partition is searched for the SOA composite application. However, no other partitions are searched.</td>
</tr>
</tbody>
</table>

9.3.3.3 Example
The following example activates revision 1.0 of the HelloWorld application.

```
wls:/mydomain/ServerConfig> sca_activateComposite("myhost", "7001", "weblogic", "welcome1", "HelloWorld", "1.0")
```

The following example activates revision 1.0 of the HelloWorld application in the partition myPartition.

```
wls:/mydomain/ServerConfig> sca_activateComposite("stadp10", "7001", "weblogic", "weblogic", "HelloWorld", "1.0", partition="myPartition")
```

9.3.4 sca_retireComposite
Command Category: Application Management Commands
Use with WLST: Offline

9.3.4.1 Description
Stops and retires a SOA composite application and all its running instances. If the process life cycle is retired, you cannot create a new instance. Existing instances are allowed to complete normally.

9.3.4.2 Syntax
sca_retireComposite(host, port, user, password, compositeName, revision, [label], [partition])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>Hostname of the Oracle WebLogic Server (for example, myhost).</td>
</tr>
<tr>
<td>port</td>
<td>Port of the Oracle WebLogic Server (for example, 7001).</td>
</tr>
</tbody>
</table>
9.3.4.3 Example

The following example retires revision 1.0 of the HelloWorld application.

```
wls:/mydomain/ServerConfig> sca_retireComposite("myhost", "7001", "weblogic", "welcome1", "HelloWorld", "1.0")
```

The following example retires revision 1.0 of the HelloWorld application in the partition myPartition.

```
wls:/mydomain/ServerConfig> sca_retireComposite("stadp10", "7001", "weblogic", "weblogic", "HelloWorld", "1.0", partition="myPartition")
```

9.3.5 sca_assignDefaultComposite

Command Category: Application Management Commands

Use with WLST: Offline

9.3.5.1 Description

Sets a SOA composite application revision as the default version. This revision is instantiated when a new request comes in.

9.3.5.2 Syntax

```
sca_assignDefaultComposite(host, port, user, password, compositeName, revision, [partition])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>Hostname of the Oracle WebLogic Server (for example, myhost).</td>
</tr>
<tr>
<td>port</td>
<td>Port of the Oracle WebLogic Server (for example, 7001).</td>
</tr>
<tr>
<td>user</td>
<td>User name for connecting to the running server to get MBean information (for example, weblogic).</td>
</tr>
<tr>
<td>password</td>
<td>Password for the user name.</td>
</tr>
<tr>
<td>compositeName</td>
<td>Name of the SOA composite application.</td>
</tr>
<tr>
<td>revision</td>
<td>Revision of the SOA composite application.</td>
</tr>
</tbody>
</table>
9.3.5.3 Example
The following example sets revision 1.0 of the HelloWorld application as the default version.

```
wlsh://mydomain/ServerConfig> sca_assignDefaultComposite("myhost", "7001", "weblogic", "welcome1", "HelloWorld", "1.0")
```

The following example sets revision 1.0 of the HelloWorld application located in the partition myPartition as the default version.

```
wlsh://mydomain/ServerConfig> sca_assignDefaultComposite("stadp10", "7001", "weblogic", "weblogic", "HelloWorld", "1.0", partition="myPartition")
```

9.3.6 sca_getDefaultCompositeRevision

Command Category: Application Management Commands
Use with WLST: Offline

9.3.6.1 Description
Lists the revision of the default composite of the given composite series.

9.3.6.2 Syntax
```
sca_getDefaultCompositeRevision(host, port, user, password, compositeName, partition)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>Hostname of the Oracle WebLogic Server (for example, myhost).</td>
</tr>
<tr>
<td>port</td>
<td>Port of the Oracle WebLogic Server (for example, 7001).</td>
</tr>
<tr>
<td>user</td>
<td>User name for connecting to the running server to get MBean information (for example, weblogic).</td>
</tr>
<tr>
<td>password</td>
<td>Password for the user name.</td>
</tr>
<tr>
<td>compositeName</td>
<td>Name of the SOA composite application.</td>
</tr>
<tr>
<td>partition</td>
<td>Optional. The name of the partition in which the SOA composite application is located. The default value is default. If you do not specify a partition, the default partition is searched for the SOA composite application. However, no other partitions are searched.</td>
</tr>
</tbody>
</table>

9.3.6.3 Example
The following example returns the revision of the default composite of the given composite series.

```
wlsh://mydomain/ServerConfig> sca_getDefaultCompositeRevision("myhost", "7001", "weblogic", "weblogic", "HelloWorld")
```

The following example returns the revision of the default composite of the given composite series in the partition named myPartition.
9.3.7 **sca_listDeployedComposites**

Command Category: Application Management Commands
Use with WLST: Offline

9.3.7.1 **Description**
Lists all SOA composite applications deployed to the SOA platform.

9.3.7.2 **Syntax**
```
sca_listDeployedComposites(host, port, user, password)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>Hostname of the Oracle WebLogic Server (for example, myhost).</td>
</tr>
<tr>
<td>port</td>
<td>Port of the Oracle WebLogic Server (for example, 7001).</td>
</tr>
<tr>
<td>user</td>
<td>User name for connecting to the running server to get MBean information (for example, weblogic).</td>
</tr>
<tr>
<td>password</td>
<td>Password for the user name.</td>
</tr>
</tbody>
</table>

9.3.7.3 **Example**
The following example lists all the deployed SOA composite applications on the server myhost.
```
wls:/mydomain/ServerConfig> sca_listDeployedComposites('myhost', '7001', 'weblogic', 'welcome1')
```

9.3.8 **sca_listCompositesFailedInServerStart**

Command Category: Application Management Commands
Use with WLST: Offline

9.3.8.1 **Description**
List the SOA composite applications that failed during server startup.

9.3.8.2 **Syntax**
```
sca_listCompositesFailedInServerStart (host, port, user, password)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>Hostname of the Oracle WebLogic Server (for example, myhost).</td>
</tr>
<tr>
<td>port</td>
<td>Port of the SOA server (for example, 7001).</td>
</tr>
<tr>
<td>user</td>
<td>User name for connecting to the running server to get MBean information (for example, weblogic).</td>
</tr>
<tr>
<td>password</td>
<td>Password for the user name.</td>
</tr>
</tbody>
</table>
9.3.8.3 Example

The following example lists the SOA composite applications that failed during server startup.

```
wlst:/mydomain/ServerConfig> sca_listCompositesFailedInServerStart('myhost'
'7001','weblogic', 'welcome1')
```

If a SOA composite application failed, output is displayed in the following format:

```
Following 1 composites failed in SOA server start process:
1. BPELInvokesHello[1.0], partition=default, mode=active, state=on,
isDefault=true, deployedTime=2012-09-28T00:48:13.182-07:00
```

9.4 SOA Composite Application Compilation Commands

Use the compilation commands, listed in Table 9–4, to compile SOA composite applications.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>sca_setProp</td>
<td>Set JVM system properties.</td>
<td>Offline</td>
</tr>
<tr>
<td>sca_compile</td>
<td>Compile a SOA composite application.</td>
<td>Offline</td>
</tr>
</tbody>
</table>

9.4.1 sca_setProp

Command Category: Application Compilation Commands

Use with WLST: Offline

9.4.1.1 Description

Sets JVM system properties. This command can also set secure socket layer (SSL) system properties before using `sca_deployComposite` and `sca_undeployComposite` over SSL.

9.4.1.2 Syntax

```
sca_setProp(propName, propValue)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>propName</td>
<td>Property name.</td>
</tr>
<tr>
<td>propValue</td>
<td>Property value.</td>
</tr>
</tbody>
</table>

9.4.1.3 Example

The following example sets the property name and property value.

```
wls:/mydomain/ServerConfig> sca_setProp("oracle.home",
"/scratch/myusername/beahome/AS11gR1SOA")
```

9.4.2 sca_compile

Command Category: Application Compilation Commands

Use with WLST: Offline
9.4.2.1 Description
Compiles a SOA composite application.

**Note:** The `sca_compile` command requires the `oracle.home` property to find the `ant-sca-compile.xml` script. This must be set once. You can use the `scac_setProp` command or the `oracleHome` property to set a value.

9.4.2.2 Syntax
`sca_compile(composite, [outXml], [error], [appHome], [displayLevel], [oracleHome])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>composite</td>
<td>Absolute path of a composite file in the expanded (unzipped) SAR directory.</td>
</tr>
<tr>
<td>outXml</td>
<td>Optional. Absolute path of an output XML file.</td>
</tr>
<tr>
<td>error</td>
<td>Optional. Absolute path of an error file.</td>
</tr>
<tr>
<td>appHome</td>
<td>Optional. Absolute path of the application home directory. This property is required if you have shared data.</td>
</tr>
<tr>
<td>displayLevel</td>
<td>Optional. The level of information to display. The default value is 1.</td>
</tr>
<tr>
<td>oracleHome</td>
<td>Optional. The <code>oracle.home</code> property.</td>
</tr>
</tbody>
</table>

9.4.2.3 Examples
The following example compiles the `FirstComposite` application.

```bash
wls:/mydomain/ServerConfig> sca_compile("/tmp/FirstComposite_rev1.0/composite.xml", displayLevel=2)
```

The following example compiles the `FirstComposite` application and captures details in the `myout.xml` file. The `error.out` file captures any errors.

```bash
wls:/mydomain/ServerConfig> sca_compile("/tmp/FirstComposite_rev1.0/composite.xml", outXml="/tmp/myout.xml", error="error.out")
```

The following example compiles the `FirstComposite` application. The `oracleHome` property is set to find the `ant-sca-compile.xml` script.

```bash
wls:/mydomain/ServerConfig> sca_compile("/tmp/FirstComposite_rev1.0/composite.xml", displayLevel=2, oracleHome="/scratch/myusername/beahome/AS11gR1SOA")
```

9.5 SOA Composite Application Packaging Commands
Use the packaging command, listed in Table 9–5, to package SOA composite applications into a composite SAR file.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sca_package</code></td>
<td>Package the SOA composite application files into a composite SAR file.</td>
<td>Offline</td>
</tr>
</tbody>
</table>

Oracle SOA Suite Custom WLST Commands 9-13
9.5.1 sca_package

Command Category: Application Packaging Commands

Use with WLST: Offline

9.5.1.1 Description

Packages the SOA composite application files into a composite SAR file. This command performs the following operations:

- Calls `sca_compile` to compile the composite artifacts in ${compositeDir}.
- Calls `javac` to compile any source code under ${compositeDir}/src.
- Replaces the revision in ${compositeDir}/composite.xml.
- Packages the artifacts to create `sca_${compositeName}_rev${revision}.jar` in ${compositeDir}/deploy.

**Note:** The `sca_package` command requires `oracle.home` to find the `ant-sca-package.xml` script. This must be set once. You can use the `scac_setProp` command or `oracleHome` property to set this property.

9.5.1.2 Syntax

`sca_package(compositeDir, compositeName, revision, [appHome], [oracleHome])`

9.5.1.3 Examples

The following example packages the OrderBookingComposite application. The `appHome` property is set because this application uses shared data.

```bash
wls:/mydomain/ServerConfig> sca_package("/tmp/app_data/OrderBookingComposite", "OrderBookingComposite", "1.0", appHome="/tmp/app_data")
```

The following example packages the HelloSOAComposite application.

```bash
wls:/mydomain/ServerConfig> sca_package("/tmp/HelloSOAApplication/HelloSOAComposite", "HelloSOAComposite", "1.0")
```

The following example packages the HelloSOAComposite application. The `oracleHome` property is set to find the `ant-sca-compile.xml` script.

```bash
wls:/mydomain/ServerConfig> sca_package("/tmp/HelloSOAApplication/HelloSOAComposite", "HelloSOAComposite", "1.0", oracleHome="/scratch/myusername/beahome/AS11gR1SOA")
```
9.6 SOA Composite Application Test Commands

Use the SOA composite application test command, listed in Table 9–6, to test a SOA composite application.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>sca_test</td>
<td>Test deployed SOA composite applications.</td>
<td>Offline</td>
</tr>
</tbody>
</table>

9.6.1 sca_test

Command Category: Application Test Commands

Use with WLST: Offline

9.6.1.1 Description

Tests deployed SOA composite applications prior to deployment in a production environment. You create suites of tests in Oracle JDeveloper. The sca_test command calls ant-sca-test.xml.

9.6.1.2 Syntax

sca_test('compositeName', 'revision', 'testsuiteName', 'jndiPropFile', [oracleHome='oracleHome'], [javaHome='javaHome'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>compositeName</td>
<td>Name of the SOA composite application.</td>
</tr>
<tr>
<td>revision</td>
<td>Revision ID of the SOA composite application.</td>
</tr>
<tr>
<td>testsuiteName</td>
<td>Name of the test suite.</td>
</tr>
<tr>
<td>jndiPropFile</td>
<td>Absolute path to the JNDI property file.</td>
</tr>
<tr>
<td>oracleHome</td>
<td>Optional. The oracle.home system property.</td>
</tr>
<tr>
<td>javaHome</td>
<td>Optional. The java.passed.home system property.</td>
</tr>
</tbody>
</table>

9.6.1.3 Examples

The following example runs the OrderBookingMainTestsuite test suite.

wls:/mydomain/ServerConfig> sca_test('OrderBookingComposite', '1.0', 'OrderBookingMainTestsuite', '/tmp/tmp-jndi.properties', oracleHome='/scratch/<user>/beahome/AS11gR1SOA/', javaHome='/scratch/<user>/beahome/jdk160_05')

9.7 SOA Composite Application HTTP Client-Based Export and Import Commands

Use the SOA composite application commands, listed in Table 9–7, to export and import SOA composite applications based on the HTTP client. The SOA Infrastructure must be running to use these commands.
9.7.1 sca_exportComposite

Command Category: Application Export and Import Commands

Use with WLST: Offline

9.7.1.1 Description
Exports a SOA composite application into a SAR file.

9.7.1.2 Syntax
sca_exportComposite(serverURL, updateType, sarFile, compositeName, revision, [user], [password], [partition])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverURL</td>
<td>URL of the server that hosts the SOA Infrastructure application (for example, <a href="http://stabc:8001">http://stabc:8001</a>).</td>
</tr>
<tr>
<td>updateType</td>
<td>Type of postdeployment changes to be exported:</td>
</tr>
<tr>
<td></td>
<td>all: Includes all postdeployment changes.</td>
</tr>
<tr>
<td></td>
<td>property: Includes only property postdeployment changes (binding component properties, composite properties such as audit level settings and payload validation status, and policy attachments).</td>
</tr>
<tr>
<td></td>
<td>runtime: Includes only runtime (rules dictionary and domain value maps (DVMs)) and metadata postdeployment changes.</td>
</tr>
<tr>
<td></td>
<td>none: Exports the original composite without any postdeployment changes (including property changes and runtime changes).</td>
</tr>
<tr>
<td>sarFile</td>
<td>Absolute path of a SAR file to generate (a .jar file that begins with sca__).</td>
</tr>
<tr>
<td>compositeName</td>
<td>Name of the composite to export.</td>
</tr>
<tr>
<td>revision</td>
<td>Revision of the composite to export.</td>
</tr>
<tr>
<td>user</td>
<td>Optional. The user name for accessing the server when basic configuration is configured. Use the following syntax for this argument: user='username'</td>
</tr>
</tbody>
</table>
### 9.7.1.3 Examples

The following example exports the composite without including any postdeployment changes.

```
wlst:/offline/mydomain/ServerConfig> sca_exportComposite('http://stabc:8001',
'none', '/tmp/sca_HelloWorld_rev1.0.jar', 'HelloWorld', '1.0')
```

The following example exports a composite with all postdeployment updates.

```
wlst:/offline/mydomain/ServerConfig> sca_exportComposite('http://stabc:8001',
'all', '/tmp/sca_HelloWorld_rev1.0-all.jar', 'HelloWorld', '1.0')
```

The following example exports a composite with property postdeployment updates.

```
wlst:/offline/mydomain/ServerConfig> sca_exportComposite('http://stabc:8001',
'property', '/tmp/sca_HelloWorld_rev1.0-prop.jar', 'HelloWorld', '1.0')
```

The following example exports a composite with runtime/metadata postdeployment updates.

```
wlst:/offline/mydomain/ServerConfig> sca_exportComposite('http://stabc:8001',
'runtime', '/tmp/sca_HelloWorld_rev1.0-runtime.jar', 'HelloWorld', '1.0')
```

The following example exports a composite in the `myPartition` partition without including any postdeployment updates:

```
wlst:/offline/mydomain/ServerConfig> sca_exportComposite('http://stabc:8001',
'none', '/tmp/sca_HelloWorld_rev1.0.jar', 'HelloWorld', '1.0',
partition='myPartition')
```

### 9.7.2 sca_exportUpdates

**Command Category:** Application Export and Import Commands  

**Use with WLST:** Offline

#### 9.7.2.1 Description

Exports postdeployment changes of a SOA composite application into a JAR file.

#### 9.7.2.2 Syntax

```
sca_exportUpdates(serverURL, updateType, jarFile, compositeName, revision, 
[user], [password], [partition])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverURL</td>
<td>URL of the server that hosts the SOA Infrastructure application (for example, <a href="http://stabc:8001">http://stabc:8001</a>).</td>
</tr>
<tr>
<td>password</td>
<td>Optional. The password for accessing the server when basic configuration is configured. Use the following syntax for this argument: passwords='password'</td>
</tr>
<tr>
<td>partition</td>
<td>Optional. The name of the partition in which the SOA composite application is located. The default value is default.</td>
</tr>
</tbody>
</table>
SOA Composite Application HTTP Client-Based Export and Import Commands

9.7.2.3 Examples

The following example exports all postdeployment updates.

```
wls:/offline/mydomain/ServerConfig> sca_exportUpdates('http://stabc:8001', 'all', '/tmp/all-HelloWorld_rev1.0.jar','HelloWorld', '1.0')
```

The following example exports property postdeployment updates.

```
wls:/offline/mydomain/ServerConfig> sca_exportUpdates('http://stabc:8001', 'property', '/tmp/prop-HelloWorld_rev1.0.jar', 'HelloWorld', '1.0')
```

The following example exports runtime/metadata postdeployment updates.

```
wls:/offline/mydomain/ServerConfig> sca_exportUpdates('http://stabc:8001', 'runtime', '/tmp/runtime-HelloWorld_rev1.0.jar', 'HelloWorld', '1.0')
```

The following example exports postdeployment changes of a composite in the partition myPartition into a JAR file.

```
wls:/offline/mydomain/ServerConfig> sca_exportUpdates(serverURL, updateType, jarFile, compositeName, revision, user=None, password=None, partition='myPartition')
```

9.7.3 sca_importUpdates

Command Category: Application Export and Import Commands

Use with WLST: Offline
9.7.3.1 Description
Imports postdeployment changes of a SOA composite application.

9.7.3.2 Syntax
sca_importUpdates(serverURL, jarFile, compositeName, revision, [user], [password])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>serverURL</td>
<td>URL of the server that hosts the SOA Infrastructure application (for example, <a href="http://stabc:8001">http://stabc:8001</a>).</td>
</tr>
<tr>
<td>jarFile</td>
<td>Absolute path of a JAR file that contains postdeployment changes.</td>
</tr>
<tr>
<td>compositeName</td>
<td>Name of the composite to which the postdeployment changes are imported.</td>
</tr>
<tr>
<td>revision</td>
<td>Revision of the composite to which the postdeployment changes are imported.</td>
</tr>
<tr>
<td>user</td>
<td>Optional. The user name for accessing the server when basic configuration is configured. Use the following syntax for this argument: user='username'</td>
</tr>
<tr>
<td>password</td>
<td>Optional. The password for accessing the server when basic configuration is configured. Use the following syntax for this argument: password='password'</td>
</tr>
<tr>
<td>partition</td>
<td>Optional. The name of the partition in which the SOA composite application is located. The default value is default.</td>
</tr>
</tbody>
</table>

9.7.3.3 Examples
The following example imports postdeployment changes of a SOA composite application.

wls:/ offline/mydomain/ServerConfig> sca_importUpdates('http://stabc:8001', '/tmp/all-HelloWorld_rev1.0.jar', 'HelloWorld', '1.0')

The following example imports postdeployment changes of a composite in the partition myPartition.

wls:/ offline/mydomain/ServerConfig> sca_importUpdates(serverURL, jarFile, compositeName, revision, user=None, password=None, partition='myPartition')

9.7.4 sca_exportSharedData
Command Category: Application Export and Import Commands
Use with WLST: Offline

9.7.4.1 Description
Exports shared data of a given pattern into a JAR file.

9.7.4.2 Syntax
sca_exportSharedData(serverURL, jarFile, pattern, [user], [password])
9.7.4.3 Examples

The following example exports shared data of a given pattern into a JAR file.

```
wls:/offline/mydomain/ServerConfig> sca_exportSharedData('http://stabc:8001',
'/tmp/MySharedData.jar', '/Project1/**')
```

9.7.5 sca_removeSharedData

Command Category: Application Export and Import Commands
Use with WLST: Offline

9.7.5.1 Description
Removes a top-level shared data folder, even if there are composites deployed in the service engine.

9.7.5.2 Syntax

```
sca_removeSharedData(serverURL, folderName, [user], [password])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
</table>
9.7.5.3 Examples
The following example removes the top-level shared data Project1 folder.

    sca_removeSharedData('http://stabc:8001', 'Project1')

9.8 SOA Composite Application MBean-Based Export and Import Commands

Use the deployment commands, listed in Table 9–8, to export and import SOA composite applications on the server-based composite store MBean (CompositeStoreMXBean).

<table>
<thead>
<tr>
<th>Table 9–8 SOA Composite Application Export and Import Commands for WLST Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use this command...</td>
</tr>
<tr>
<td>sca_exportCompositeMb</td>
</tr>
<tr>
<td>sca_exportUpdatesMb</td>
</tr>
<tr>
<td>sca_importUpdatesMb</td>
</tr>
<tr>
<td>sca_exportSharedDataMb</td>
</tr>
</tbody>
</table>

If you use this option, note that the file generated in the export commands and the file read in the import command must be on the host where the server is running (either an Oracle WebLogic Administration Server or a managed SOA server).

The composite store MBean is registered as both a server runtime MBean of the SOA server and as a domain runtime MBean of the Oracle WebLogic Administration Server, which allows the import and export to continue working while SOA servers are down. Only WLST commands are provided for using the composite store MBean; there are no ant commands.

You must run the connect() command to connect to either a SOA server or an Oracle WebLogic Administration Server.

    wls:offline>connect('weblogic', 'password', 't3://stabc:8001')

If you use the domain runtime MBean while the SOA servers are down, you must run the domainRuntime() command.

    wls:offline>connect('weblogic', 'password', 't3://stabc:7001')
    wls:/soainfra/serverConfig>domainRuntime()

9.8.1 sca_exportCompositeMb

Command Category: Application Export and Import Commands
Use with WLST: Online

9.8.1.1 Description
Exports a SOA composite application into a SAR file.
9.8.1.2 Syntax

sca_exportCompositeMb(updateType, sarFile, compositeName, revision)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>updateType</td>
<td>Type of postdeployment changes to be exported: all, property, or runtime.</td>
</tr>
<tr>
<td>sarFile</td>
<td>Absolute path of a SAR file to generate.</td>
</tr>
<tr>
<td>compositeName</td>
<td>Name of the composite to export.</td>
</tr>
<tr>
<td>revision</td>
<td>Revision of the composite to export.</td>
</tr>
</tbody>
</table>

9.8.1.3 Examples

This example exports composite without including any postdeployment changes.

```wls
wls:/mydomain/ServerConfig> sca_exportCompositeMb('none', '/tmp/sca_HelloWorld_rev1.0.jar', 'HelloWorld', '1.0')
```

This example exports a composite with all postdeployment updates.

```wls
wls:/mydomain/ServerConfig> sca_exportCompositeMb('all', '/tmp/sca_HelloWorld_rev1.0-all.jar', 'HelloWorld', '1.0')
```

This example exports a composite with property postdeployment updates.

```wls
wls:/mydomain/ServerConfig> sca_exportCompositeMb('property', '/tmp/sca_HelloWorld_rev1.0-prop.jar', 'HelloWorld', '1.0')
```

This example exports a composite with runtime/metadata postdeployment updates.

```wls
wls:/mydomain/ServerConfig> sca_exportCompositeMb('runtime', '/tmp/sca_HelloWorld_rev1.0-runtime.jar', 'HelloWorld', '1.0')
```

9.8.2 sca_exportUpdatesMb

Command Category: Application Export and Import Commands

Use with WLST: Online

9.8.2.1 Description

Exports postdeployment changes of a SOA composite application into a JAR file.

9.8.2.2 Syntax

sca_exportUpdatesMb(updateType, jarFile, compositeName, revision)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>updateType</td>
<td>Type of postdeployment changes to be exported: all, property, or runtime.</td>
</tr>
<tr>
<td>jarFile</td>
<td>Absolute path of a JAR file to generate.</td>
</tr>
<tr>
<td>compositeName</td>
<td>Name of the composite to export.</td>
</tr>
</tbody>
</table>
9.8.2.3 Examples

The following example exports all postdeployment updates.

```
wls:/mydomain/ServerConfig> sca_exportUpdatesMb('all', '/tmp/all-HelloWorld_rev1.0.jar', 'HelloWorld', '1.0')
```

The following example exports property postdeployment updates.

```
wls:/mydomain/ServerConfig> sca_exportUpdatesMB('property', '/tmp/prop-HelloWorld_rev1.0.jar', 'HelloWorld', '1.0')
```

The following example exports runtime/metadata postdeployment updates.

```
wls:/mydomain/ServerConfig> sca_exportUpdatesMB('runtime', '/tmp/runtime-HelloWorld_rev1.0.jar', 'HelloWorld', '1.0')
```

9.8.3 sca_importUpdatesMb

Command Category: Application Export and Import Commands
Use with WLST: Online

9.8.3.1 Description

Imports postdeployment changes of a SOA composite application.

9.8.3.2 Syntax

```
sca_importUpdatesMb(jarFile, compositeName, revision)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>jarFile</td>
<td>Absolute path of a JAR file that contains postdeployment changes.</td>
</tr>
<tr>
<td>compositeName</td>
<td>Name of the composite to which the postdeployment changes are imported.</td>
</tr>
<tr>
<td>revision</td>
<td>Revision of the composite to which the postdeployment changes are imported.</td>
</tr>
</tbody>
</table>

9.8.3.3 Examples

The following example imports postdeployment changes of a SOA composite application.

```
wls:/mydomain/ServerConfig> sca_importUpdatesMb('/tmp/all-HelloWorld_rev1.0.jar', 'HelloWorld', '1.0')
```

9.8.4 sca_exportSharedDataMb

Command Category: Application Export and Import Commands
Use with WLST: Online

9.8.4.1 Description

Exports shared data of a given pattern into a JAR file.
9.8.4.2 Syntax

sca_exportSharedDataMb(jarFile, pattern)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>jarFile</td>
<td>Absolute path of a JAR file to generate.</td>
</tr>
<tr>
<td>pattern</td>
<td>The file pattern supported by MDS transfer APIs. Use the semicolon delimiter (;) if more than one pattern is specified. Exclude the shared data namespace /apps in the pattern. For example: /Project1/<strong>;/Project2/</strong>. This example exports all documents under /apps/Project1 and /apps/Project2.</td>
</tr>
</tbody>
</table>

9.8.4.3 Examples

This example exports shared data of given pattern into a JAR file.

wls:/mydomain/ServerConfig> sca_exportSharedDataMb('/tmp/MySharedData.jar', '/Project1/**')

9.9 SOA Composite Application Partition Management Commands

Use the deployment commands, listed in Table 9–9, to manage partitions. Partitioning enable you to logically group different revisions of your SOA composite applications into separate sections. This is similar to the concept of domains in the 10.1.x releases of Oracle BPEL Process Manager.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>sca_createPartition</td>
<td>Create a partition.</td>
<td>Online</td>
</tr>
<tr>
<td>sca_deletePartition</td>
<td>Undeploy all SOA composite applications in a partition before deleting the partition.</td>
<td>Online</td>
</tr>
<tr>
<td>sca_startCompositesInPartition</td>
<td>Start all SOA composite applications in a partition.</td>
<td>Online</td>
</tr>
<tr>
<td>sca_stopCompositesInPartition</td>
<td>Stop all SOA composite applications in a partition.</td>
<td>Online</td>
</tr>
<tr>
<td>sca_activateCompositesInPartition</td>
<td>Activate all SOA composite applications in a partition.</td>
<td>Online</td>
</tr>
<tr>
<td>sca_retireCompositesInPartition</td>
<td>Retire all SOA composite applications in a partition.</td>
<td>Online</td>
</tr>
<tr>
<td>sca_listPartitions</td>
<td>List all partitions in the SOA Infrastructure.</td>
<td>Online</td>
</tr>
<tr>
<td>sca_listCompositesInPartition</td>
<td>List all composites in a specific partition.</td>
<td>Online</td>
</tr>
</tbody>
</table>

9.9.1 sca_createPartition

Command Category: Application Partition Management Commands
9.9.1 **Description**
Creates a partition.

9.9.1.2 **Syntax**
`sca_createPartition(partitionName)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partitionName</td>
<td>The name of the partition.</td>
</tr>
</tbody>
</table>

9.9.1.3 **Examples**
This example creates a partition named `myPartition`.

```sh
wls:/mydomain/ServerConfig> sca_createPartition('myPartition')
```

9.9.2 **sca_deletePartition**

Command Category: Application Partition Management Commands
Use with WLST: Online

9.9.2.1 **Description**
Undeploys all composites in a partition before deleting the partition.

9.9.2.2 **Syntax**
`sca_deletePartition(partitionName)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partitionName</td>
<td>The name of the partition.</td>
</tr>
</tbody>
</table>

9.9.2.3 **Examples**
This example undeploys all composites in the `myPartition` partition before deleting the partition.

```sh
wls:/mydomain/ServerConfig> sca_deletePartition('myPartition')
```

9.9.3 **sca_startCompositesInPartition**

Command Category: Application Partition Management Commands
Use with WLST: Online

9.9.3.1 **Description**
Starts all composites in a partition.

9.9.3.2 **Syntax**
`sca_startCompositesInPartition(partitionName)`
### 9.9.3.3 Examples

This example starts all composites in the myPartition partition.

```wls
wls:/mydomain/ServerConfig> sca_startCompositesInPartition('myPartition')
```

### 9.9.4 sca_stopCompositesInPartition

Command Category: Application Partition Management Commands

Use with WLST: Online

#### 9.9.4.1 Description

Stops all composites in a partition.

#### 9.9.4.2 Syntax

```java
sca_stopCompositesInPartition(partitionName)
```

#### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partitionName</td>
<td>The name of the partition.</td>
</tr>
</tbody>
</table>

#### 9.9.4.3 Examples

This example stops all composites in the myPartition partition.

```wls
wls:/mydomain/ServerConfig> sca_stopCompositesInPartition('myPartition')
```

### 9.9.5 sca_activateCompositesInPartition

Command Category: Application Partition Management Commands

Use with WLST: Online

#### 9.9.5.1 Description

Activates all composites in a partition.

#### 9.9.5.2 Syntax

```java
sca_activateCompositesInPartition(partitionName)
```

#### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partitionName</td>
<td>The name of the partition.</td>
</tr>
</tbody>
</table>

#### 9.9.5.3 Examples

This example activates all composites in the myPartition partition.

```wls
wls:/mydomain/ServerConfig> sca_activateCompositesInPartition('myPartition')
```

### 9.9.6 sca_retireCompositesInPartition

Command Category: Application Partition Management Commands

#### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partitionName</td>
<td>The name of the partition.</td>
</tr>
</tbody>
</table>
9.9.6.1 Description
Retires all composites in a partition.

9.9.6.2 Syntax
sca_retireCompositesInPartition(partitionName)

---

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partitionName</td>
<td>The name of the partition.</td>
</tr>
</tbody>
</table>

9.9.6.3 Examples
This example retires all composites in the myPartition partition.
wls:/mydomain/ServerConfig> sca_retireCompositesInPartition('myPartition')

9.9.7 sca_listPartitions
Command Category: Application Partition Management Commands
Use with WLST: Online

9.9.7.1 Description
Lists all partitions in the SOA Infrastructure.

9.9.7.2 Syntax
sca_listPartitions()

9.9.7.3 Examples
This example lists all partitions in the SOA Infrastructure.
wls:/mydomain/ServerConfig> sca_listPartitions()

9.9.8 sca_listCompositesInPartition
Command Category: Application Partition Management Commands
Use with WLST: Online

9.9.8.1 Description
Lists all composites in a partition.

9.9.8.2 Syntax
sca_listCompositesInPartition(partitionName)

---

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>partitionName</td>
<td>The name of the partition.</td>
</tr>
</tbody>
</table>

9.9.8.3 Examples
This example lists all composites in the myPartition partition.
sca_listCompositesInPartition(myPartition)
9.10 SOA Composite Application Preconfigured WLDF Watches Command

Use the diagnostic command, listed in Table 9–10, to enable preconfigured WebLogic Diagnostic Framework (WLDF) watches for deployment, memory, and elapsed time of web service calls. You use WLDF watches and notifications to collect diagnostic data to identify problems. This data enables you to isolate and diagnose faults when they occur.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>sca_createWatches</td>
<td>Enable preconfigured WLDF watches.</td>
<td>Online</td>
</tr>
</tbody>
</table>

9.10.1 sca_createWatches

Command Category: Application Diagnostic Command

Use with WLST: Online

9.10.1.1 Description

Enable preconfigured WLDF watches for deployment, memory, and elapsed time of web service calls.

9.10.1.2 Syntax

sca_createWatches()

Note: There are no arguments to specify with this command.

9.10.1.3 Examples

This example enables the preconfigured WLDF watches for deployment, memory, and elapsed time of web service calls.

sca_createWatches()

9.11 Configuration Plan Management Commands

Use the configuration plan management commands, listed in Table 9–11, to attach, extract, generate, and validate configuration plans for SOA composite applications.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>sca_attachPlan</td>
<td>Attach the configuration plan file to the SOA composite application JAR file.</td>
<td>Offline</td>
</tr>
<tr>
<td>sca_extractPlan</td>
<td>Extract a configuration plan packaged with the JAR file for editing.</td>
<td>Offline</td>
</tr>
<tr>
<td>sca_generatePlan</td>
<td>Generate a configuration plan for editing.</td>
<td>Offline</td>
</tr>
<tr>
<td>sca_validatePlan</td>
<td>Validate the configuration plan.</td>
<td>Offline</td>
</tr>
</tbody>
</table>
9.11.1 sca_attachPlan

Command Category: Configuration Plan Management Commands
Use with WLST: Offline

9.11.1.1 Description
Attaches the configuration plan file to the SOA composite application file. If a plan already exists in the file, it is overwritten with this new plan.

9.11.1.2 Syntax

sca_attachPlan(sar, configPlan, [overwrite], [verbose])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>sar</td>
<td>Absolute path of the SAR file.</td>
</tr>
<tr>
<td>configPlan</td>
<td>Absolute path of the configuration plan file.</td>
</tr>
<tr>
<td>overwrite</td>
<td>Optional. Indicates whether to overwrite an existing configuration plan in the SAR file.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Indicates whether to print more information about the configuration plan attachment.</td>
</tr>
</tbody>
</table>

9.11.1.3 Examples
The following example attaches the configplan.xml configuration plan file to the HelloWorld application.

wlst:/mydomain/ServerConfig> sca_attachPlan("/tmp/sca_HelloWorld_rev1.0.jar", "/tmp/configplan.xml")

The following example overwrites the existing configuration plan with configplan.xml file in the HelloWorld application.

wlst:/mydomain/ServerConfig> sca_attachPlan("/tmp/sca_HelloWorld_rev1.0.jar", "/tmp/configplan.xml", overwrite=true)

9.11.2 sca_extractPlan

Command Category: Configuration Plan Management Commands
Use with WLST: Offline

9.11.2.1 Description
Extracts a configuration plan packaged with the SOA composite application file for editing. This is an optional step. If no plan exists, this is the same as creating a new file with sca_generatePlan.

9.11.2.2 Syntax

sca_extractPlan(sar, configPlan, [overwrite], [verbose])
9.11.2.3 Example
The following example extracts the configplan.xml file for editing from the HelloWorld application.

```
<DOMAIN>/mydomain/ServerConfig> sca_extractPlan("/tmp/sca_HelloWorld_rev1.0.jar", 
"/tmp/configplan.xml")
```

The following example extracts the configplan.xml file for editing from the HelloWorld application. This command also overwrites the existing plan.

```
<DOMAIN>/mydomain/ServerConfig> sca_extractPlan("/tmp/sca_HelloWorld_rev1.0.jar", 
"/tmp/configplan.xml", overwrite=true)
```

9.11.3 sca_generatePlan

Command Category: Configuration Plan Management Commands
Use with WLST: Offline

9.11.3.1 Description
Generates a configuration plan for editing.

9.11.3.2 Syntax

`sca_generatePlan(configPlan, sar, composite, [overwrite], [verbose])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>configPlan</td>
<td>Absolute path of the configuration plan file to be generated.</td>
</tr>
<tr>
<td>sar</td>
<td>Absolute path of the SAR file.</td>
</tr>
<tr>
<td>composite</td>
<td>Absolute path of the composite.xml file in the expanded (unzipped) SAR directory.</td>
</tr>
<tr>
<td>overwrite</td>
<td>Optional. Indicates whether to overwrite an existing configuration plan file:</td>
</tr>
<tr>
<td></td>
<td>false (default): Does not overwrite the plan.</td>
</tr>
<tr>
<td></td>
<td>true: Overwrites the plan.</td>
</tr>
<tr>
<td>verbose</td>
<td>Indicates whether to print more information about plan generation:</td>
</tr>
<tr>
<td></td>
<td>true (default): Prints more information.</td>
</tr>
<tr>
<td></td>
<td>false: Does not print more information.</td>
</tr>
</tbody>
</table>
9.11.3.3 Examples
The following example generates the myplan.xml configuration plan file for the HelloWorld application.

`wls:/mydomain/ServerConfig> sca_generatePlan("/tmp/myplan.xml", sar="/tmp/sca_HelloWorld_rev1.0.jar")`

The following example generates the myplan2.xml configuration plan file for the HelloWorld application. The myplan2.xml file overwrites the existing plan.

`wls:/mydomain/ServerConfig> sca_generatePlan("/tmp/myplan2.xml", composite="/tmp/HelloWorld_rev1.0/composite.xml", overwrite=true)`

9.11.4 sca_validatePlan

Command Category: Configuration Plan Management Commands
Use with WLST: Offline

9.11.4.1 Description
Validates the configuration plan. This command identifies all search and replacement changes to be made during deployment. Use this option for debugging only.

9.11.4.2 Syntax
`sca_validatePlan(reportFile, configPlan, [sar], [composite], [overwrite], [verbose])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>reportFile</td>
<td>Absolute path of the report file to be generated. Validation results are written to this file.</td>
</tr>
<tr>
<td>configPlan</td>
<td>Absolute path of the configuration plan file.</td>
</tr>
<tr>
<td>sar</td>
<td>Optional. The absolute path of the SAR file.</td>
</tr>
<tr>
<td>composite</td>
<td>Optional. The absolute path of the composite.xml file in the expanded (unzipped) SAR directory.</td>
</tr>
<tr>
<td>overwrite</td>
<td>Optional. Indicates whether to overwrite an existing configuration plan file:</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Indicates whether to print more information about configuration plan validation.</td>
</tr>
<tr>
<td></td>
<td>■ false (default): Does not overwrite the plan.</td>
</tr>
<tr>
<td></td>
<td>■ true: Overwrites the plan.</td>
</tr>
<tr>
<td></td>
<td>■ true (default): Prints more information.</td>
</tr>
<tr>
<td></td>
<td>■ false: Does not print more information.</td>
</tr>
</tbody>
</table>

9.11.4.3 Examples
The following example validates the configplan.xml configuration plan file for the HelloWorld application.

`wls:/mydomain/ServerConfig> sca_validatePlan("/tmp/myreport.xml", "/tmp/configplan.xml", sar="/tmp/sca_HelloWorld_rev1.0.jar")`

The following example validates the configplan.xml configuration plan file for the HelloWorld application. The configplan.xml plan overwrites the existing plan.
9.12 Task Validation Commands

Use the task validation command, listed in Table 9–12, to validate human workflow tasks.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>sca_validateTask</td>
<td>Validate a human workflow task.</td>
<td>Offline</td>
</tr>
</tbody>
</table>

### 9.12.1 sca_validateTask

Command Category: Task Validation Commands

Use with WLST: Offline

#### 9.12.1.1 Description

Validates a human workflow task contained in the .task file that you created when designing a human task in the Human Task Editor.

#### 9.12.1.2 Syntax

`sca_validateTask(taskFile, outXml, [displayLevel])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>taskFile</td>
<td>Absolute path to the task definition file (.task).</td>
</tr>
<tr>
<td>outXml</td>
<td>Absolute path to an output XML file.</td>
</tr>
<tr>
<td>displayLevel</td>
<td>Optional. The level of information to display. The default value is 1.</td>
</tr>
</tbody>
</table>

#### 9.12.1.3 Example

The following example validates the WFTaskDefinition.task file of the human task.

`wls:/mydomain/ServerConfig> sca_validateTask("/tmp/WFTaskDefinition.task", "/tmp/out.xml", displayLevel=2)`

9.13 SOA Health Check Commands

Use the SOA Health Check commands, listed in Table 9–13, to configure and execute SOA health checks.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>executeHealthCheckAndGenerateReport</td>
<td>Execute a health check and retrieve the report.</td>
<td>Online</td>
</tr>
<tr>
<td>executeHealthCheckCategoryAndGenerateReport</td>
<td>Execute a health check category and retrieve the report.</td>
<td>Online</td>
</tr>
</tbody>
</table>

wls:/mydomain/ServerConfig> sca_validatePlan("/tmp/myreport.xml", "/tmp/configplan.xml", composite="/tmp/HelloWorld_rev1.0/composite.xml", overwrite=true)
### Table 9–13 (Cont.) SOA Health Check Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>executeHealthChecksAndGenerateReport</td>
<td>Execute the specified health checks and retrieve the report.</td>
<td>Online</td>
</tr>
<tr>
<td>getUnsuccessfulHCRequestIds</td>
<td>Retrieve all unsuccessful run ids in the specified time interval.</td>
<td>Online</td>
</tr>
<tr>
<td>getUnsuccessfulHCRequestResultReport</td>
<td>Retrieve the report corresponding to the specified unsuccessful health check run-id.</td>
<td>Online</td>
</tr>
<tr>
<td>listHealthChecks</td>
<td>List all registered health checks.</td>
<td>Online</td>
</tr>
<tr>
<td>listHealthChecksInCategory</td>
<td>List all health checks in the specified category.</td>
<td>Online</td>
</tr>
<tr>
<td>listHealthCheckCategories</td>
<td>List all health check categories.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAllHCRequestResults</td>
<td>Delete all health check request results.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteHCRequestResults</td>
<td>Delete health check request results gathered during the specified time interval.</td>
<td>Online</td>
</tr>
<tr>
<td>getUnsuccessfulHCResultReport</td>
<td>Retrieve the results report of unsuccessful health check runs that match a filter criteria.</td>
<td>Online</td>
</tr>
<tr>
<td>getHCParameter</td>
<td>Retrieve the current value of a health check parameter.</td>
<td>Online</td>
</tr>
<tr>
<td>setHCParameter</td>
<td>Set the value for a health check parameter.</td>
<td>Online</td>
</tr>
<tr>
<td>resetHCParameter</td>
<td>Reset a health check parameter to its default value.</td>
<td>Online</td>
</tr>
<tr>
<td>getHCParameters</td>
<td>Retrieve the list of health check parameters and their values.</td>
<td>Online</td>
</tr>
<tr>
<td>enableHCStartupRun</td>
<td>Enable or disable the health check startup run.</td>
<td>Online</td>
</tr>
<tr>
<td>setHCStartupRunCategory</td>
<td>Change the category used for the health check startup run.</td>
<td>Online</td>
</tr>
<tr>
<td>getHCStartupRun</td>
<td>Retrieve the health check startup run details.</td>
<td>Online</td>
</tr>
<tr>
<td>createHCPeriodicRun</td>
<td>Create a health check periodic run with the specified name and settings.</td>
<td>Online</td>
</tr>
<tr>
<td>enableHCPeriodicRun</td>
<td>Enable or disable a health check periodic run.</td>
<td>Online</td>
</tr>
<tr>
<td>setHCPeriodicRunInterval</td>
<td>Change the interval for a periodic run.</td>
<td>Online</td>
</tr>
<tr>
<td>setHCPeriodicRunSchedule</td>
<td>Change the schedule for a periodic run.</td>
<td>Online</td>
</tr>
<tr>
<td>setHCPeriodicRunCategory</td>
<td>Change the category of a periodic run to the specified category.</td>
<td>Online</td>
</tr>
<tr>
<td>getHCPeriodicRun</td>
<td>Retrieve settings for the specified periodic run.</td>
<td>Online</td>
</tr>
<tr>
<td>getHCPeriodicRuns</td>
<td>Retrieve details of all health check periodic runs.</td>
<td>Online</td>
</tr>
<tr>
<td>removeHCPeriodicRun</td>
<td>Delete a health check periodic run.</td>
<td>Online</td>
</tr>
<tr>
<td>getDisabledHealthChecks</td>
<td>Retrieve the list of disabled health checks.</td>
<td>Online</td>
</tr>
<tr>
<td>addDisabledHealthCheck</td>
<td>Disable a health check.</td>
<td>Online</td>
</tr>
</tbody>
</table>
Table 9–13 (Cont.) SOA Health Check Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>removeDisabledHealthCheck</td>
<td>Re-enable a disabled health check.</td>
<td>Online</td>
</tr>
</tbody>
</table>

9.13.1 executeHealthCheckAndGenerateReport

Command Category: SOA Health Check Commands

Use with WLST: Online

9.13.1.1 Description

Executes the specified health check and retrieves a report of the run.

9.13.1.2 Syntax

executeHealthCheckAndGenerateReport(hcName, fileName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>hcName</td>
<td>Name of the health check to execute.</td>
</tr>
<tr>
<td>fileName</td>
<td>Name and path of the file used to store the output report.</td>
</tr>
</tbody>
</table>

9.13.1.3 Example

The following example executes the DatasourceCheck health check and stores the result in /tmp/myfile.

```
wlst:/mydomain/ServerConfig> executeHealthCheckAndGenerateReport('DatasourceCheck', '/tmp/myfile')
```

9.13.2 executeHealthCheckCategoryAndGenerateReport

Command Category: SOA Health Check Commands

Use with WLST: Online

9.13.2.1 Description

Executes the specified health check category and retrieves a report of the run.

9.13.2.2 Syntax

executeHealthCheckCategoryAndGenerateReport(categoryName, fileName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>categoryName</td>
<td>Name of the health check category to execute.</td>
</tr>
<tr>
<td>fileName</td>
<td>Name and path of the file used to store the output report.</td>
</tr>
</tbody>
</table>

9.13.2.3 Example

The following example executes the Startup health check category and stores the result in /tmp/myfile.

```
wlst:/mydomain/ServerConfig> executeHealthCheckCategoryAndGenerateReport('Startup', '/tmp/myfile')
```
9.13.3 executeHealthChecksAndGenerateReport

Command Category: SOA Health Check Commands

Use with WLST: Online

9.13.3.1 Description
Executes the specified health checks and retrieves a report of the run.

9.13.3.2 Syntax
executeHealthChecksAndGenerateReport(hcNames, fileName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>hcNames</td>
<td>Names of the health checks to execute.</td>
</tr>
<tr>
<td>fileName</td>
<td>Name and path of the file used to store the output report.</td>
</tr>
</tbody>
</table>

9.13.3.3 Example

wls:/mydomain/ServerConfig>
executeHealthChecksAndGenerateReport(['DatasourceCheck', 'SOALibraryCheck'], '/tmp/myfile')

9.13.4 getUnsuccessfulHCRequestIds

Command Category: SOA Health Check Commands

Use with WLST: Online

9.13.4.1 Description
Retrieves all unsuccessful run ids in the specified time interval.

9.13.4.2 Syntax
getUnsuccessfulHCRequestIds(startTime, endTime)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>startTime</td>
<td>Start time of the interval.</td>
</tr>
<tr>
<td>endTime</td>
<td>End time of the interval.</td>
</tr>
</tbody>
</table>

9.13.4.3 Example

The following example retrieves all unsuccessful run ids between 1:00 pm and 3:00 pm on 17th June, 2014.

wls:/mydomain/ServerConfig> getUnsuccessfulHCRequestIds('Jun 17, 2014 1:00:00 PM', 'Jun 17, 2014 3:00:00 PM')

9.13.5 getUnsuccessfulHCRequestResultReport

Command Category: SOA Health Check Commands

Use with WLST: Online

9.13.5.1 Description
Retrieves the report corresponding to the specified unsuccessful health check run-id.
9.13.5.2 Syntax
getUnsuccessfulHCRequestResultReport(id, fileName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Health check run id.</td>
</tr>
<tr>
<td>fileName</td>
<td>Name and path of the file used to store the output report.</td>
</tr>
</tbody>
</table>

9.13.5.3 Example
The following example retrieves the report corresponding to run id 2 and stores it in /tmp/myfile.

wls:/mydomain/ServerConfig> getUnsuccessfulHCRequestResultReport('2', '/tmp/myfile')

9.13.6 listHealthChecks
Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.6.1 Description
Lists all registered health checks.

9.13.6.2 Syntax
listHealthChecks()

9.13.6.3 Example
The following example lists the registered health checks.

wls:/mydomain/ServerConfig> listHealthChecks()

Successfully executed the command. Registered health checks are:
-------------------------------------------------------------------
[DatasourceCheck]
[SOAAppCheck]
[JDBCpoolsCheck]
[SOALibraryCheck]
[BPELAsyncRequestCheck]
[BPELComponentInstanceCheck]
[EDNMessageCheck]
[ServiceCheck]
[MemoryCheck]
-------------------------------------------------------------------

9.13.7 listHealthChecksInCategory
Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.7.1 Description
Lists all health checks in the specified category.
9.13.7.2 Syntax

listHealthChecksInCategory(categoryName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>categoryName</td>
<td>Name of the health check category.</td>
</tr>
</tbody>
</table>

9.13.7.3 Example

The following example lists all health checks under the Startup category.

```
wlst:/mydomain/ServerConfig> listHealthChecksInCategory('Startup')
```

Successfully executed the command. Registered health checks in the given category are:

```
---------------------------------------------------------------------------------
---
[DatasourceCheck]
[SOAAppCheck]
[JDBCpoolsCheck]
[SOALibraryCheck]
[ServiceCheck]
[MemoryCheck]
---------------------------------------------------------------------------------
---
```

9.13.8 listHealthCheckCategories

Command Category: SOA Health Check Commands

Use with WLST: Online

9.13.8.1 Description

Lists all health check categories.

9.13.8.2 Syntax

listHealthCheckCategories()

9.13.8.3 Example

The following example lists the health check categories:

```
wls:/mydomain/ServerConfig> listHealthCheckCategories()
```

Successfully executed the command. Registered health check categories are:

```
-----------------------------------------------------------------------------
[Startup]
[Quick]
[Intermediate]
[Extensive]
[Config]
-----------------------------------------------------------------------------
```

9.13.9 deleteAllHCRequestResults

Command Category: SOA Health Check Commands
9.13.9.1 Description
Deletes all health check request results.

9.13.9.2 Syntax
deleteAllHCRequestResults()

9.13.9.3 Example
The following example deletes all health check request results:
```
wls:/mydomain/ServerConfig> deleteAllHCRequestResults()
Successfully executed the command.
```

9.13.10 deleteHCRequestResults
Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.10.1 Description
Deletes health check request results gathered during the specified time interval.

9.13.10.2 Syntax
deleteHCRequestResults(startTime, endTime)

9.13.10.3 Example
The following example deletes health check request results between 1:00 pm and 3:00 pm on 17th June, 2014.
```
wls:/mydomain/ServerConfig> deleteHCRequestResults('Jun 17, 2014 1:00:00 PM', 'Jun 17, 2014 3:00:00 PM')
```

9.13.11 getUnsuccessfulHCResultReport
Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.11.1 Description
Retrieves the results report of unsuccessful health check runs that match the specified filter criteria.

If stateNames is not passed, then all states are used. If hcName is not passed, then all health checks are considered. If fileName is not passed, a report called hcResultReport.html is generated in the current working directory.
9.13.11.2 Syntax
getUnsuccessfulHCResultReport(startTime, endTime, stateNames=[], hcName=None, fileName='hcResultReport.html')

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>startTime</td>
<td>Start time of the interval.</td>
</tr>
<tr>
<td>endTime</td>
<td>End time of the interval.</td>
</tr>
<tr>
<td>stateNames</td>
<td>Optional. State names (for example, ['Success'])</td>
</tr>
<tr>
<td>hcName</td>
<td>Optional. Name of the health check for which unsuccessful result reports are to be retrieved.</td>
</tr>
<tr>
<td>fileName</td>
<td>Optional. Name and path of the file in which the output report is to be stored.</td>
</tr>
</tbody>
</table>

9.13.11.3 Example
wls:/mydomain/ServerConfig> getUnsuccessfulHCResultReport('Jan 1, 1970 0:00:00 AM', 'Jul 22, 2014 8:00:00 PM', ['Success'], 'DatasourceCheck', '/tmp/myfile')

9.13.12 getHCParameter
Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.12.1 Description
Retrieves the current value of the specified health check parameter.

9.13.12.2 Syntax
getHCParameter(paramName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramName</td>
<td>Name of the health check parameter to check.</td>
</tr>
</tbody>
</table>

9.13.12.3 Example
wls:/mydomain/ServerConfig> getHCParameter("MemoryCheck.memThreshold")

Health Check Parameter:
----------------------------
Name : MemoryCheck.memThreshold
Value : 25

9.13.13 setHCParameter
Command Category: SOA Health Check Commands
Use with WLST: Online
9.13.13.1 **Description**
Sets the specified value for a health check parameter. This command overrides the default value of the health check parameter.

9.13.13.2 **Syntax**
```
setHCParameter(paramName, paramValue)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramName</td>
<td>Name of the health check parameter to update.</td>
</tr>
<tr>
<td>paramValue</td>
<td>New value of the health check parameter.</td>
</tr>
</tbody>
</table>

9.13.13.3 **Example**
```
wls:/mydomain/ServerConfig> setHCParameter("MemoryCheck.memThreshold", "35")
```

9.13.14 **resetHCParameter**
Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.14.1 **Description**
Resets the specified health check parameter to its default value.

9.13.14.2 **Syntax**
```
resetHCParameter(paramName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>paramName</td>
<td>Name of the health check parameter to update.</td>
</tr>
</tbody>
</table>

9.13.14.3 **Example**
```
wls:/mydomain/ServerConfig> resetHCParameter("MemoryCheck.memThreshold")
```

9.13.15 **getHCParameters**
Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.15.1 **Description**
Retrieves the list of health check parameters and their values.

9.13.15.2 **Syntax**
```
getHCParameters()
```

9.13.15.3 **Example**
```
wls:/mydomain/ServerConfig> getHCParameters()
```

Successfully executed the command. Health check parameters are:
9.13.16 enableHCStartupRun

Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.16.1 Description
Enables or disables the health check startup run.

9.13.16.2 Syntax
enableHCStartupRun(enabledFlag)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>enabledFlag</td>
<td>Boolean value (true or false) used to enable or disable the health check startup run.</td>
</tr>
</tbody>
</table>

9.13.16.3 Example
The following example enables the startup run.

wls:/mydomain/ServerConfig> enableHCStartupRun(true)

9.13.17 setHCStartupRunCategory

Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.17.1 Description
Changes the category used for the health check startup run.

9.13.17.2 Syntax
setHCStartupRunCategory(categoryName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>categoryName</td>
<td>Name of the health check category to be used for the startup run.</td>
</tr>
</tbody>
</table>

9.13.17.3 Example
The following example sets the startup run category to Quick.

wls:/mydomain/ServerConfig> setHCStartupRunCategory('Quick')

JDBCPoolsCheck.ConnectionMaxMillis=2000
EDNMessageCheck.pendingMsgOaooCount=5000
EDNMessageCheck.pendingMsgCount=2000
BPELAsyncRequestCheck.pendingAsyncReqCount=1000
BPELComponentInstanceCheck.faultInstanceCount=500
BPELComponentInstanceCheck.openInstanceCount=1000
JDBCPoolsCheck.TestMaxMillis=1000
MemoryCheck.memThreshold=35
9.13.18 getHCStartupRun

Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.18.1 Description
Retrieves the health check startup run details. The details include the startup run health check category and the status of the run (enabled or disabled).

9.13.18.2 Syntax
getHCStartupRun()

9.13.18.3 Example
The following example sets the startup run category to Quick.

```
wls:/mydomain/ServerConfig> getHCStartupRun()
```

```
Startup Run
--------------------
Category Name : Startup
Enabled       : true
```

9.13.19 createHCPeriodicRun

Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.19.1 Description
Creates a health check periodic run with the specified name and settings.

9.13.19.2 Syntax
createHCPeriodicRun(name, enabledFlag, interval, minuteOfHour, hourOfDay, dayOfWeek, categoryName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the periodic run.</td>
</tr>
<tr>
<td>enabledFlag</td>
<td>Boolean value (true or false) that determines whether the periodic run is enabled or disabled.</td>
</tr>
<tr>
<td>interval</td>
<td>Interval or time period (in minutes) between two periodic runs.</td>
</tr>
<tr>
<td>minuteOfHour</td>
<td>Minute of the hour at which to start the first run. The value -1 stands for immediate.</td>
</tr>
<tr>
<td>hourOfDay</td>
<td>Hour of the day at which to start the first run. The value -1 stands for immediate.</td>
</tr>
<tr>
<td>dayOfWeek</td>
<td>Day of the week at which to start the first run. The value -1 stands for immediate.</td>
</tr>
<tr>
<td>categoryName</td>
<td>Name of the health check category to be used for the periodic run.</td>
</tr>
</tbody>
</table>

9.13.19.3 Example
```
wls:/mydomain/ServerConfig> createHCPeriodicRun('myrun', true, 30, 0, -1, -1, -1,
```
9.13.20 enableHCPeriodicRun

Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.20.1 Description
Enables or disables a health check periodic run.

9.13.20.2 Syntax

\[
\text{enableHCPeriodicRun(name, enabledFlag)}
\]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the periodic run to enable or disable.</td>
</tr>
<tr>
<td>enabledFlag</td>
<td>Boolean value (true or false) that determines whether the periodic run is to be enabled or disabled.</td>
</tr>
</tbody>
</table>

9.13.20.3 Example
The following example enables the myrun periodic health check:

\[
wls:/mydomain/ServerConfig> \text{enableHCPeriodicRun('myrun', true)}
\]

9.13.21 setHCPeriodicRunInterval

Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.21.1 Description
Changes the interval of the specified periodic run.

9.13.21.2 Syntax

\[
\text{setHCPeriodicRunInterval(name, interval)}
\]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the health check periodic run.</td>
</tr>
<tr>
<td>interval</td>
<td>Time interval, in minutes, between successive executions of the specified periodic run.</td>
</tr>
</tbody>
</table>

9.13.21.3 Example
The following example sets the interval to 2 hours for the myrun periodic health check:

\[
wls:/mydomain/ServerConfig> \text{setHCPeriodicRunInterval('myrun', 120)}
\]

9.13.22 setHCPeriodicRunSchedule

Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.22.1 Description
Changes the schedule for the specified periodic run.

9.13.22.2 Syntax
setHCPeriodicRunSchedule(name, minuteOfHour, hourOfDay, dayOfWeek)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the health check periodic run.</td>
</tr>
<tr>
<td>minuteOfHour</td>
<td>Minute of the hour at which to start the first run. The value -1 stands for immediate.</td>
</tr>
<tr>
<td>hourOfDay</td>
<td>Hour of the day at which to start the first run. The value -1 stands for immediate.</td>
</tr>
<tr>
<td>dayOfWeek</td>
<td>Day of the week at which to start the first run. The value -1 stands for immediate.</td>
</tr>
</tbody>
</table>

9.13.22.3 Example
wls:/mydomain/ServerConfig> setHCPeriodicRunSchedule('hourly', 0, -1, -1)

9.13.23 setHCPeriodicRunCategory
Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.23.1 Description
Changes the category of a periodic run to the specified category.

9.13.23.2 Syntax
setHCPeriodicRunCategory(name, categoryName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the health check periodic run.</td>
</tr>
<tr>
<td>categoryName</td>
<td>Name of the health check category to be used for the periodic run.</td>
</tr>
</tbody>
</table>

9.13.23.3 Example
wls:/mydomain/ServerConfig> setHCPeriodicRunCategory('myrun', 'Extensive')

9.13.24 getHCPeriodicRun
Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.24.1 Description
Retrieves details (like category, schedule, and enabled status) for the specified periodic run.
9.13.24.2 Syntax
getHCPeriodicRun(name)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the health check periodic run.</td>
</tr>
</tbody>
</table>

9.13.24.3 Example

wls:/mydomain/ServerConfig> getHCPeriodicRun('myrun')

Periodic Run
----------
Name       : myrun
Category Name  : Extensive
Enabled     : false
Interval    : 2
Day of week : -1
Hour of day : -1
Minute of hour : 0

9.13.25 getHCPeriodicRuns

Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.25.1 Description
Retrieves the details of all the health check periodic runs.

9.13.25.2 Syntax
getHCPeriodicRuns()

9.13.25.3 Example

wls:/mydomain/ServerConfig> getHCPeriodicRuns()

Periodic Run Table
------------------
PeriodicRun: frequently, enabled=false, interval=60, minuteOfHour=7, hourOfDay=-1, dayOfWeek=-1, category=Quick
PeriodicRun: daily, enabled=false, interval=1440, minuteOfHour=15, hourOfDay=5, dayOfWeek=-1, category=Intermediate
PeriodicRun: weekly, enabled=false, interval=10080, minuteOfHour=30, hourOfDay=3, dayOfWeek=1, category=Extensive
PeriodicRun: myrun, enabled=false, interval=2, minuteOfHour=0, hourOfDay=-1, dayOfWeek=-1, category=Extensive
PeriodicRun: myrun2, enabled=false, interval=2, minuteOfHour=0, hourOfDay=-1, dayOfWeek=-1, category=Intermediate
PeriodicRun: myrun3, enabled=false, interval=2, minuteOfHour=0, hourOfDay=-1, dayOfWeek=-1, category=Intermediate
PeriodicRun: myrun4, enabled=false, interval=30, minuteOfHour=0, hourOfDay=-1, dayOfWeek=-1, category=Intermediate
9.13.26 removeHCPeriodicRun

Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.26.1 Description
Deletes the specified health check periodic run.

9.13.26.2 Syntax
removeHCPeriodicRun(name)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the health check periodic run to delete.</td>
</tr>
</tbody>
</table>

9.13.26.3 Example
wls:/mydomain/ServerConfig> removeHCPeriodicRun('myrun')

9.13.27 getDisabledHealthChecks

Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.27.1 Description
Retrieves the list of disabled health checks.

9.13.27.2 Syntax
getDisabledHealthChecks()

9.13.27.3 Example
wls:/mydomain/ServerConfig> getDisabledHealthChecks()

9.13.28 addDisabledHealthCheck

Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.28.1 Description
Disables the specified health check.

9.13.28.2 Syntax
addDisabledHealthCheck(hcName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>hcName</td>
<td>Name of the health check to disable.</td>
</tr>
</tbody>
</table>
9.13.28.3 Example

wls:/mydomain/ServerConfig> addDisabledHealthCheck('DatasourceCheck')

9.13.29 removeDisabledHealthCheck

Command Category: SOA Health Check Commands
Use with WLST: Online

9.13.29.1 Description
Re-enables the disabled health check.

9.13.29.2 Syntax
removeDisabledHealthCheck(hcName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>hcName</td>
<td>Name of the health check to enable.</td>
</tr>
</tbody>
</table>

9.13.29.3 Example

wls:/mydomain/ServerConfig> removeDisabledHealthCheck('DatasourceCheck')
This chapter describes WebLogic Scripting Tool (WLST) commands for Oracle WebCenter Portal. These commands enable you to configure WebCenter Portal or Portal Framework applications and components from the command-line. For additional details about Oracle WebCenter Portal configuration, see Oracle Fusion Middleware Administering Oracle WebCenter Portal.

Notes: To use these commands, you must invoke WLST from the Oracle home in which the component has been installed. See "Using Custom WLST Commands” in the Oracle Fusion Middleware Administrator’s Guide.

Most configuration changes made using these WLST commands are only effective after you restart the Managed Server on which WebCenter Portal or your own Portal Framework application is deployed. There are some exceptions, including WLST commands for External Applications, Portlet Producers, and Lifecycle.

Oracle WebCenter Portal’s custom WLST commands are described in the following sections:

- Section 10.1, "Overview of Oracle WebCenter Portal WLST Command Categories"
- Section 10.2, "General"
- Section 10.3, "Analytics"
- Section 10.4, "Activity Graph"
- Section 10.5, "Activity Stream"
- Section 10.6, "Content Repository"
- Section 10.7, "Discussions and Announcements"
- Section 10.8, "External Applications"
- Section 10.9, "Instant Messaging and Presence"
- Section 10.10, "Identity Store"
- Section 10.11, "Lifecycle"
- Section 10.12, "Mail"
- Section 10.13, "Migration"
- Section 10.14, "Notifications"
Overview of Oracle WebCenter Portal WLST Command Categories

- Section 10.15, "People Connections"
- Section 10.16, "Personal Events"
- Section 10.17, "Personalization"
- Section 10.18, "Portlet Producers"
- Section 10.19, "Proxy Server"
- Section 10.20, "Search - Oracle SES Search"
- Section 10.21, "Search - Oracle SES Search Crawlers"
- Section 10.22, "Search - WebCenter Portal Search"
- Section 10.23, "Worklists"
- Section 10.24, "WebCenter Portal Application"
- Section 10.25, "Upgrade"

10.1 Overview of Oracle WebCenter Portal WLST Command Categories

Oracle WebCenter Portal’s WLST commands are grouped into the several categories (Table 10–1).

Most configuration changes made using these WLST commands are only effective after you restart the Managed Server on which WebCenter Portal or your own Portal Framework application is deployed. The only exceptions are the External Applications, Portlet Producers, and Lifecycle WLST commands.

<table>
<thead>
<tr>
<th>Command Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td>Manage connections for WebCenter Portal and Portal Framework applications.</td>
</tr>
<tr>
<td>Analytics</td>
<td>Manage analytics collector connections and configure the analytics collector (on WC_Utilities).</td>
</tr>
<tr>
<td>Activity Graph</td>
<td>Manage activity graph metadata and provider configuration (on WC_Utilities).</td>
</tr>
<tr>
<td>Activity Stream</td>
<td>Archive and restore activity stream data.</td>
</tr>
<tr>
<td>Content Repository</td>
<td>Manage content repository connections and configure document services.</td>
</tr>
<tr>
<td>Discussions and Announcements</td>
<td>Manage discussions server connections and configure discussions and announcements.</td>
</tr>
<tr>
<td>External Applications</td>
<td>Manage external application connections.</td>
</tr>
<tr>
<td>Instant Messaging and Presence</td>
<td>Manage instant messaging and presence server connections and configure instant messaging and presence.</td>
</tr>
<tr>
<td>Mail</td>
<td>Manage mail server connections and configure mail.</td>
</tr>
<tr>
<td>Notifications</td>
<td>Manage settings for notifications.</td>
</tr>
<tr>
<td>People Connections</td>
<td>Manage profile information.</td>
</tr>
<tr>
<td>Personal Events</td>
<td>Manage personal event server connections.</td>
</tr>
<tr>
<td>Personalization</td>
<td>Manage personalization server connections.</td>
</tr>
<tr>
<td>Portlet Producers</td>
<td>Manage portlet producers.</td>
</tr>
<tr>
<td>Proxy Server</td>
<td>Manage proxy settings for RSS and activity stream.</td>
</tr>
</tbody>
</table>
10.2 General

Use the General commands, listed in Table 10–2, to manage connections, and perform other general tasks.

Configuration changes made using these WLST commands are only effective after restarting the Managed Server on which WebCenter Portal or your own Portal Framework application is deployed. For details, see Oracle Fusion Middleware Administering Oracle WebCenter Portal.

<table>
<thead>
<tr>
<th>Table 10–2 General WLST Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use This Command</strong></td>
</tr>
<tr>
<td>deleteConnection</td>
</tr>
<tr>
<td>getWebCenterConnectionTypes</td>
</tr>
<tr>
<td>setWebCenterServiceFrameworkConfig</td>
</tr>
<tr>
<td>getWebCenterServiceFrameworkConfig</td>
</tr>
<tr>
<td>webcenterErrorOccurred</td>
</tr>
</tbody>
</table>

10.2.1 deleteConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

10.2.1.1 Description

Deletes a named connection currently configured for WebCenter Portal or your own Portal Framework application.

If you use deleteConnection to delete a WSRP or PDK-Java producer connection (instead of using deregisterWSRPProducer or deregisterPDKJavaProducer), unused
secondary connections will remain, which you might want to remove. For example, when you delete a WSRP producer connection, its associated web service connection remains and when you delete a PDK-Java producer connection, its associated URL connection remains.

**Note:** deleteConnection cannot be used to delete connections configured for personalization services. Instead, use deleteWCPSCMISConnection, deleteWCPSActivityGraphConnection, deleteWCPSPeopleConnection, or deleteWCPSCustomConnection.

### 10.2.1.2 Syntax
```
deleteConnection(appName, name, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Connection name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

### 10.2.1.3 Example
The following example deletes a connection configured for WebCenter Portal (webcenter):
```
wls:/weblogic/serverConfig> deleteConnection(appName='webcenter', name='MyConnection')
```

### 10.2.2 setWebCenterServiceFrameworkConfig
Module: Oracle WebCenter Portal
Use with WLST: Online

#### 10.2.2.1 Description
Sets configuration properties for Oracle WebCenter Portal’s services framework, such as the Resource Action Handler class and display as popup properties.

#### 10.2.2.2 Syntax
```
setWebCenterServiceFrameworkConfig(appName, [resourceActionHandlerClassName, resourceActionHandlerDisplayInPopup, server, applicationVersion])
```
WebCenter Portal Custom WLST Commands

10.2.2.3 Example

The following example sets the WebCenter Portal Service Framework Resource Action Handler class to my.company.ResourceActionHandler:

```wls
wls:/wc_domain/domainRuntime>
setWebCenterServiceFrameworkConfig(appName='webcenter',
resourceActionHandlerClassName='my.company.ResourceActionHandler')
```


To effect connection changes, you must restart the managed server on which the application is deployed.

The following example sets only the WebCenter Portal Service Framework Resource Action Handler display as popup value to 1 (true):

```wls
wls:/wc_domain/domainRuntime>
setWebCenterServiceFrameworkConfig(appName='webcenter',
resourceActionHandlerDisplayInPopup=1)
```

Successfully set the WebCenter Portal service framework configuration. Resource Action Handler Display In Popup: true

To effect connection changes, you must restart the managed server on which the application is deployed.

10.2.3 getWebCenterServiceFrameworkConfig

Module: Oracle WebCenter Portal

Use with WLST: Online

10.2.3.1 Description

Returns WebCenter Portal Service Framework configuration property settings, such as:

- `resourceActionHandlerClassName`: Class currently used by the WebCenter Portal Service Framework Resource Action Handler
- `resourceActionHandlerDisplayInPopup`: Indicates whether the Resource Action Handler displays resources in a popup or inline. Valid options are 1 (true) and 0 (false).
10.2.3.2 Syntax

`getWebCenterServiceFrameworkConfig(appName, [server, applicationVersion])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td><code>server</code></td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><code>applicationVersion</code></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.2.3.3 Example

The following example returns the service framework resource action handler class and display as popup properties, for WebCenter Portal (`webcenter`):

```
wlscopy/weblogic/serverConfig>getWebCenterServiceFrameworkConfig(appName='webcenter')
```

Resource Action Handler Class: `my.company.ResourceActionHandler`

Resource Action Handler Display In Popup: `true`

10.2.4 `webcenterErrorOccurred`

Module: Oracle WebCenter Portal

Use with WLST: Online

10.2.4.1 Description

Returns the status of last WebCenter Portal command executed.

Use the `webcenterErrorOccurred` command to determine the status of the last WebCenter Portal command executed. The command returns 1 if an error occurred or 0 otherwise.

10.2.4.2 Syntax

`webcenterErrorOccurred ()`

10.2.4.3 Example

The following example returns 1 if an error occurred:

```
wls:/mydomain/serverConfig> webcenterErrorOccurred()
```

10.2.5 `getWebCenterConnectionTypes`

Module: Oracle WebCenter Portal

Use with WLST: Online
10.2.5.1 Description
Lists all the Oracle WebCenter Portal connection types.

10.2.5.2 Syntax
getWebCenterConnectionTypes(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.2.5.3 Example
The following example returns connection types for WebCenter Portal (webcenter):

wls:/mydomain/serverConfig>getWebCenterConnectionTypes(appName='webcenter')

10.3 Analytics
This section includes the following sub sections:
- Analytics Collector Connections
- Analytics Collector and Cluster Configuration

Analytics Collector Connections
Use the commands listed in Table 10–3 to manage analytics collector connections for WebCenter Portal and Portal Framework applications. Events raised in these portal applications using OpenUsage APIs can be sent to an analytics collector for use by analytics and activity graph.

Connection configuration changes made using these Oracle WebCenter Portal WLST commands are only effective after your restart the Managed Server on which the application is deployed. For details, see Oracle Fusion Middleware Administering Oracle WebCenter Portal.

Table 10–3 Analytics Collector Connection WLST Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>createAnalyticsCollectorConnection</td>
<td>Create a connection to an analytics collector for WebCenter Portal or a Portal Framework application.</td>
<td>Online</td>
</tr>
<tr>
<td>setAnalyticsCollectorConnection</td>
<td>Edit an existing analytics collector connection.</td>
<td>Online</td>
</tr>
</tbody>
</table>
Use the commands listed in Table 10–4 to configure event collection properties for the analytics collector that is deployed on the WC.Utilities managed server.

If you reconfigure the analytics collector or set up clustering, you must restart the managed server on which the analytic collector is deployed (WC.Utilities).

### Table 10–4  Analytics Collector Configuration WLST Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>listAnalyticsCollectorConnections</td>
<td>List all of the analytics collector connections that are configured for WebCenter Portal or a Portal Framework application.</td>
<td>Online</td>
</tr>
<tr>
<td>setDefaultAnalyticsCollectorConnection</td>
<td>Specify the default (or active) analytics collector connection for WebCenter Portal or a Portal Framework application.</td>
<td>Online</td>
</tr>
<tr>
<td>listDefaultAnalyticsCollectorConnection</td>
<td>Return connection details for the analytics collector being used by WebCenter Portal and Portal Framework application.</td>
<td>Online</td>
</tr>
</tbody>
</table>

**Analytics Collector and Cluster Configuration**

Use the commands listed in Table 10–4 to configure event collection properties for the analytics collector that is deployed on the WC.Utilities managed server.

If you reconfigure the analytics collector or set up clustering, you must restart the managed server on which the analytic collector is deployed (WC.Utilities).

### 10.3.1 createAnalyticsCollectorConnection

**Module:** Oracle WebCenter Portal

**Use with WLST:** Online

**10.3.1.1 Description**

Creates a connection to an analytics collector for a named application.

Events raised in WebCenter Portal or a Portal Framework application using OpenUsage APIs can be sent to an analytics collector for use by analytics and activity graph.

While you can register multiple analytics collector connections for an application, only one analytics collector connection is used - the default (or active) connection where default=1.

**10.3.1.2 Syntax**

```java
createAnalyticsCollectorConnection(appName, connectionName, {isUnicast, collectorhost, clusterName, collectorPort, isEnabled, timeout, default, server, applicationVersion})
```
Example

The following example creates a connection named MyAnalyticsCollector for WebCenter Portal (webcenter). Events are sent to a single analytics collector using unicast communication:

```
wls:/weblogic/serverConfig>createAnalyticsCollectorConnection(appName='webcenter', connectionName='MyAnalyticsCollector', isUnicast=1, collectorHost='myhost.com', collectorPort=31314, isEnabled=1, timeout=30,
```
default=1)

The following example creates a connection named MyAnalyticsCollector for
WebCenter Portal. Events are sent to a clustered analytics collector in multicast mode

wls:/weblogic/serverConfig>createAnalyticsCollectorConnection(appName='webcenter',
connectionName='MyAnalyticsCollector', isUnicast=0,
clusterName='collector-cluster',
ccollectorPort=31314, isEnabled=1, timeout=30, default=1)

10.3.2 setAnalyticsCollectorConnection
Module: Oracle WebCenter Portal
Use with WLST: Online

10.3.2.1 Description
Edits an existing analytics collector connection for a named application.

Events raised in WebCenter Portal or a Portal Framework application using
OpenUsage APIs can be sent to an analytics collector for use by analytics and activity

graph.

While you can register multiple analytics collector connections for an application, only
one analytics collector connection is used - the default (or active) connection.

10.3.2.2 Syntax
setAnalyticsCollectorConnection(appName, connectionName, [isUnicast,
collectorHost, clusterName, collectorPort, isEnabled, timeout, default, server,
applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>connectionName</td>
<td>Connection name. The name must be unique across all connection types within application.</td>
</tr>
<tr>
<td>isUnicast</td>
<td>Optional. Specifies whether events are sent to a clustered analytics collector in multicast mode or whether a single analytics collector using unicast communication is required. Valid values are 1 (true) and 0 (false). The default value is 1 (unicast).</td>
</tr>
<tr>
<td>collectorHost</td>
<td>Optional. Host name where the analytics collector is running. The default value is localhost. Only required for unicast communication, that is, where isUnicast=1.</td>
</tr>
<tr>
<td>clusterName</td>
<td>Optional. Name of the cluster where a clustered analytics collector is running. Only required for multicast communication, that is, where isUnicast=0.</td>
</tr>
<tr>
<td>collectorPort</td>
<td>Optional. Port on which the analytics collector listens for events. The default value is 31314.</td>
</tr>
</tbody>
</table>
### 10.3.2.3 Example

The following example edits host and port details for an existing analytics collector connection named `MyAnalyticsCollector`. On this connection, events are sent to a single analytics collector in *unicast* mode:

```
<wls:/weblogic/serverConfig> setAnalyticsCollectorConnection(appName='webcenter',
connectionName='MyAnalyticsCollector', collectorHost='myhost.com',
collectorPort=31314)
```

The following example edits cluster, port, and timeout details for an existing analytics collector connection named `MyAnalyticsCollector`. On this connection, events are sent to a clustered analytics collector in *multicast* mode:

```
<wls:/weblogic/serverConfig> setAnalyticsCollectorConnection(appName='webcenter',
connectionName='MyAnalyticsCollector', collectorHost='myhost.com',
collectorPort=31314, clusterName='collector-cluster', timeout=60)
```

### 10.3.3 listAnalyticsCollectorConnections

**Module:** Oracle WebCenter Portal  
**Use with WLST:** Online

#### 10.3.3.1 Description

Lists connection names and details for all analytics collector connections that are configured for a named application.

#### 10.3.3.2 Syntax

```
listAnalyticsCollectorConnections(appName, [server, applicationVersion])
```
### 10.3.3 Examples

The following example lists connection names and details for all the analytics collector connections that are currently configured for WebCenter Portal (`webcenter`):

```
wlsc:/weblogic/serverConfig> listAnalyticsCollectorConnections(appName='webcenter')
```

```
------------------
MyAnalyticsCollector
------------------
ClusterName/HostName: localhost
Port: 31314
Timeout: 30
Unicast: 1 (true)
Enabled: 1 (true)
------------------
TestAnalyticsCollector
------------------
ClusterName/HostName: localhost
Port: 32456
Timeout: 456
Unicast: 1 (true)
Enabled: 1 (true)
```

### 10.3.4 setDefaultAnalyticsCollectorConnection

**Module**: Oracle WebCenter Portal

**Use with WLST**: Online

#### 10.3.4.1 Description

Specifies the default analytics collector connection for a named application.

The default analytics collector connection is used to send events raised in WebCenter Portal or a Portal Framework application using OpenUsage APIs to an analytics collector for use by analytics and activity graph.

While you can register multiple analytics collector connections for an application, only one analytics collector connection is used-- the default (or active) connection.

#### 10.3.4.2 Syntax

```
setDefaultAnalyticsCollectorConnection(appName, name, [server, applicationVersion])
```
### 10.3.4.3 Example

The following example configures the connection `MyAnalyticsCollector` for events raised in WebCenter Portal (`webcenter`):

```
wlst:/weblogic/serverConfig> setDefaultAnalyticsCollectorConnection
(appName='webcenter', name='MyAnalyticsCollector')
```

The following example resets the default connection name:

```
wlst:/weblogic/serverConfig> setDefaultAnalyticsCollectorConnection
(appName='webcenter', name='')
```

### 10.3.5 listDefaultAnalyticsCollectorConnection

**Module:** Oracle WebCenter Portal

**Use with WLST:** Online

#### 10.3.5.1 Description

Return details about the analytics collector connection that is currently configured for a named application.

While you can register multiple analytics collector connections for an application, only one analytics collector connection is used—the default (or active) connection.

#### 10.3.5.2 Syntax

```
listDefaultAnalyticsCollectorConnection(appName, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>appName</strong></td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td><strong>name</strong></td>
<td>Name of an existing analytics collector connection.</td>
</tr>
<tr>
<td><strong>server</strong></td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, <code>WC_Spaces</code>.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><strong>applicationVersion</strong></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.3.5.3 Examples

The following example returns details about the analytics collector connection that is currently configured for a WebCenter Portal (webcenter):

```
wl:weblogic/serverConfig>listDefaultAnalyticsCollectorConnection(appName='webcenter')
```

```
------------------
MyAnalyticsCollector
------------------
ClusterName/HostName: localhost
Port: 31314
Timeout: 30
Unicast: 1 (true)
Enabled: 1 (true)
------------------
```

10.3.6 setAnalyticsCollectorConfig

Module: Oracle WebCenter Portal

Use with WLST: Online

10.3.6.1 Description

Configure the analytics collector deployed on the WC_Utilities managed server. Additionally, in a clustered environment, use this commands to set cluster settings.

10.3.6.2 Syntax

```
setAnalyticsCollectorConfig(appName, [collectorHost, defaultPort, maxPort, broadcastType, clusterEnabled, clusterName, heartbeatFrequency, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the analytics collector application in which to perform this operation--always analytics-collector.</td>
</tr>
<tr>
<td>collectorHost</td>
<td>Optional. Name of the host on which the analytics collector is running. The default value is localhost.</td>
</tr>
<tr>
<td>defaultPort</td>
<td>Optional. Default port number on which the analytics collector listens. The default value is 31314.</td>
</tr>
<tr>
<td>maxPort</td>
<td>Optional. Highest port number that the analytics collector can use when allocating a listener. This property is mostly used in a clustered environment where more than one collector is running in the same box. Each collector listens for incoming UDP messages on a free port within a given port range. The range is from the default port number to the maxPort number.</td>
</tr>
</tbody>
</table>
| broadcastType  | Optional. Indicates the network channel on which the analytics collector broadcasts a 'heartbeat' to advertise its location to event producers. Valid values are Broadcast and Multicast.  
|                | - Broadcast - use the standard network broadcast channel.                                                                                 |
|                | - Multicast - use a special fixed multicast address.                                                                                     |
The following example changes the default port to 31315:

```
setAnalyticsCollectorConfig(appName='analytics-collector', defaultPort=31315)
```

### 10.3.7 listAnalyticsCollectorConfig

**Module:** Oracle WebCenter Portal  
**Use with WLST:** Online

10.3.7.1 **Description**  
Returns analytics collector settings.

10.3.7.2 **Syntax**  
```
listAnalyticsCollectorConfig(appName, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>appName</strong></td>
<td>Name of the analytics collector application in which to perform this operation—always analytics-collector.</td>
</tr>
<tr>
<td><strong>server</strong></td>
<td>Optional. Name of the managed server where the analytics collector is deployed. For example, WC_Utilities. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><strong>applicationVersion</strong></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.3.7.3 **Examples**  
The following command lists current settings for the analytics collector that is configured for an application named webcenter:
wls:/weblogic/serverConfig> listAnalyticsCollectorConfig(appName='analytics-collector')

This is sample output for a standalone analytics collector:

CollectorHost = localhost
CollectorDefaultPort = 31314
CollectorMaximumPort = 31314
BroadcastType = Multicast
ClusterEnabled =
ClusterName =
ClusterBroadcastFrequency = 55

This is sample output for an analytics collector in a clustered environment:

CollectorHost = localhost
CollectorDefaultPort = 31314
CollectorMaximumPort = 31318
BroadcastType = Multicast
ClusterEnabled = 1
ClusterName = myCluster
ClusterBroadcastFrequency = 55

10.3.8 listAnalyticsEventTypes
Module: Oracle WebCenter Portal
Use with WLST: Online

10.3.8.1 Description
Lists all the events currently registered with the analytics collector.

10.3.8.2 Syntax
listAnalyticsEventTypes(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.3.8.3 Examples
The following command lists all the events currently registered with the analytics collector for use by WebCenter Portal (webcenter):

wls:/weblogic/serverConfig> listAnalyticsEventTypes(appName='webcenter')

Sample output:

{HTTP://WWW.ORACLE.COM/ANALYTICS/WC}DISCUSSION_ANNOUNCEMENTEDIT
{HTTP://WWW.ORACLE.COM/ANALYTICS/WC}DISCUSSION_TOPICDELETE
10.4 Activity Graph

Use the commands listed in Table 10–5 to manage system properties and metadata for activity graph.

Configuration changes made using the setAGProperty WLST command are only effective after your restart the managed server on which the activity graph application is deployed (WC_Utility). For all other commands, configuration changes are effective immediately.

See also, ”Managing Activity Graph” in the Oracle Fusion Middleware Administering Oracle WebCenter Portal.

Table 10–5 Activity Graph WLST Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>exportAGMetadata</td>
<td>Export activity graph metadata definitions to an XML file.</td>
<td>Online</td>
</tr>
<tr>
<td>importAGMetadata</td>
<td>Import activity graph metadata definitions from an XML file.</td>
<td>Online</td>
</tr>
<tr>
<td>exportAGProviderConfiguration</td>
<td>Export provider configuration, for a given provider, to an activity graph metadata definition file.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAllAGMetadata</td>
<td>Delete all the activity graph metadata that is defined for an application.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAGAction</td>
<td>Delete the metadata for an action registered with activity graph.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAGNodeClass</td>
<td>Delete the metadata for a node class registered with activity graph.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAGSimilarityCalculation</td>
<td>Delete the metadata for a similarity calculation registered with activity graph.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAGRankCalculation</td>
<td>Delete the metadata for a rank calculation registered with activity graph.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAGProviderAssignment</td>
<td>Delete the metadata for a provider assignment registered with activity graph.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteAGQRPPRegistration</td>
<td>Delete the metadata for a QRPP registered with activity graph.</td>
<td>Online</td>
</tr>
<tr>
<td>renameAGAction</td>
<td>Change the URN of an action registered with activity graph.</td>
<td>Online</td>
</tr>
<tr>
<td>renameAGNodeClass</td>
<td>Change the URN of a node class registered with activity graph.</td>
<td>Online</td>
</tr>
<tr>
<td>setAGProperty</td>
<td>Set a system property for activity graph.</td>
<td>Online</td>
</tr>
<tr>
<td>getAGProperty</td>
<td>Return the current setting for a given activity graph property.</td>
<td>Online</td>
</tr>
</tbody>
</table>
10.4.1 `exportAGMetadata`

Module: Oracle WebCenter Portal

Use with WLST: Online

10.4.1.1 Description

Exports activity graph metadata definitions to an XML file.

10.4.1.2 Syntax

```
exportAGMetadata(appName, directoryPath, definitionFileName, includeProviderConfigurations, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the activity graph application in which to perform this operation—always <code>activitygraph-engines</code>.</td>
</tr>
<tr>
<td><code>directoryPath</code></td>
<td>Destination directory for the XML file that will be generated. If you specify a directory that does not exist then it will be created.</td>
</tr>
<tr>
<td><code>definitionFileName</code></td>
<td>Name for the XML file that will be generated. If a file with the same name exists in the destination directory then it will be overwritten.</td>
</tr>
<tr>
<td><code>includeProviderConfigurations</code></td>
<td>Determines whether the export includes provider configuration metadata. Valid values are 1 (true) and 0 (false). Provider configurations are a subset of activity graph metadata that you may want to manage separately from the other metadata.</td>
</tr>
<tr>
<td><code>server</code></td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Utilities</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><code>applicationVersion</code></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the activity graph application is deployed.</td>
</tr>
</tbody>
</table>

10.4.1.3 Example

The following example exports activity graph metadata definitions to an XML file named `ag-metadata.xml`, at the specified location:

```
wls:/weblogic/serverConfig> exportAGMetadata(appName='activitygraph-engines', directoryPath='/scratch/myAGmetadata', definitionFileName='ag-metadata.xml', includeProviderConfigurations=1)
```

10.4.2 `importAGMetadata`

Module: Oracle WebCenter Portal

Use with WLST: Online
10.4.2.1 Description
Imports activity graph metadata definitions from an XML file.
On import, new activity graph metadata definitions are created on the target and existing definitions are overwritten.

10.4.2.2 Syntax
importAGMetadata(appName, definitionFilePath, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the activity graph application in which to perform this operation—always activitygraph-engines.</td>
</tr>
<tr>
<td>definitionFilePath</td>
<td>Relative path to the XML file containing metadata definitions. For example, metadata/import-metadata.xml.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Utilities. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the activity graph application is deployed.</td>
</tr>
</tbody>
</table>

10.4.2.3 Example
The following example imports activity graph metadata definitions from a file name import-metadata.xml:

```bash
wls:/weblogic/serverConfig> importAGMetadata(appName='activitygraph-engines', definitionFilePath='metadata/import-metadata.xml')
```

10.4.3 exportAGProviderConfiguration
Module: Oracle WebCenter Portal
Use with WLST: Online

10.4.3.1 Description
Exports provider configuration, for a given provider, to an activity graph metadata definition file.

10.4.3.2 Syntax
exportAGProviderConfiguration(appName, directoryPath, definitionFileName, urn, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the activity graph application in which to perform this operation—always activitygraph-engines.</td>
</tr>
<tr>
<td>directoryPath</td>
<td>Destination directory for the XML file that will be generated. If you specify a directory that does not exist, then it will be created.</td>
</tr>
<tr>
<td>definitionFilePath</td>
<td>Name for the XML file that will be generated. If a file with the same name exists in the destination directory then it will be overwritten.</td>
</tr>
</tbody>
</table>
### 10.4.3.3 Example

The following example exports configuration information for the activity graph provider `oracle.webcenter.activitygraph.providers.analytics` to an XML file named `ag-provider-config.xml`, at the specified location:

```
<node name="wls:/weblogic/serverConfig">
  <subnode name="exportAGProviderConfiguration">
    <arg name="appName" value="activitygraph-engines"/>
    <arg name="directoryPath" value="/scratch/myAGmetadata"/>
    <arg name="definitionFileName" value="ag-provider-config.xml"/>
    <arg name="urn" value="oracle.webcenter.activitygraph.providers.analytics"/>
  </subnode>
</node>
```

### 10.4.4 deleteAllAGMetadata

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.4.4.1 Description

Deletes all the activity graph metadata that is defined for WebCenter Portal or a Portal Framework application. The delete operation is immediate and non-reversible.

You can use this command in conjunction with the WLST command `importAGMetadata` to completely re-install activity graph metadata.

**Note:** Any data in the relation store, similarity store, and rank store will be deleted the next time the activity graph engines run.

#### 10.4.4.2 Syntax

```
deleteAllAGMetadata(appName, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the activity graph application in which to perform this operation—always <code>activitygraph-engines</code>.</td>
</tr>
<tr>
<td><code>server</code></td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Utilities</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><code>applicationVersion</code></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the activity graph application is deployed.</td>
</tr>
</tbody>
</table>
10.4.3 Example
The following example deletes all existing activity graph metadata:

```
wls:/weblogic/serverConfig> deleteAllAGMetadata(appName='activitygraph-engines')
```

10.4.5 deleteAGAction
Module: Oracle WebCenter Portal
Use with WLST: Online

10.4.5.1 Description
Deletes the metadata for an action that is currently registered with activity graph. The
delete operation is immediate and non-reversible.

**Note:** Any data in the relation store that is associated with the action will be deleted the next time the activity graph engines run.

10.4.5.2 Syntax
```
deleteAGAction(appName, urn, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the activity graph application in which to perform this operation—always activitygraph-engines.</td>
</tr>
<tr>
<td>urn</td>
<td>URN for the activity graph action that you want to delete.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC.Utilities. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the activity graph application is deployed.</td>
</tr>
</tbody>
</table>

10.4.5.3 Example
The following example deletes activity graph metadata for the connect action:

```
wls:/weblogic/serverConfig> deleteAGAction(appName='activitygraph-engines', urn='connect')
```

10.4.6 deleteAGNodeClass
Module: Oracle WebCenter Portal
Use with WLST: Online

10.4.6.1 Description
Deletes the metadata for a node class that is currently registered with activity graph. The delete operation is immediate and non-reversible.

**Note:** Any data in the relation store that is associated with the node class will be deleted the next time the activity graph engines run.
10.4.6.2 Syntax

deleteAGNodeClass(appName, urn, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the activity graph application in which to perform this operation—always activitygraph-engines.</td>
</tr>
<tr>
<td>urn</td>
<td>URN for the activity graph node class that you want to delete.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Utility. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the activity graph application is deployed.</td>
</tr>
</tbody>
</table>

10.4.6.3 Example

The following example deletes activity graph metadata for the node class WC.wiki-page:

wls:/weblogic/serverConfig> deleteAGNodeClass(appName='activitygraph-engines', urn='WC.wiki-page')

10.4.7 deleteAGSimilarityCalculation

Module: Oracle WebCenter Portal

Use with WLST: Online

10.4.7.1 Description

Deletes the metadata for a similarity calculation that is currently registered with activity graph. The delete operation is immediate and non-reversible.

10.4.7.2 Syntax

deleteAGSimilarityCalculation(appName, urn, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the activity graph application in which to perform this operation—always activitygraph-engines.</td>
</tr>
<tr>
<td>urn</td>
<td>URN for the activity graph similarity calculation that you want to delete.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Utility. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the activity graph application is deployed.</td>
</tr>
</tbody>
</table>

10.4.7.3 Example

The following example deletes activity graph metadata for the similarity calculation item-edit:

wls:/weblogic/serverConfig>
**10.4.8 deleteAGRankCalculation**

Module: Oracle WebCenter Portal
Use with WLST: Online

10.4.8.1 Description
Deletes the metadata for a rank calculation that is currently registered with activity graph. The delete operation is immediate and non-reversible.

10.4.8.2 Syntax
```
deleteAGRankCalculation(appName, urn, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the activity graph application in which to perform this operation—always <code>activitygraph-engines</code>.</td>
</tr>
<tr>
<td><code>urn</code></td>
<td>URN for the activity graph rank calculation that you want to delete.</td>
</tr>
<tr>
<td><code>server</code></td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Utilities</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><code>applicationVersion</code></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the activity graph application is deployed.</td>
</tr>
</tbody>
</table>

10.4.8.3 Example
The following example deletes activity graph metadata for the activity-rank calculation:
```
wls:/weblogic/serverConfig>
deleteAGRankCalculation(appName='activitygraph-engines', urn='activity-rank')
```

**10.4.9 deleteAGProviderAssignment**

Module: Oracle WebCenter Portal
Use with WLST: Online

10.4.9.1 Description
Deletes the metadata for a provider assignment that is currently registered with activity graph, that is, a provider assignment defined by the unique triple combination `(action, sourceClass, trgClass)`. The delete operation is immediate and non-reversible.

10.4.9.2 Syntax
```
deleteAGProviderAssignment(appName, actionURN, srcClassURN, trgClassURN [server, applicationVersion])
```

```
deleteAGProviderAssignment(appName='activitygraph-engines', actionURN='activity-rank', srcClassURN='sourceClass', trgClassURN='trgClass')
```
10.4.9.3 Example

The following example deletes activity graph metadata for the provider assignment specified:

```
ws:/weblogic/serverConfig>
deleteAGRProviderAssignment(appName='activitygraph-engines', actionURN='connect',
srcClassURN='WC.user', trgClassURN='WC.user')
```

10.4.10 deleteAGQRPPRegistration

Module: Oracle WebCenter Portal

Use with WLST: Online

10.4.10.1 Description

Deletes the metadata for a QRPP (Query Result Post Processor) that is currently registered with activity graph.

The delete operation is immediate and non-reversible.

10.4.10.2 Syntax

```
deleteAGQRPPRegistration(appName, urn [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the activity graph application in which to perform this operation—always activitygraph-engines.</td>
</tr>
<tr>
<td>urn</td>
<td>URN for the QRPP that you want to delete.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Utilities. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the activity graph application is deployed.</td>
</tr>
</tbody>
</table>

10.4.10.3 Example

The following example deletes activity graph metadata for a QRPP named Event store metadata QRPP:

```
```
Activity Graph

wls:/weblogic/serverConfig>
deleteAGQRPPRegistration(appName='activitygraph-engines', urn='Event store metadata QRPP')

10.4.11 deleteAGProviderConfiguration
Module: Oracle WebCenter Portal
Use with WLST: Online

10.4.11.1 Description
Deletes the metadata for a provider configuration. The delete operation is immediate and non-reversible.

10.4.11.2 Syntax
deleteAGProviderConfiguration(appName, urn [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the activity graph application in which to perform this operation—always activitygraph-engines.</td>
</tr>
<tr>
<td>urn</td>
<td>URN for the activity graph provider that you want to delete.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Utilities.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the activity graph application is deployed.</td>
</tr>
</tbody>
</table>

10.4.11.3 Example
The following example deletes configuration information for the activity graph provider oracle.webcenter.activitygraph.providers.analytics:
wls:/weblogic/serverConfig>
deleteAGProviderConfiguration(appName='activitygraph-engines', urn='oracle.webcenter.activitygraph.providers.analytics')

10.4.12 renameAGAction
Module: Oracle WebCenter Portal
Use with WLST: Online

10.4.12.1 Description
Changes the URN of an action that is currently registered with activity graph. Any data in the relation store that is associated with the action is preserved.

**Note:** This command does not delete the action and create an action with a different name as this causes data associated with the original action to be deleted.

10.4.12.2 Syntax
renameAGAction(appName, currentURN, newURN [server, applicationVersion])
10.4.12.3 Example
The following example changes the `connect` action URN to `people-connect`:

```
(wls:/weblogic/serverConfig> renameAGAction(appName='activitygraph-engines', currentURN='connect', newURN='people-connect'))
```

10.4.13 renameAGNodeClass
Module: Oracle WebCenter Portal
Use with WLST: Online

10.4.13.1 Description
Changes the URN of a node class that is currently registered with activity graph. Any data in the relation store that is associated with the node class is preserved.

**Note:** This command does not delete the node class and create a node class with a different name as this would cause data associated with the original node class to be deleted.

10.4.13.2 Syntax

```
renameAGNodeClass(appName, currentURN, newURN, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the activity graph application in which to perform this operation—always <code>activitygraph-engines</code>.</td>
</tr>
<tr>
<td><code>currentURN</code></td>
<td>Current node class URN.</td>
</tr>
<tr>
<td><code>newURN</code></td>
<td>New node class URN.</td>
</tr>
<tr>
<td><code>server</code></td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Utilities</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><code>applicationVersion</code></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the activity graph application is deployed.</td>
</tr>
</tbody>
</table>

10.4.13.3 Example
The following example changes the `WC.user` node class URN to `WC.people`:
10.4.14 setAGProperty

Module: Oracle WebCenter Portal

Use with WLST: Online

10.4.14.1 Description

Sets a system property for activity graph. This command sets a value based on the property's datatype (String, Integer, Float, Boolean).

Activity graph system properties include settings for:

- Oracle Secure Enterprise Search (SES) Admin API web service connection
  (oracle.webcenter.activitygraph.providers.datasources.ses.soap.admin.url and
  oracle.webcenter.activitygraph.providers.datasources.ses.soap.query.url)
- Rank engine configuration
  (oracle.webcenter.activitygraph.rankengine.enabled)

See also, "Managing Activity Graph" in the Oracle Fusion Middleware Administering Oracle WebCenter Portal for a list of system properties and their datatypes.

Configuration changes made using the setAGProperty WLST command are only effective after your restart the managed server on which the activity graph application is deployed (WC_Utilities).

10.4.14.2 Syntax

```
setAGProperty(appName, propertyName, propertyValue, propertyType, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the activity graph application in which to perform this operation—always activitygraph-engines.</td>
</tr>
<tr>
<td>propertyName</td>
<td>Name of the activity graph property.</td>
</tr>
<tr>
<td>propertyValue</td>
<td>Value for the activity graph property.</td>
</tr>
<tr>
<td>propertyType</td>
<td>Datatype of the property. Valid values are: String, Int, Float or Boolean. Values are case sensitive.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Utilities. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the activity graph application is deployed.</td>
</tr>
</tbody>
</table>

10.4.14.3 Example

The following example enables the Rank Engine:

```
wls:/weblogic/serverConfig> setAGProperty(appName='activitygraph-engines', propertyName='oracle.webcenter.activitygraph.rankengine.enabled', newURN='WC.people')
```
**getAGProperty**

Module: Oracle WebCenter Portal

Use with WLST: Online

### 10.4.15.1 Description

Returns the current setting for a given activity graph property.

See also, "Managing Activity Graph" in the Oracle Fusion Middleware Administering Oracle WebCenter Portal for a list of valid system properties.

### 10.4.15.2 Syntax

```
getAGProperty(appName, propertyName, propertyType [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the activity graph application in which to perform this operation—always activitygraph-engines.</td>
</tr>
<tr>
<td>propertyName</td>
<td>Name of the activity graph property.</td>
</tr>
<tr>
<td>propertyType</td>
<td>Datatype of the property. Valid values are: String, Int, Float or Boolean. Values are case sensitive.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Utilities. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the activity graph application is deployed.</td>
</tr>
</tbody>
</table>

### 10.4.15.3 Example

The following example returns the current value of the system property oracle.webcenter.activitygraph.providers.datasources.ses.soap.admin.url:

```
wls:/weblogic/serverConfig> getAGProperty(appName='activitygraph-engines', propertyName='oracle.webcenter.activitygraph.providers.datasources.ses.soap.admin.url', propertyType='String')
```

**setAGPasswordCredential**

Module: Oracle WebCenter Portal

Use with WLST: Online

### 10.4.16.1 Description

Sets credentials (user name and password) for an activity graph credential property.

See also, "Managing Activity Graph" in the Oracle Fusion Middleware Administering Oracle WebCenter Portal for a list of properties with the PasswordCredential datatype, for example, oracle.webcenter.activitygraph.providers.datasources.ses.soap.admin.credential.
### 10.4.16.2 Syntax

```
setAGPasswordCredentialProperty(appName, propertyName, userName, password,[server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
</table>
| appName           | Name of the activity graph application in which to perform this operation—always `activitygraph-engines`.
| propertyName      | Name of the activity graph property that specifies credentials (and has `PasswordCredential` datatype).
| userName          | User name associated with the credential property.                         |
| password          | Password associated with the user name specified.                          |
| server            | Optional. Name of the managed server where the application is deployed. For example, `WC_Utilities`. Required when applications with the same name are deployed to different servers and also when you have a cluster. |
| applicationVersion| Optional. Version number of the deployed application. Required if more than one version of the activity graph application is deployed. |

### 10.4.16.3 Example

The following example sets user name and password credentials for the Oracle SES Admin tool:

```
wls:/weblogic/serverConfig> setAGProperty(appName='activitygraph-engines', propertyName='oracle.webcenter.activitygraph.providers.datasources.ses.soap.admin.credential', userName='eqsys', password='mypassword')
```

### 10.5 Activity Stream

Use the commands listed in Table 10–6 to archive and restore activity stream data generated for WebCenter Portal and Portal Framework applications.

Configuration changes made using these WLST commands are only effective after your restart the Managed Server on which the application is deployed. For details, see Oracle Fusion Middleware Administering Oracle WebCenter Portal.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>archiveASByDate</td>
<td>Archive activity stream data that is older than a specified date.</td>
<td>Online</td>
</tr>
<tr>
<td>archiveASByDeletedObjects</td>
<td>Archive activity stream data associated with deleted objects.</td>
<td>Online</td>
</tr>
<tr>
<td>archiveASByClosedSpaces</td>
<td>Archive activity stream data associated with portals that are currently closed.</td>
<td>Online</td>
</tr>
<tr>
<td>archiveASByInactiveSpaces</td>
<td>Archive activity stream data associated with portals that have been inactive since a specified date.</td>
<td>Online</td>
</tr>
<tr>
<td>restoreASByDate</td>
<td>Restore archived activity stream data from a specified date into production tables.</td>
<td>Online</td>
</tr>
<tr>
<td>truncateASArchive</td>
<td>Truncates activity stream archive data.</td>
<td>Online</td>
</tr>
<tr>
<td>archiveASBySpace</td>
<td>Archive activity stream data associated with a portal.</td>
<td>Online</td>
</tr>
</tbody>
</table>
10.5.1 archiveASByDate

Module: Oracle WebCenter Portal

Use with WLST: Online

10.5.1.1 Description
Archives activity stream data that is older than a specified date.

This command moves data from production tables to archive tables. Exceptions include WC_ACTOR_DETAIL and WC_OBJECT_DETAIL—data in these tables is copied to archive tables rather than moved.

Rows in WC_OBJECT_DETAIL that are not used by any activity element are deleted.

10.5.1.2 Syntax
archiveASByDate(appName, year, month, day, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>year</td>
<td>Year before which to archive activity stream data. For example, 2009.</td>
</tr>
<tr>
<td>month</td>
<td>Month before which to archive activity stream data. For example, enter 1 for January, 2 for February, and so on.</td>
</tr>
<tr>
<td>day</td>
<td>Day of the month before which to archive activity stream data.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.5.1.3 Example
The following example archives activity stream data that is older than October 1, 2009 for WebCenter Portal (webcenter):

```wls:/weblogic/serverConfig> archiveASByDate(appName='webcenter', year=2009, month=10, day=1)```
10.5.2 archiveASByDeletedObjects
Module: Oracle WebCenter Portal
Use with WLST: Online

10.5.2.1 Description
Archives activity stream data associated with deleted objects.
This command moves data from production tables to archive tables, except for $WC_{ACTOR\\_DETAIL}$ as data in this table is copied to the archive table rather than moved.
Rows in $WC_{OBJECT\\_DETAIL}$ that satisfy the criteria (in this case, deleted objects) are deleted.

10.5.2.2 Syntax
archiveASByDeletedObjects(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always \textit{webcenter}.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, $WC_{Spaces}$.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.5.2.3 Example
The following example archives activity stream data associated with deleted objects from WebCenter Portal (\textit{webcenter}):

```
wls:/weblogic/serverConfig> archiveASByDeletedObjects(appName='webcenter')
```

10.5.3 archiveASByClosedSpaces
Module: Oracle WebCenter Portal
Use with WLST: Online

10.5.3.1 Description
Archives activity stream data associated with portals that are currently closed.
This command moves data from production tables to archive tables, except for $WC_{ACTOR\\_DETAIL}$ as data in this table is copied to the archive table rather than moved.
Rows in $WC_{OBJECT\\_DETAIL}$ that satisfy the criteria (in this case, objects involved in activities of portals that are closed) are deleted.

10.5.3.2 Syntax
archiveASByClosedSpaces(appName, [server, applicationVersion])
Example

The following example archives activity stream data associated with portals that are currently marked as closed in WebCenter Portal (webcenter):

\texttt{wls:/weblogic/serverConfig> archiveASByClosedSpaces(appName='webcenter')}

### 10.5.4 \texttt{archiveASByInactiveSpaces}

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.5.4.1 Description

Archives activity stream data associated with portals that have been inactive since a specified date. An inactive portals an open or closed portal in which there has been no activity since the specified date.

This command moves data from production tables to archive tables, except for \texttt{WC\_ACTOR\_DETAIL}—data in this table is copied to the archive table rather than moved.

Rows in \texttt{WC\_OBJECT\_DETAIL} that satisfy the criteria (in this case, objects involved in activities of portals that have been inactive since the specified date) are deleted.

#### 10.5.4.2 Syntax

\texttt{archiveASByInactiveSpaces(appName, year, month, day, [server, applicationVersion])}

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{appName}</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>\textit{server}</td>
<td>Optional. Name of the managed server where WebCenter Portal is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>\textit{applicationVersion}</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.5.4.3 Example
The following example archives activity stream data associated with portals that have been inactive (no activities have occurred, regardless of open or closed status) since October 1, 2009:

```plaintext
wlst:/weblogic/serverConfig> archiveASByInactiveSpaces(appName='webcenter', year=2009, month=10, day=1)
```

10.5.5 restoreASByDate
Module: Oracle WebCenter Portal
Use with WLST: Online

10.5.5.1 Description
Restores archived activity stream data from a specified date into production tables.

This command moves data from archive tables to production tables, except for `WC_ACTOR_DETAIL` as data in this table is not restored because data is not deleted from this table during the archive process.

Rows that already exist in the production tables are not changed during the restore process.

10.5.5.2 Syntax
```
restoreASByDate(appName, year, month, day, [server, applicationVersion])
```

### Argument | Definition
--- | ---
`appname` | Name of the application in which to perform this operation. For WebCenter Portal, the application name is always `webcenter`. For Portal Framework applications, specify the appropriate name.

`year` | Year from which to restore activity stream data. For example, 2009.

`month` | Month from which to restore activity stream data. For example, enter 1 for January, 2 for February, and so on.

`day` | Day of the month from which to restore activity stream data.

`server` | Optional. Name of the managed server where the application is deployed. For example, `WC_Spaces`. Required when applications with the same name are deployed to different servers and also when you have a cluster.

`applicationVersion` | Optional. Version number of the deployed application. Required if more than one version of the application is deployed.

10.5.5.3 Example
The following example restores activity stream data archived since October 1, 2009:

```plaintext
wlst:/weblogic/serverConfig> restoreASByDate(appName='webcenter', year=2009, month=10, day=1)
```

10.5.6 truncateASArchive
Module: Oracle WebCenter Portal
Use with WLST: Online

**10.5.6.1 Description**
Truncates activity stream archive data.

**10.5.6.2 Syntax**

```plaintext
truncateASArchive(appName, [server, applicationVersion])
```

**Argument** | **Definition**
---|---
`appName` | Name of the application in which to perform this operation. For WebCenter Portal, the application name is always `webcenter`. For Portal Framework applications, specify the appropriate name.
`server` | Optional. Name of the managed server where the application is deployed. For example, `WC_Spaces`. Required when applications with the same name are deployed to different servers and also when you have a cluster.
`applicationVersion` | Optional. Version number of the deployed application. Required if more than one version of the application is deployed.

**10.5.6.3 Example**
The following example truncates activity stream archive data:

```plaintext
wls:/weblogic/serverConfig>truncateASArchive(appName='webcenter')
```

**10.5.7 archiveASBySpace**
Module: Oracle WebCenter Portal
Use with WLST: Online

**10.5.7.1 Description**
Archives activity stream data associated with a named portal, only keeping a fixed number of activities.
This command moves data from production tables to archive tables.

**10.5.7.2 Syntax**

```plaintext
archiveASBySpace(appName, space, cnt, [server, applicationVersion])
```

**Argument** | **Definition**
---|---
`appName` | Name of the application in which to perform this operation. For WebCenter Portal, the application name is always `webcenter`.
`space` | Name of the portal whose data you want to archive. For example, `MySalesPortal`.
`cnt` | Number of portal activities you want to keep in the production table. For example, `2000`.
`server` | Optional. Name of the managed server where the application is deployed. For example, `WC_Spaces`. Required when applications with the same name are deployed to different servers and also when you have a cluster.
### 10.5.7.3 Example

The following example keeps the last 2000 activities associated with the portal MySalesPortal and archives the rest:

```bash
wls:/weblogic/serverConfig> archiveASBySpace(appName='webcenter', space='MySalesPortal', cnt=2000)
```

### 10.5.8 archiveASAllSpaces

#### Module: Oracle WebCenter Portal

#### Use with WLST: Online

#### 10.5.8.1 Description

Archives activity stream data for all portals, only keeping a fixed number of activities. This command moves data from production tables to archive tables.

#### 10.5.8.2 Syntax

```
archiveASAllSpaces(appName, cnt, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>cnt</td>
<td>Number of portal activities you want to keep in the production table.</td>
</tr>
<tr>
<td></td>
<td>For example, 2000.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different</td>
</tr>
<tr>
<td></td>
<td>servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more</td>
</tr>
<tr>
<td></td>
<td>than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

#### 10.5.8.3 Example

The following example keeps the last 2000 activities for each portal in production tables and archives the remaining activity stream data:

```bash
wls:/weblogic/serverConfig> archiveASAllSpaces(appName='webcenter', cnt=2000)
```

### 10.5.9 archiveASByUser

#### Module: Oracle WebCenter Portal

#### Use with WLST: Online

#### 10.5.9.1 Description

Archives activity stream data associated with a single user, only keeping a fixed number of activities.
This command moves data from production tables to archive tables.

### 10.5.9.2 Syntax

archiveASByUser(appName, actor, cnt, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>actor</td>
<td>Name of the user whose data you want to archive. For example, Monty.</td>
</tr>
<tr>
<td>cnt</td>
<td>Number of user activities you want to keep in the production table. For example, 2000.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

### 10.5.9.3 Example

The following example keeps the last 2000 activities associated with the user Monty and archives the rest:

```
<wls:/weblogic/serverConfig> archiveASByUser(appName='webcenter', actor='Monty', cnt=2000)
```

### 10.5.10 archiveASAllUsers

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.5.10.1 Description

Archives activity stream data for all users, only keeping a fixed number of activities. This command moves data from production tables to archive tables.

#### 10.5.10.2 Syntax

archiveASAllUsers(appName, cnt, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>cnt</td>
<td>Number of user activities you want to keep in the production table. For example, 2000.</td>
</tr>
</tbody>
</table>
Example

The following example keeps the last 2000 activities for user in production tables and archives the remaining activity stream data:

```
wlst:/weblogic/serverConfig> archiveASAllUsers(appName='webcenter', cnt=2000)
```

10.5.11 archiveASByDeletedActors

Module: Oracle WebCenter Portal

Use with WLST: Online

10.5.11.1 Description

Archives activity stream data associated with deleted users (actors).

This command moves data from production tables to archive tables. Rows in `WC_AS_ACTOR_DETAIL` that satisfy the criteria (in this case, deleted actors) are deleted.

10.5.11.2 Syntax

```
archiveASByDeletedActors(appName, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.5.11.3 Example

The following example archives activity stream data associated with deleted users for WebCenter Portal (`webcenter`):

```
wlst:/weblogic/serverConfig> archiveASByDeletedActors(appName='webcenter')
```

10.5.12 showASStatistics

Module: Oracle WebCenter Portal

Use with WLST: Online
10.5.12.1 Description
Reports various activity stream statistics:
- Number of activities for top "N" portals, ordered by activity count
- Number of activities for top "N" users, ordered by activity count
- Number of activities after a specific date
- Number of activities after a specific date for top "N" portals, ordered by activity count
- Number of activities after a specified date for top "N" users, ordered by activity count

10.5.12.2 Syntax
showASStatistics(appName, year, month, day, cnt, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>year</td>
<td>Year from which to report activity stream statistics. For example, 2009.</td>
</tr>
<tr>
<td>month</td>
<td>Month from which to report activity stream statistics. For example, enter 1 for January, 2 for February, and so on.</td>
</tr>
<tr>
<td>day</td>
<td>Day of the month from which to report activity stream statistics.</td>
</tr>
<tr>
<td>cnt</td>
<td>Number of portals or users included in the report. For example, 50.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.5.12.3 Example
The following example reports activity stream statistics for the top 50 portals and top 50 users in WebCenter Portal (webcenter) since 6/12/2012:

wls:/weblogic/serverConfig> showASStatistics(appName='webcenter', year=2012, month=7, day=18, cnt=3)

============================================================
Cutoff Date = 7/18/12 12:23 PM
Top Count = 3

1. Portals with most activity count are:
   Portal Finance Count =10
   Portal Photography Count =9
   Portal Sport Count =7

2. Users with most activity count are:
   User Monty Count =30
   User Pat Count =20
User Dave Count = 10

3. Total Activities after 7/18/12 12:23 PM = 80

4. Portals with most activity count after specific date, are:
   Portal Photography Count = 9
   Portal Finance Count = 6
   Portal Sport Count = 6

5. Users with most activity count after specific date, are:
   User Monty Count = 10
   User Dave Count = 8
   User Monica Count = 7

10.6 Content Repository

Use the commands listed in Table 10–7 to manage content repository connections and configure document services for WebCenter Portal and Portal Framework applications.

Configuration changes made using these WLST commands are only effective after your restart the Managed Server on which the application is deployed. For details, see Oracle Fusion Middleware Administering Oracle WebCenter Portal.

<table>
<thead>
<tr>
<th>Table 10–7 Content Repository WLST Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use this command...</strong></td>
</tr>
<tr>
<td>createJCRContentServerConnection</td>
</tr>
<tr>
<td>setJCRContentServerConnection</td>
</tr>
<tr>
<td>listJCRContentServerConnections</td>
</tr>
<tr>
<td>createJCRPortalConnection</td>
</tr>
<tr>
<td>setJCRPortalConnection</td>
</tr>
<tr>
<td>listJCRPortalConnections</td>
</tr>
<tr>
<td>createJCRFileSystemConnection</td>
</tr>
<tr>
<td>setJCRFileSystemConnection</td>
</tr>
<tr>
<td>listJCRFileSystemConnections</td>
</tr>
<tr>
<td>createJCRSharePointConnection</td>
</tr>
<tr>
<td>setJCRSharePointConnection</td>
</tr>
<tr>
<td>listJCRSharePointConnections</td>
</tr>
</tbody>
</table>
### Table 10–7 (Cont.) Content Repository WLST Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>listDocumentsSpacesProperties</td>
<td>List properties for the back-end Content Server that is being used by WebCenter Portal.</td>
<td>Online</td>
</tr>
<tr>
<td>setDocumentsSpacesProperties</td>
<td>Modify properties for the back-end Content Server used by WebCenter Portal.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteDocumentsSpacesProperties</td>
<td>Delete properties for the back-end Content Server used by WebCenter Portal.</td>
<td>Online</td>
</tr>
<tr>
<td>exportFoldersGData</td>
<td>Export Folders_g data to a specified location.</td>
<td>Online</td>
</tr>
<tr>
<td>migrateFoldersGDataToFrameworkFolders</td>
<td>Migrate Folders_g data to FrameworkFolders.</td>
<td>Online</td>
</tr>
</tbody>
</table>

### 10.6.1 createJCRContentServerConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.6.1.1 Description

Creates a connection to an Oracle WebCenter Content repository for a named application.

#### 10.6.1.2 Syntax

```java
createJCRContentServerConnection(appName, name, socketType, [url, serverHost, serverPort, keystoreLocation, keystorePassword, privateKeyAlias, privateKeyPassword, webContextRoot, clientSecurityPolicy, cacheInvalidationInterval, binaryCacheMaxEntrySize, adminUsername, adminPassword, extAppId, timeout, isPrimary, server, applicationVersion])
```

#### Argument | Definition
--- | ---
`appName` | Name of the application in which to perform this operation. For WebCenter Portal, the application name is always `webcenter`. For Portal Framework applications, specify the appropriate name.

`name` | Connection name. The name must be unique (across all connection types) within the application.
<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>socketType</td>
<td>Specifies whether Oracle WebCenter Content's Content Server connects on the content server listener port or the web server filter, and whether the listener port is SSL enabled. Valid values are socket, web, socketssl, and jaxws. This option has no default. Choose from:</td>
</tr>
<tr>
<td></td>
<td>■ socket—Use an intradoc socket connection to connect to the Content Server. The client IP address must be added to the list of authorized addresses in the Content Server. In this case, the client is the machine on which Oracle WebCenter Portal is running.</td>
</tr>
<tr>
<td></td>
<td>■ socketssl—Use an intradoc socket connection to connect to the Content Server that is secured using the SSL protocol. The client’s certificates must be imported in the server's trust store for the connection to be allowed. This is the most secure option, and the recommended option whenever identity propagation is required (for example, in WebCenter Portal).</td>
</tr>
<tr>
<td></td>
<td>■ web—Use an HTTP(S) connection to connect to the Content Server. Note that for WebCenter Portal, this option is not suitable for the active connection, that is, the back-end Content Server repository that is being used to store portal-specific documents and Home portal documents, because it does not allow identity propagation.</td>
</tr>
<tr>
<td></td>
<td>■ jaxws—Use a Java API for XML web services connection to connect to the Content Server.</td>
</tr>
<tr>
<td>url</td>
<td>Optional. Content Server URL. Required only if socketType is set to web or jaxws. URL should be in the format: <a href="http://hostname:port/web_root/plugin_root">http://hostname:port/web_root/plugin_root</a>. For example: <a href="http://mycontentserver/cms/idcplg">http://mycontentserver/cms/idcplg</a></td>
</tr>
<tr>
<td>serverHost</td>
<td>Optional. Host name of the machine where the Content Server is running. Required if socketType is set to socket or socketssl.</td>
</tr>
<tr>
<td>serverPort</td>
<td>Optional. Port on which the Content Server listens. Required if socketType is set to socket or socketssl:</td>
</tr>
<tr>
<td></td>
<td>■ Socket—Port specified for the incoming provider in the server.</td>
</tr>
<tr>
<td></td>
<td>■ Socket SSL—Port specified for the sslIncoming provider in the server.</td>
</tr>
<tr>
<td></td>
<td>This property corresponds to the IntradocServerPort setting in the Content Server configuration file, which defaults to port 4444.</td>
</tr>
<tr>
<td>keyStoreLocation</td>
<td>Optional. Location of key store that contains the private key used to sign the security assertions. Required only if socketType is set to socketssl. The key store location must be an absolute path.</td>
</tr>
<tr>
<td>keyStorePassword</td>
<td>Optional. Password required to access the key store. Required only if socketType is set to socketssl.</td>
</tr>
<tr>
<td>privateKeyAlias</td>
<td>Optional. Client private key alias in the key store. The key is used to sign messages to the server. The public key corresponding to this private key must be imported in the server keystore. Required only if socketType is set to socketssl. The value for this argument must be a string that contains neither special characters nor white space.</td>
</tr>
</tbody>
</table>
privateKeyPassword

Optional. Password to be used with the private key alias in the key store. Required only if socketType is set to socketssl.

webContextRoot

Optional. Web server context root for the Content Server. Use the format /<context_root>. For example, /cs.

When specified, several Oracle WebCenter Content features based on iFrame are available in the application. This includes:

- **Associating a content profile with files when uploading new or updated files to Content Server.**
  
  For more information, see "Uploading New Files" and "Uploading a New Version of an Existing File" in Oracle Fusion Middleware Using Oracle WebCenter Portal.

- **Using the document review functionality available in Oracle AutoVue.**
  
  For more information, see “Collaborating on Documents Using Oracle AutoVue” in Oracle Fusion Middleware Using Oracle WebCenter Portal.

- **Editing advanced document properties.**
  
  For more information, see "Working with File Properties" in Oracle Fusion Middleware Using Oracle WebCenter Portal.

- **Viewing folder and file workflow details.**
  
  For more information, see "Working with Workflow-Enabled Content" in Oracle Fusion Middleware Using Oracle WebCenter Portal.

- **Previewing files in a slide viewer.**
  
  For more information, see "Opening a File" in Oracle Fusion Middleware Using Oracle WebCenter Portal.

- **Site Studio integration**
  
  For more information, see Oracle Fusion Middleware Using Oracle WebCenter Portal.

webContextRoot is only applicable when IDENTITY_PROPAGATION is used for authentication, that is, when extAppId is set to an empty string.

**Note:** To fully enable these Oracle WebCenter Content features you must access WebCenter Portal (or your Portal Framework application) through Oracle HTTPS Server (OHS) to expose Content Server and the application under the same host and port. Both the application and Content Server must also use single sign on. For information about setting up OHS to front-end WebCenter Portal and Portal Framework applications, see "Content Server - Configuration" in Oracle Fusion Middleware Administering Oracle WebCenter Portal.

If your application is connected to multiple Content Servers, Oracle recommends that each Content Server has a unique Web Server Context Root so that OHS re-direction works correctly.

clientSecurityPolicy

Optional. Client security policy to be used when the socketType is jaxws. For example: oracle/wss11_saml_token_with_message_protection_service_policy.

Leave the field blank if your environment supports Global Policy Attachments (GPA).

cacheInvalidationInterval

Optional. Time between checks for external Content Server content changes (in minutes). WebCenter Portal automatically clears items that have changed from the cache. The minimum interval is 2 minutes. By default, cache invalidation is disabled (specified as 0) which means that no periodic checks are made for content changes.
binaryCacheMaxEntrySize

Optional. Maximum cacheable size (in bytes) for Content Server binary documents. Documents larger than this size are not cached by WebCenter Portal or Portal Framework applications.

Default is 102400 bytes (100K).

Tune this value based on your machine’s memory configuration and the types of binary documents that you expect to cache. Be aware that, unless Coherence is enabled, there is no maximum total size for the cache.

If you are using Coherence you can additionally specify the total amount of memory to be used for binary caches. For this reason, using Coherence for any type of production environment is strongly recommended, and is a requirement for High Availability (HA) environments. For more information, see the ‘Modifying Cache Settings for Content Presenter’ section in Oracle Fusion Middleware Administering Oracle WebCenter Portal.

Most documents stored in Content Server are considered binary content, that is, images, plain text, Word documents, and so on. The only exception is Site Studio content which is stored in CDF data files and cached separately in a Virtual Content Repository (VCR) cache (or node cache).

adminUsername

Optional. User name with administrative rights for this Content Server instance. This user will be used to fetch content type information based on profiles and track document changes for cache invalidation purpose.

Defaults to sysadmin.

adminPassword

Optional. Password for the Content Server administrator specified in adminUsername. Required when socketType is set to web.

extAppId

Optional. External application used to authenticate users against the Content Server. This value should match the name of an existing external application connection. See also listExtAppConnections. If extAppId is not set, no change is made to the authentication method or external application ID.

If extAppId is set to an empty string, the authentication method used is IDENTITY_PROPAGATION. With this method, the application and Content Server use the same identity store to authenticate users. Note that extAppID is mandatory when socketType is set to web.

timeout

Length of time allowed to log in to the Content Server (in ms) before issuing a connection timeout message, and the RIDC socket timeout used for all service requests for connection types web, socket and socketssl.

If the timeout property is not set, the following values are used:

- **Login timeout** - the default concurrency timeout for the oracle.webcenter.content resource is used (30s or 30000ms). Refer to “Configuring Concurrency Management” in Oracle Fusion Middleware Performance and Tuning Guide for more information.

- **RIDC socket timeout** - the default RIDC socket timeout (60s or 60000ms) is used for all service requests for connection types socket, socketssl, or web.

If the timeout property is set and the connection type is socket, socketssl, or web, Oracle recommends that you do not specify a value less than 60000ms as this would reduce the RIDC socket timeout and increase the likelihood that long running requests will time out. For example, timeouts may occur during long running searches, long file uploads, or long copy operations.
Example:
The following example creates a socket-based connection to an Oracle WebCenter Content repository running on myhost.com at port 4444. For authentication purposes, an existing external application named myExtApp is used. See also, `createExtAppConnection`.

```wls:
createJCRContentServerConnection(appName='webcenter', name='myContentServerConnection', socketType='socket', serverHost='myhost.com', serverPort=4444, extAppId='myExtApp', isPrimary=1)
```

The following example creates an SSL socket-based connection to an Oracle WebCenter Content repository.

```wls:
createJCRContentServerConnection(appName='webcenter', name='myContentServerConnection', socketType='socketssl', serverHost='myhost.com', serverPort=4444, keystoreLocation='d:/keys/here', keystorePassword='AlphaSquad7', privateKeyAlias='enigma', privateKeyPassword='S0larPl3x1s', extAppId='myExtApp')
```

The following example creates a JAX-WS (Java API for XML web services) connection to an Oracle WebCenter Content repository:

```wls:
createJCRContentServerConnection(appName='webcenter', name='myContentServerConnection', socketType='jaxws', url='http://myhost.com:9044/idcnativews', clientSecurityPolicy='oracle/wss10_saml_token_client_policy')
```

### 10.6.2 setJCRContentServerConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.6.2.1 Description

Edits an existing Oracle WebCenter Content repository connection. This command requires that you specify values for `appName` and `name`, plus one additional argument.
10.6.2.2 Syntax

```java
setJCRContentServerConnection(appName, name, [socketType, url, serverHost, serverPort, keystoreLocation, keystorePassword, privateKeyAlias, privateKeyPassword, webContextRoot, clientSecurityPolicy, cacheInvalidationInterval, binaryCacheMaxEntrySize, adminUsername, adminPassword, extAppId, timeout, isPrimary, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td><code>name</code></td>
<td>Name of an existing Oracle WebCenter Content repository connection.</td>
</tr>
<tr>
<td><code>socketType</code></td>
<td>Optional. Specifies whether the Oracle WebCenter Content's Content Server connects on the content server listener port or the web server filter, and whether the listener port is SSL enabled. Valid values are <code>socket</code>, <code>web</code>, and <code>socketssl</code>. This option has no default. Choose from:</td>
</tr>
<tr>
<td></td>
<td>■ socket—Use an intradoc socket connection to connect to the Content Server. The client IP address must be added to the list of authorized addresses in the Content Server. In this case, the client is the machine on which Oracle WebCenter Portal is running.</td>
</tr>
<tr>
<td></td>
<td>■ socketssl—Use an intradoc socket connection to connect to the Content Server that is secured using the SSL protocol. The client's certificates must be imported in the server's trust store for the connection to be allowed. This is the most secure option, and the recommended option whenever identity propagation is required (for example, in WebCenter Portal).</td>
</tr>
<tr>
<td></td>
<td>■ web—Use an HTTP(S) connection to connect to the Content Server. Note that for WebCenter Portal, this option is not suitable for the active connection, that is, the back-end Content Server repository that is being used to store portal-specific documents and Home portal documents, because it does not allow identity propagation.</td>
</tr>
<tr>
<td></td>
<td>■ jaxws—Use a Java API for XML web services connection to connect to the Content Server.</td>
</tr>
<tr>
<td><code>url</code></td>
<td>Optional. Content Server URL. Required only if <code>socketType</code> is set to <code>web</code> or <code>jaxws</code>. URL should be in the format: <code>http://hostname:port/web_root/plugin_root</code></td>
</tr>
<tr>
<td><code>serverHost</code></td>
<td>Optional. Host name of the machine where the Content Server is running. Required if <code>socketType</code> is set to <code>socket</code> or <code>socketssl</code>.</td>
</tr>
<tr>
<td><code>serverPort</code></td>
<td>Optional. Port on which the Content Server listens. Required if <code>socketType</code> is set to <code>socket</code> or <code>socketssl</code>:</td>
</tr>
<tr>
<td></td>
<td>■ Socket—Port specified for the incoming provider in the server.</td>
</tr>
<tr>
<td></td>
<td>■ Socket SSL—Port specified for the sslincoming provider in the server.</td>
</tr>
</tbody>
</table>

For example, 4444
<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>keystoreLocation</td>
<td>Optional. Location of key store that contains the private key used to sign the security assertions. Required only if socketType is set to socketssl. The key store location must be an absolute path.</td>
</tr>
<tr>
<td>keystorePassword</td>
<td>Optional. Password required to access the key store. Required only if socketType is set to socketssl.</td>
</tr>
<tr>
<td>privateKeyAlias</td>
<td>Optional. Client private key alias in the key store. Required only if socketType is set to socketssl. The value for this argument must be a string that contains neither special characters nor white space.</td>
</tr>
<tr>
<td>privateKeyPassword</td>
<td>Optional. Password to be used with the private key alias in the key store. Required only if socketType is set to socketssl.</td>
</tr>
<tr>
<td>webContextRoot</td>
<td>Optional. Web server context root for the Content Server. Use the format /&lt;context_root&gt;. For example, /cs. When specified, several Oracle WebCenter Content features based on iFrame, such as previewing files in a slide viewer, are available in WebCenter Portal or the Portal Framework application. <strong>Note:</strong> To fully enable these features you must access WebCenter Portal or the Portal Framework application through Oracle HTTPS Server (OHS) to expose Content Server and the application under the same host and port. In addition, both the application and the Content Server must use single sign on. For information about setting up OHS to front-end WebCenter Portal and Portal Framework applications, see &quot;Content Server - Configuration&quot; in Oracle Fusion Middleware Administering Oracle WebCenter Portal. webContextRoot is only applicable when IDENTITY_PROPAGATION is used for authentication, that is, when extAppId is set to an empty string.</td>
</tr>
<tr>
<td>clientSecurityPolicy</td>
<td>Optional. Client security policy to be used when the socketType is jaxws. For example: oracle/wss11_saml_token_with_message_protection_service_policy Leave the field blank if your environment supports Global Policy Attachments (GPA).</td>
</tr>
<tr>
<td>cacheInvalidationInterval</td>
<td>Optional. Time between checks for external Content Server content changes (in minutes). WebCenter Portal automatically clears items that have changed from the cache. The minimum interval is 2 minutes. By default, cache invalidation is disabled (specified as 0) which means that no periodic checks are made for content changes.</td>
</tr>
</tbody>
</table>
**binaryCacheMaxEntrySize**  
Optional. Maximum cacheable size (in bytes) for Content Server binary documents. Documents larger than this size are not cached by WebCenter Portal or the Portal Framework application. Defaults is 102400 bytes (100K).

Tune this value based on your machine’s memory configuration and the types of binary documents that you expect to cache. Be aware that, unless Coherence is enabled, there is no maximum total size for the cache.

If you are using Coherence you can additionally specify the total amount of memory to be used for binary caches. For this reason, using Coherence for any type of production environment is strongly recommended, and is a requirement for High Availability (HA) environments. For more information, see the “Modifying Cache Settings for Content Presenter” section in Oracle Fusion Middleware Administering Oracle WebCenter Portal.

Most documents stored in Content Server are considered binary content, that is, images, plain text, Word documents, and so on. The only exception is Site Studio content which is stored in CDF data files and cached separately in a Virtual Content Repository (VCR) cache (or node cache).

**adminUsername**  
Optional. User name with administrative rights for this Content Server instance. This user will be used to fetch content type information based on profiles and track document changes for cache invalidation purpose. Defaults to sysadmin.

**adminPassword**  
Optional. Password for the Content Server administrator specified in adminUsername. Required when socketType is set to web.

**extAppId**  
Optional. External application used to authenticate users against the Content Server. This value should match the name of an existing external application connection. See also listExtAppConnections. If extAppId is not set, no change is made to the authentication method or external application ID.

If extAppId is set to an empty string, the authentication method used is IDENTITY_PROPAGATION. With this method, the application and Content Server use the same identity store to authenticate users.

**timeout**  
Length of time allowed to log in to the Content Server (in ms) before issuing a connection timeout message, and the RIDC socket timeout used for all service requests for connection types web, socket and socketssl.

If the timeout property is not set, the following values are used:

- **Login timeout** - the default concurrency timeout for the oracle.webcenter.content resource is used (30s or 30000ms). Refer to "Configuring Concurrency Management" in Oracle Fusion Middleware Performance and Tuning Guide for more information.
- **RIDC socket timeout** - the default RIDC socket timeout (60s or 60000ms) is used for all service requests for connection types socket, socketssl, or web.

If the timeout property is set and the connection type is socket, socketssl, or web, Oracle recommends that you do not specify a value less than 60000ms as this would reduce the RIDC socket timeout and increase the likelihood that long running requests will time out. For example, timeouts may occur during long running searches, long file uploads, or long copy operations.
The following example edits a socket-based connection to an Oracle WebCenter Content repository.

```
setJCRContentServerConnection(appName='webcenter',
   name='myContentServerConnection', socketType='socket',
   serverHost='myhost.com', serverPort=4444,
   extAppId='myExtApp', isPrimary=1)
```

The following example edits an SSL socket-based connection to an Oracle WebCenter Content repository.

```
setJCRContentServerConnection(appName='webcenter',
   name='myContentServerConnection', socketType='socketssl',
   serverHost='myhost.com', serverPort=8443,
   keystoreLocation='d:/keys/here', keystorePassword='T0PS3CR3T',
   privateKeyAlias='TekJansen', privateKeyPassword='LadyNocturne',
   extAppId='myExtApp', isPrimary=1)
```

The following example edits a JAX-WS (Java API for XML web services) connection to an Oracle WebCenter Content repository.

```
setJCRContentServerConnection(appName='webcenter',
   socketType='jaxws', url='http://myhost.com:9044/idcnativews',
   clientSecurityPolicy='oracle/wss10_saml_token_client_policy')
```

### 10.6.3 listJCRContentServerConnections

**Module:** Oracle WebCenter Portal

**Use with WLST:** Online

#### 10.6.3.1 Description

Without any arguments, this command lists all of the Oracle WebCenter Content repository connections that are configured for a named application.

#### 10.6.3.2 Syntax

```
listJCRContentServerConnections(appName, [verbose, name, server, applicationVersion])
```
### 10.6.3.3 Examples

The following example lists Oracle WebCenter Content repository connections configured for WebCenter Portal (webcenter):

```
weblogic/serverConfig> listJCRContentServerConnections(appName='webcenter')
```

The following example lists all properties of the Oracle WebCenter Content repository connection named myContentServerConnection1. The connection named myContentServerConnection1 must exist and be an Oracle WebCenter Content repository connection. If, for example, you specify an Oracle Portal connection, the properties are not listed and an error displays.

```
weblogic/serverConfig> listJCRContentServerConnections(appName='webcenter',
verbose=1, name='myContentServerConnection1')
```

### 10.6.4 createJCRPortalConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.6.4.1 Description

Creates an Oracle Portal repository connection for a named application.

#### 10.6.4.2 Syntax

```
createJCRPortalConnection(appName, name, dataSource, [extAppId, isPrimary, timeout, server, applicationVersion])
```
### 10.6.4.3 Example

The following example creates a Oracle Portal connection named `myPortalConnection` using the data source `jdbc/portalDS` and specifies that an external application, named `myExtApp`, is used for authentication:

```
<wl:script scope='weblogic'>
    createJCRPortalConnection(appName='myApp',
                             name='myPortalConnection',
                             dataSource='jdbc/portalDS',
                             extAppId='myExtApp',
                             isPrimary=1)
</wl:script>
```

### 10.6.5 setJCRPortalConnection

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Connection name. The name must be unique (across all connection types) within the application.</td>
</tr>
<tr>
<td>dataSource</td>
<td>JNDI DataSource location used to connect to the portal. For example: <code>jdbc/MyPortalDS</code>. The datasource must be on the server where the application is deployed.</td>
</tr>
<tr>
<td>extAppId</td>
<td>Optional. External application used to authenticate users against Oracle Portal. This value should match the name of an existing external application connection. See also <code>listExtAppConnections</code>. If <code>extAppId</code> is not set, no change is made to the authentication method or external application ID. If <code>extAppId</code> is set to an empty string, the authentication method used is <code>IDENTITY_PROPAGATION</code>. With this method, the application and Oracle Portal use the same identity store to authenticate users.</td>
</tr>
<tr>
<td>timeout</td>
<td>Optional. Length of time allowed to log in to Oracle Portal (in ms) before issuing a connection timeout message. If no timeout is set, the default concurrency timeout for the <code>oracle.webcenter.content</code> resource is used (30s or 30000ms). Refer to &quot;Configuring Concurrency Management&quot; in Oracle Fusion Middleware Performance and Tuning Guide for more information.</td>
</tr>
<tr>
<td>isPrimary</td>
<td>Optional. Valid string values are 1 (true) and 0 (false). 1 specifies that this connection is the primary connection used for document services in the application. This argument defaults to 0. When omitted or set to 0, the primary connection used for documents does not change. In WebCenter Portal, the primary connection must be an Oracle WebCenter Content repository connection.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

### 10.6.5 Module: Oracle WebCenter Portal

Use with WLST: Online
10.6.5.1 Description
Edits an existing Oracle Portal connection. This command requires that you specify values for either the dataSource or isPrimary argument.

10.6.5.2 Syntax
setJCRPortalConnection(appName, name, [dataSource, extAppId, timeout, isPrimary, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing Oracle Portal connection.</td>
</tr>
<tr>
<td>dataSource</td>
<td>Optional. JNDI DataSource location used to connect to the portal. For example: jdbc/MyPortalDS. The datasource must be on the server where the application is deployed.</td>
</tr>
<tr>
<td>extAppId</td>
<td>Optional. External application used to authenticate users against Oracle Portal. This value should match the name of an existing external application connection. See also listExtAppConnections. If extAppId is not set, no change is made to the authentication method or external application ID. If extAppId is set to an empty string, the authentication method used is IDENTITY_PROPAGATION. With this method, the application and Oracle Portal use the same identity store to authenticate users.</td>
</tr>
<tr>
<td>timeout</td>
<td>Optional. Length of time allowed to log in to Oracle Portal (in ms) before issuing a connection timeout message. If no timeout is set, the default concurrency timeout for the oracle.webcenter.content resource is used (30s or 30000ms). Refer to “Configuring Concurrency Management” in Oracle Fusion Middleware Performance and Tuning Guide for more information.</td>
</tr>
<tr>
<td>isPrimary</td>
<td>Optional. Valid string values are 1 (true) and 0 (false). 1 specifies that this connection is the primary connection used for document services in the application. This argument defaults to 0. When omitted or set to 0, the primary connection used for documents does not change. In WebCenter Portal, the primary connection must be an Oracle WebCenter Content repository connection.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.6.5.3 Example
The following example edits Oracle Portal repository connection details:

```bash
wls:/weblogic/serverConfig> setJCRPortalConnection(appName='webcenter', name='myPortalConnection', dataSource='/newPortalDS', extAppId='myExtApp', isPrimary=0)
```
10.6.6 listJCRPortalConnections

Module: Oracle WebCenter Portal

Use with WLST: Online

10.6.6.1 Description

Without any arguments, this command lists all of the Oracle Portal connections that are configured for a named application.

10.6.6.2 Syntax

listJCRPortalConnections(appName, [verbose, name, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Displays content repository connection details in verbose mode. Valid options are 1 (true) and 0 (false). When set to 1, listJCRPortalConnections lists all Oracle Portal connections that are configured for the application, along with their details. When set to 0, only connection names are listed. This argument defaults to 0.</td>
</tr>
<tr>
<td>name</td>
<td>Optional. Name of an existing Oracle Portal connection. When specified you can view connection details for a specific Oracle Portal connection. If you supply a value for name, you must supply a value for verbose.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.6.6.3 Example

The following example lists all of the Oracle Portal connections that are configured for a WebCenter Portal (webcenter):

```
wls:/weblogic/serverConfig> listJCRPortalConnections(appName='webcenter', verbose=1, name='myPortalConnection')
```

10.6.7 createJCRFileSystemConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

10.6.7.1 Description

Creates a connection to a file system repository.
**10.6.7.2 Syntax**

```
createJCRFileSystemConnection(appName, name, path, [isPrimary, server, applicationVersion])
```

**Argument** | **Definition**
---|---
`appName` | Name of the application in which to perform this operation.
  For WebCenter Portal, the application name is always `webcenter`.
  For Portal Framework applications, specify the appropriate name.

`name` | Connection name. The name must be unique (across all connection types) within the application.

`path` | Full path to a folder whose contents you want to expose through this file system connection.
  For example, if you have a folder called `C:\ProjectDocuments` and you want to expose content under that folder in your application, you need to specify this folder as the path argument to this command.

`isPrimary` | Optional. Valid values are 1 (true) and 0 (false).
  1 specifies that this connection is the primary connection used for document services.
  When set to 0, and when the specified connection is the primary connection used for document services, the primary connection is reset. If this parameter is not set, the primary connection used for document services does not change. This argument has no default.
  In WebCenter Portal, the primary connection must be an Oracle WebCenter Content repository connection.

`server` | Optional. Name of the managed server where the application is deployed. For example, `WC_Spaces`.
  Required when applications with the same name are deployed to different servers and also when you have a cluster.

`applicationVersion` | Optional. Version number of the deployed application. Required if more than one version of the application is deployed.

**Note:** File system connections must not be used in production or enterprise application deployments. This feature is provided for development purposes only.

**10.6.8 setJCRFileSystemConnection**

Module: Oracle WebCenter Portal

Use with WLST: Online

**10.6.8.1 Description**

Edits an existing file system repository connection. This command requires that you specify values for either the `path` or `isPrimary` arguments.
10.6.8.2 Syntax

setJCRFileSystemConnection(appName, name, [path, isPrimary, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td>name</td>
<td>Name for the connection to be used for document services.</td>
</tr>
<tr>
<td>path</td>
<td>Full path to a folder whose contents you want to expose through this file system connection.</td>
</tr>
<tr>
<td>isPrimary</td>
<td>Optional. Valid values are 1 (true) and 0 (false).</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

**Note:** File system connections must not be used in production or enterprise application deployments. This feature is provided for development purposes only.

10.6.8.3 Example

The following example edits connection details for a file system repository.

```
  wls:/weblogic/serverConfig> setJCRFileSystemConnection(appName='webcenter', name='FSAConnection', path='C:\ProjectDocuments')
```

10.6.9 listJCRFileSystemConnections

Module: Oracle WebCenter Portal

Use with WLST: Online

10.6.9.1 Description

Without any arguments, this command lists all of the file system connections that are configured for a named application.
10.6.9.2 Syntax

listJCRFileSystemConnections(appName, [verbose, name, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always &quot;webcenter&quot;. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Displays content repository connection details in verbose mode. Valid options are 1 (true) and 0 (false). When set to 1, listJCRFileSystemConnections lists all file system connections that are configured for a named application, along with their details. When set to 0, only connection names are listed. This argument defaults to 0.</td>
</tr>
<tr>
<td>name</td>
<td>Optional. Name of an existing file system connection. When specified you can view connection details for a specific file system connection. If you supply a value for name, you must supply a value for verbose.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.6.9.3 Examples

The following example lists all of the file system connections that are configured for WebCenter Portal (webcenter):

wls:/weblogic/serverConfig> listJCRFileSystemConnections(appName='webcenter')

The following example lists all of the file system connections that are configured, in verbose mode:

wls:/weblogic/serverConfig> listJCRFileSystemConnections(appName='webcenter', verbose=1)

10.6.10 createJCRSharePointConnection

Module: Oracle WebCenter Portal
Use with WLST: Online

10.6.10.1 Description

Creates a connection to a Microsoft SharePoint 2007 repository for a named application.

---

**Note:** File system connections must not be used in production or enterprise application deployments. This feature is provided for development purposes only.
## 10.6.10.2 Syntax

```bash
createJCRSharePointConnection(appName, name, url, [likeLimit, extAppId, timeout, isPrimary, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Connection name. The name must be unique (across all connection types) within the application.</td>
</tr>
<tr>
<td>url</td>
<td>Web address of the SharePoint site to which you want to connect. For example, if the SharePoint site address is <code>http://mysharepoint.mycompany.com</code>, enter this value for the <code>url</code> argument.</td>
</tr>
<tr>
<td>likeLimit</td>
<td>Optional. Number of characters the LIKE operator matches. The default is 64. The SharePoint query language can use a LIKE keyword to constrain URL queries (document paths) that match a search pattern. By default, the LIKE operator supports a pattern match on strings up to 64 characters. Use this argument to specify a different character limit (any positive integer between 1 and 64) or enter <code>likeLimit=0</code> to disable the LIKE limit, that is, always send the full query string to the Microsoft SharePoint server. As Oracle recommends the default value (64), there is no need to specify this argument when you create a connection using the WLST command <code>createJCRSharePointConnection</code>. <strong>Note:</strong> Only specify a value above 64 if your SharePoint instance supports LIKE queries on URLs greater than 64 characters.</td>
</tr>
<tr>
<td>extAppId</td>
<td>Optional. External application used to authenticate users against the SharePoint repository. This value should match the name of an existing external application connection. See also <code>listExtAppConnections</code>. If <code>extAppId</code> is not set, the SharePoint repository connection will not work. <code>extAppId</code> can be set or changed at any time using the <code>setJCRSharePointConnection</code> command.</td>
</tr>
<tr>
<td>timeout</td>
<td>Optional. Length of time allowed to log in to the SharePoint repository (in ms) before issuing a connection timeout message. If no timeout is set, the default concurrency timeout for the <code>oracle.webcenter.content</code> resource is used (30s or 30000ms). Refer to &quot;Configuring Concurrency Management&quot; in Oracle Fusion Middleware Performance and Tuning Guide for more information.</td>
</tr>
<tr>
<td>isPrimary</td>
<td>Optional. Valid values are 1 (true) and 0 (false). 1 specifies that this connection is the primary connection used for document services. The argument defaults to 0. If this parameter is omitted, the primary connection used for document services does not change. In WebCenter Portal, the primary connection must be an Oracle WebCenter Content repository connection.</td>
</tr>
</tbody>
</table>
Example
The following example creates a connection to a Microsoft SharePoint site.

```
<serverName>/weblogic/serverConfig> createJCRSharePointConnection(appName='webcenter', name='MySPConnection', url='http://mysharepoint.mycompany.com', extAppId='myExtApp')
```

### setJCRSharePointConnection

**Module:** Oracle WebCenter Portal

**Use with WLST:** Online

#### Description
Edits an existing Microsoft SharePoint 2007 repository connection. This command requires that you specify values for `appName` and `name`, plus at least one additional argument.

#### Syntax

```
setJCRSharePointConnection(appName, name, [url, likeLimit, extAppId, timeout, isPrimary, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td><code>name</code></td>
<td>Name of an existing SharePoint connection.</td>
</tr>
<tr>
<td><code>url</code></td>
<td>Optional. Web address of the SharePoint site to which you want to connect. For example, if the SharePoint site address is <code>http://mysharepoint.mycompany.com</code>, enter this value for the <code>url</code> argument.</td>
</tr>
</tbody>
</table>
Example

The following example edits SharePoint repository connection details.

```wls:/weblogic/serverConfig> setJCRSharePointConnection(appName='webcenter', name='MySPConnection', url='http://mysharepoint.mycompany.com', extAppId='myExtApp')```
10.6.12 listJCRSharePointConnections

Module: Oracle WebCenter Portal
Use with WLST: Online

10.6.12.1 Description
Without any arguments, this command lists all of the SharePoint connections that are configured for a named application.

10.6.12.2 Syntax
listJCRSharePointConnections(appName, [verbose, name, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Displays SharePoint connection details in verbose mode.</td>
</tr>
<tr>
<td></td>
<td>Valid options are 1 (true) and 0 (false).</td>
</tr>
<tr>
<td></td>
<td>When set to 1, listJCRSharePointConnections lists all SharePoint connections</td>
</tr>
<tr>
<td></td>
<td>that are configured for an application, along with their details.</td>
</tr>
<tr>
<td></td>
<td>When set to 0, only connection names are listed.</td>
</tr>
<tr>
<td></td>
<td>This argument defaults to 0.</td>
</tr>
<tr>
<td>name</td>
<td>Optional. Name of an existing SharePoint connection. When specified you can</td>
</tr>
<tr>
<td></td>
<td>view connection details for a specific SharePoint connection. If you supply</td>
</tr>
<tr>
<td></td>
<td>a value for name, you must supply a value for verbose.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different</td>
</tr>
<tr>
<td></td>
<td>servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more</td>
</tr>
<tr>
<td></td>
<td>than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.6.12.3 Example
The following example lists the names of all the SharePoint connections that are configured for WebCenter Portal (webcenter):

    wls:/weblogic/serverConfig> listJCRSharePointConnections(appName='webcenter')

The following example lists connection details for all of the SharePoint connections that are configured for WebCenter Portal (webcenter):

    wls:/weblogic/serverConfig> listJCRSharePointConnections(appName='webcenter', verbose=1)

10.6.13 listDocumentsSpacesProperties

Module: Oracle WebCenter Portal
Use with WLST: Online
10.6.13.1 Description
Lists properties for the back-end Oracle WebCenter Content repository that is being used by WebCenter Portal to store portal-specific documents and Home portal documents. This command is only valid for WebCenter Portal.

10.6.13.2 Syntax
listDocumentsSpacesProperties(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different</td>
</tr>
<tr>
<td></td>
<td>servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more</td>
</tr>
<tr>
<td></td>
<td>than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.6.13.3 Example
The following example lists properties for the back-end Oracle WebCenter Content repository that is being used by a WebCenter Portal instance (webcenter) to store portal-specific documents and Home portal documents.

wls:/weblogic/serverConfig> listDocumentsSpacesProperties(appName='webcenter')

The Documents Spaces container is '/WebCenter1109'
The Documents repository administrator is 'sysadmin'
The Documents Spaces container is '/WebCenter1109'
The Documents primary connection is 'myOCSCConnection'

10.6.14 setDocumentsSpacesProperties
Module: Oracle WebCenter Portal
Use with WLST: Online

10.6.14.1 Description
Modifies properties for the back-end Oracle WebCenter Content repository that is being used by WebCenter Portal to store portal-related data. This command is only valid for WebCenter Portal.

10.6.14.2 Syntax
setDocumentsSpacesProperties(appName, [spacesRoot, adminUserName, applicationName, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
</tbody>
</table>
The following example modifies connection properties for the back-end Oracle WebCenter Content repository that is being used by WebCenter Portal to store portal-related documents:

```
setDocumentsSpacesProperties(appName='webcenter',
spacesRoot='/AccountingSpaces', adminUserName='admin',
applicationName='WCAccounting')
```

The following example modifies the administrator's user name for the back-end Oracle WebCenter Content repository that is being used by WebCenter Portal to store portal-related documents:

```
```

---

### 10.6.14.3 Examples

The following example modifies connection properties for the back-end Oracle WebCenter Content repository that is being used by WebCenter Portal to store portal-related documents:

```
setDocumentsSpacesProperties(appName='webcenter',
spacesRoot='/AccountingSpaces', adminUserName='admin',
applicationName='WCAccounting')
```

The following example modifies the administrator's user name for the back-end Oracle WebCenter Content repository that is being used by WebCenter Portal to store portal-related documents:

```
```
10.6.15 deleteDocumentsSpacesProperties

Module: Oracle WebCenter Portal
Use with WLST: Online

10.6.15.1 Description
Deletes properties for the back-end Oracle WebCenter Content repository used by WebCenter Portal, that is the adminUserName, appName, and spacesRoot. This command is only valid for WebCenter Portal.

10.6.15.2 Syntax
deleteDocumentsSpacesProperties(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.6.15.3 Example
The following example deletes connection properties (adminUserName, appName, spacesRoot) of the back-end Oracle WebCenter Content repository that is being used by WebCenter Portal:

wls:/weblogic/serverConfig> deleteDocumentsSpacesProperties(appName='webcenter')

10.6.16 exportFoldersGData

Module: Oracle WebCenter Portal
Use with WLST: Online

10.6.16.1 Description
Exports Folders_g data to a specified location so you can migrate the data to FrameworkFolders using the WLST command migrateFoldersGToFrameworkFolders. After migrating Folders_g data to FrameworkFolders, you can generate comparison reports from the exported Folders_g data to verify that all the files and folders are migrated to FrameworkFolders.

10.6.16.2 Syntax
exportFoldersGData(appName, server, [connectionName, directoryPath, applicationVersion])
Example

The following example exports the Folders_g data for a WebCenter Portal application deployed to the WC_Spaces managed server. WebCenter Portal’s default content server connection details are used, and Folders_g data exported to the default location, the WCP_ORACLE_HOME/common/wlst/FRAMEWORK_FOLDER_MIGRATION directory.

```python
exportFoldersGData(appName='webcenter',server='WC_Spaces')
```

The following example exports the Folders_g data for a WebCenter Portal application deployed to the WC_Spaces managed server. The content server connection named MyContentServerConnection is used, and Folders_g data exported to the /scratch/myTemp_Dir directory.

```python
exportFoldersGData(appName='webcenter',server='WC_Spaces',connectionName='MyContentServerConnection',directoryPath='/scratch/myTemp_Dir')
```

### 10.6.17 migrateFoldersGDataToFrameworkFolders

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.6.17.1 Description

Migrates the Folders_g data to FrameworkFolders and verifies the integrity of the migrated data.

#### 10.6.17.2 Syntax

```python
migrateFoldersGDataToFrameworkFolders(appName,server,contentDbConnectionUrl,conten
tDbUserName,[connectionName,directoryPath,reportMode,applicationVersion])
```
<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Name of the managed server where the application is deployed. For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: In a clustered environment where applications share the same content server connection (that is, connect to the same content server and share the same root folder and security group), run this command against only one managed server.</td>
</tr>
<tr>
<td>contentDbConnectionUrl</td>
<td>Connection URL for the database where WebCenter Content schema (named OCS) is present. Use the connection URL format host:port:sid.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: This argument is mandatory for the WebCenter Portal application but optional for Portal Framework applications.</td>
</tr>
<tr>
<td>contentDbUserName</td>
<td>User name of the WebCenter Content schema (named OCS) that you want to migrate.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: This argument is mandatory for the WebCenter Portal application but optional for Portal Framework applications.</td>
</tr>
<tr>
<td>connectionName</td>
<td>Optional. Content Server connection name. Only required if you want to migrate Folders_g data for a non default content server connection.</td>
</tr>
<tr>
<td></td>
<td>If not specified, the default content server connection is used, that is, the content server connection where isPrimary=1.</td>
</tr>
<tr>
<td>directoryPath</td>
<td>Optional. Directory containing the Folders_g data that you want to migrate.</td>
</tr>
<tr>
<td></td>
<td>If not specified, Folders_g data is migrated from WCP_ ORACLE_HOME/common/wlst/FRAMEWORK_FOLDER_MIGRATION directory. This is the default location for Folders_g data exported using the WLST command exportFoldersGData.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong>: The path specified in the directoryPath attribute must be the same that you specified while running the exportFoldersGData WLST command.</td>
</tr>
<tr>
<td>reportMode</td>
<td>Optional. Indicates whether to generate a report describing what will happen during migration that you can review prior to migrating the actual data. Valid values are true and false.</td>
</tr>
<tr>
<td></td>
<td>true - Generates a report that describes all the metadata changes that will be made when the Folders_g data is migrated to FrameworkFolders. No actual data migration takes place.</td>
</tr>
<tr>
<td></td>
<td>false - Pre-migration report not required. Folders_g data is migrated to FrameworkFolders. The default value is false.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.6.17.3 Example

The following example migrates the Folders_g data from the default directory (`WCP_ORACLE_HOME/common/wlst/FRAMEWORK_FOLDER_MIGRATION`) to FrameworkFolders and validates the migrated data for the WebCenter Portal application deployed to the WC_Spaces managed server. Migration is performed using the default Content Server connection details and the specified WebCenter Content database connection and user name.

```python
migrateFoldersGDataToFrameworkFolders(appName='webcenter', server='WC_Spaces', contentDbConnectionUrl='wccdbhost.example.com:wccdbport:wccdbsid', contentDbUserName='SCHEMA_PREFIX_OCS')
```

The following example migrates Folders_g data from the `/scratch/myTemp_Dir` directory to FrameworkFolders and validates the migrated data for the WebCenter Portal application deployed to the WC_Spaces managed server. Content Server connection named MyContentServerConnection and the specified WebCenter Content database connection and username are used to perform the migration.

```python
migrateFoldersGDataToFrameworkFolders(appName='webcenter', server='WC_Spaces', contentDbConnectionUrl='wccdbhost.example.com:wccdbport:wccdbsid', contentDbUserName='SCHEMA_PREFIX_OCS', connectionName='MyContentServerConnection', directoryPath='/scratch/myTemp_Dir')
```

The following example reports how Folders_g data will be migrated to FrameworkFolders for the WebCenter Portal application deployed to WC_Spaces. MyContentServerConnection connection details and specified WebCenter Content database connection and username are used to generate the migration report which is saved to the same location as the Folders_g data (`/scratch/myTemp_Dir`):

```python
migrateFoldersGDataToFrameworkFolders(appName='webcenter', server='WC_Spaces', contentDbConnectionUrl='wccdbhost.example.com:wccdbport:wccdbsid', contentDbUserName='SCHEMA_PREFIX_OCS', connectionName='MyContentServerConnection', directoryPath='/scratch/myTemp_Dir', reportMode='true')
```

The following example migrates the Folders_g data from the `/scratch/myTemp_Dir` directory to FrameworkFolders for the Portal Framework application named customapp, which is deployed to the server Custom_Server. MyContentServerConnection is the Content Server connection used to perform migration.

```python
migrateFoldersGDataToFrameworkFolders(appName='customapp', server='Custom_Server', connectionName='MyContentServerConnection', directoryPath='/scratch/myTemp_Dir')
```

10.7 Discussions and Announcements

Use the commands listed in Table 10–8 to manage discussions server connections for WebCenter Portal and Portal Framework applications.

Configuration changes made using these WLST commands are only effective after your restart the Managed Server on which the application is deployed. For details, see Oracle Fusion Middleware Administering Oracle WebCenter Portal.
10.7.1 createDiscussionForumConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

10.7.1.1 Description

Creates a new discussions server connection for a named application.

Discussions and announcements both require a discussions server connection. Discussions and announcements both use the same discussions server connection.

Table 10–8 Discussion and Announcement WLST Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>createDiscussionForumConnection</td>
<td>Create a new discussions server connection for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>setDiscussionForumConnection</td>
<td>Edit an existing discussions server connection.</td>
<td>Online</td>
</tr>
<tr>
<td>setDefaultDiscussionForumConnection</td>
<td>Specify the default connection for discussions and announcements.</td>
<td>Online</td>
</tr>
<tr>
<td>listDiscussionForumConnections</td>
<td>List all of the discussions server connections that are configured for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>listDefaultDiscussionForumConnection</td>
<td>List the default discussions server connection for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>setDiscussionForumConnectionProperty</td>
<td>Set an additional discussions server connection property.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteDiscussionForumConnectionProperty</td>
<td>Delete a discussions server connection property.</td>
<td>Online</td>
</tr>
<tr>
<td>setDiscussionForumServiceProperty</td>
<td>Specify defaults for discussions.</td>
<td>Online</td>
</tr>
<tr>
<td>removeDiscussionForumServiceProperty</td>
<td>Remove defaults for discussions.</td>
<td>Online</td>
</tr>
<tr>
<td>listDiscussionForumServiceProperties</td>
<td>List discussions settings.</td>
<td>Online</td>
</tr>
<tr>
<td>setAnnouncementServiceProperty</td>
<td>Specify defaults for announcements.</td>
<td>Online</td>
</tr>
<tr>
<td>removeAnnouncementServiceProperty</td>
<td>Remove defaults for announcements.</td>
<td>Online</td>
</tr>
<tr>
<td>listAnnouncementServiceProperties</td>
<td>List announcements settings.</td>
<td>Online</td>
</tr>
<tr>
<td>addDiscussionsServerAdmin</td>
<td>Grant system administrator permissions on the discussions server to a user or a group.</td>
<td>Online</td>
</tr>
<tr>
<td>addDiscussionsCategoryAdmin</td>
<td>Grant category administrator permissions on the discussions server to a user or a group.</td>
<td>Online</td>
</tr>
<tr>
<td>syncDiscussionServerPermissions</td>
<td>Synchronizes discussion server permissions for subportals that inherit security from their parent.</td>
<td>Online</td>
</tr>
<tr>
<td>setDiscussionsServerProperty</td>
<td>Set discussions server properties.</td>
<td>Online</td>
</tr>
<tr>
<td>getDiscussionsServerProperty</td>
<td>Return discussions server property values.</td>
<td>Online</td>
</tr>
<tr>
<td>removeDiscussionsServerProperty</td>
<td>Remove current discussions server property values.</td>
<td>Online</td>
</tr>
</tbody>
</table>
While you can register multiple discussions server connections for an application, only one connection is used for discussion and announcement services - the default (or active) connection.

10.7.1.2 Syntax

```java
createDiscussionForumConnection(appName, name, url, adminUser,
[timeout, default, policyURIForAuthAccess, policyURIForPublicAccess,
recipientKeyAlias, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Connection name. The name must be unique (across all connection types) within the application.</td>
</tr>
<tr>
<td>url</td>
<td>URL of the discussions server hosting discussion forums and announcements. For example: <a href="http://myhost:8888/owc_discussions">http://myhost:8888/owc_discussions</a>.</td>
</tr>
<tr>
<td>adminUser</td>
<td>Name of the discussions server administrator. This account is used by discussions and announcements to perform administrative operations on behalf of WebCenter Portal users. This account is mostly used for managing discussions and announcements in the out-of-the-box WebCenter Portal application. It is not necessary for this user to be a super admin. However, the user must have administrative privileges on the current application root category for WebCenter Portal, that is, the category (on the discussions server) under which all portal-related discussions and announcements are stored.</td>
</tr>
</tbody>
</table>
| policyURIForAuthAccess    | Optional. URI to the SAML token based policy required for authenticated access to the discussions server web service. The client policy specified must be compatible with the service policy that is configured for the OWCDiscussionsServiceAuthenticated endpoint in the discussions server. Out-of-the-box, the default service policy is WSS 1.0 SAML Token Service Policy (oracle/wss10_saml_token_service_policy). Valid client policy values include:  
  - oracle/wss10_saml_token_client_policy (WSS 1.0 SAML Token Client Policy)
  - oracle/wss11_saml_token_with_message_protection_client_policy (WSS 1.1 SAML Token with Message Protection Client Policy)
  - GPA (Global Policy Attachment) - Use GPA if your environment supports Global Policy Attachments. In addition, ensure that the default policy is detached from the OWCDiscussionsServiceAuthenticated endpoint in the discussions server using the WLST command detachWebServicePolicy or Enterprise Manager. See also "Managing Announcements and Discussions" in the Oracle Fusion Middleware Administering Oracle WebCenter Portal. |
10.7.1.3 Example

The following example creates a discussions server connection for WebCenter Portal:

```bash
wls:/weblogic/serverConfig>createDiscussionForumConnection(appName='webcenter',
name='MyDiscussionServer', url='http://myhost.com:8888/owc_discussions',
adminUser='admin', policyURIForAuthAccess='oracle/wss10_saml_token_client_policy',
default=0)
```

**Argument Definition**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
</table>
| `policyURIForPublicAccess`      | Optional. URI to the policy required to enforce message security and integrity for public access to the discussions server Web service. Default value is `oracle/no_authentication_client_policy`. The client policy specified must be compatible with the service policy that is configured for the `OWCDiscussionsServicePublic` endpoint in the discussions server. Out-of-the-box, a service policy is not configured for public access (`oracle/no_authentication_client_policy`). Valid client policy values include:  
  - `oracle/no_authentication_client_policy` (None)  
  - `oracle/wss11_with_message_protection_client_policy` (WSS 1.1 Message Protection Client Policy)  
  - GPA (Global Policy Attachment) - Use GPA if your environment supports Global Policy Attachments. In addition, you must ensure that the default policy attached to the `OWCDiscussionsServicePublic` endpoint in the discussions server is set to `oracle/no_authentication_service_policy`. |
| `recipientKeyAlias`             | Optional. Recipient key alias to be used for message protected policies (applicable to the `OWCDiscussionsServicePublic` and `OWCDiscussionsServiceAuthenticated` endpoints). This is the alias to the certificate that contains the public key of the discussions server in the configured keystore. The default is null. See also "Configuring WS-Security" in the Oracle Fusion Middleware Administering Oracle WebCenter Portal. |
| `timeout`                       | Optional. Length of time (in seconds) discussions waits for a response from the discussions server before issuing a connection timeout message. This argument defaults to `-1`. When set to `-1`, the service default (10 seconds) applies. |
| `default`                       | Optional. Indicates that this connection is the default connection for discussions and announcements. Valid options are 1 (true) and 0 (false). When set to 1, discussions and announcements both use this connection. When set to 0, the connection is not used. The default is 0. |
| `server`                        | Optional. Name of the managed server where the application is deployed. For example, `WC_Spaces`. Required when applications with the same name are deployed to different servers and also when you have a cluster. |
| `applicationVersion`            | Optional. Version number of the deployed application. Required if more than one version of the application is deployed. |
10.7.2 setDiscussionForumConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

10.7.2.1 Description

Edits an existing discussions server connection. Use this command to update connection attributes.

The connection is created using the createDiscussionForumConnection command.

10.7.2.2 Syntax

setDiscussionForumConnection(appName, name, [url, adminUser, policyURIForAuthAccess, policyURIForPublicAccess, recipientKeyAlias, timeout, default, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing discussions server connection.</td>
</tr>
<tr>
<td>url</td>
<td>Optional. URL to the discussions server.</td>
</tr>
<tr>
<td>adminUser</td>
<td>Name of the discussions server administrator. This account is used by discussions to perform administrative operations on behalf of WebCenter Portal users. This account is mostly used for managing discussions and announcements in the out-of-the-box WebCenter Portal application. It is not necessary for this user to be a super admin. However, the user must have administrative privileges on the current application root category for WebCenter Portal, that is, the category (on the discussions server) under which all portal-related discussions and announcements are stored.</td>
</tr>
</tbody>
</table>
| policyURIForAuthAccess | Optional. URI to the SAML token based policy required for authenticated access to the discussions server web service. The client policy specified must be compatible with the service policy that is configured for the OWCDiscussionsServiceAuthenticated endpoint in the discussions server. Out-of-the-box, the default service policy is WSS 1.0 SAML Token Service Policy (oracle/wss10_saml_token_service_policy). Valid client policy values include:  
  - oracle/wss10_saml_token_client_policy (WSS 1.0 SAML Token Client Policy)  
  - oracle/wss11_saml_token_with_message_protection_client_policy (WSS 1.1 SAML Token with Message Protection Client Policy)  
  - GPA (Global Policy Attachment) - Use GPA if your environment supports Global Policy Attachments. In addition, ensure that the default policy is detached from the OWCDiscussionsServiceAuthenticated endpoint in the discussions server using the WLST command detachWebServicePolicy or Enterprise Manager. See also "Managing Announcements and Discussions" in the Oracle Fusion Middleware Administering Oracle WebCenter Portal.
<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
</table>
| policyURIForPublicAccess  | Optional. URI to the policy required to enforce message security and integrity for public access to the discussions server web service. Default value is `oracle/no_authentication_client_policy`. The client policy specified must be compatible with the service policy that is configured for the OWCDiscussionsServicePublic endpoint in the discussions server. Out-of-the-box, a service policy is not configured for public access (`oracle/no_authentication_client_policy`). Valid client values include:  
  - `oracle/no_authentication_client_policy` (None)  
  - `oracle/wss11_with_message_protection_client_policy` (WSS 1.1 Message Protection Client Policy)  
  - GPA (Global Policy Attachment) - Use GPA if your environment supports Global Policy Attachments. In addition, you must ensure that the default policy attached to the OWCDiscussionsServicePublic endpoint in the discussions server is set to `oracle/no_authentication_service_policy`. |
| recipientKeyAlias         | Optional. Recipient key alias to be used for message protected policies applicable to the OWCDiscussionsServicePublic and OWCDiscussionsServiceAuthenticated endpoints. This is the alias to the certificate that contains the public key of the discussions server in the configured keystore. The default is null. See also “Configuring WS-Security” in the Oracle Fusion Middleware Administering Oracle WebCenter Portal. |
| timeout                   | Optional. Length of time (in seconds) discussions and announcements wait for a response from the discussions server before issuing a connection timeout message. This argument defaults to -1. When set to -1, the service default (10 seconds) applies.                                                                                                                                                                                                                                           |
| default                   | Optional. Indicates that this connection is the default connection for discussions and announcements. Required only if more than one connection is defined. Valid options are 1 (true) and 0 (false). When set to 1, discussions and announcements use this connection. When set to 0, the connection is not used. The default is 0. To specify that discussion and announcements use this connection, change the value from 0 to 1. To disable this connection, use the removeDiscussionForumServiceProperty command:  
  `removeDiscussionForumServiceProperty('appName='webcenter', property='selected.connection')`  
  **Note:** While you can register multiple discussions server connections for an application, only one connection is used for discussions and announcements— the default (or active) connection. |
| server                    | Optional. Name of the managed server where the application is deployed. For example, `WC_Spaces`. Required when applications with the same name are deployed to different servers and also when you have a cluster.                                                                                                                                                                                                                             |
| applicationVersion        | Optional. Version number of the deployed application. Required if more than one version of the application is deployed.                                                                                                                                                                                                                                                                                                                                                     |
10.7.2.3 Example
The following example updates attributes for a secure discussions server connection named MyDiscussionsServer.

wls:/weblogic/serverConfig> setDiscussionForumConnection(appName='webcenter', name='MyDiscussionServer', url='http://myhost.com:7786/owc_discussions', adminUser='admin', policyURIForAuthAccess='oracle/wss10_saml_token_client_policy', default=1)

10.7.3 setDiscussionForumConnectionProperty
Module: Oracle WebCenter Portal
Use with WLST: Online

10.7.3.1 Description
Sets a discussions server connection property. Use this command when additional parameters are required to connect to your discussions server.

This command provides an extensible way to add any connection property using a key and a value. (You are not limited to connection properties specified by createDiscussionForumConnection and setDiscussionForumConnection.)

Note: Do not use the setDiscussionForumConnectionProperty to set connection properties available through createDiscussionForumConnection or setDiscussionForumConnection. Attempting to do so, has no effect.

All known, additional connection properties are listed in Table 10–9, "Additional Discussion Server Connection Properties".

Table 10–9 Additional Discussion Server Connection Properties

<table>
<thead>
<tr>
<th>Additional Connection Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>application.root.category.id</td>
<td>(WebCenter Portal only) Application root category ID on the discussions server under which all discussion forums are stored. For example, if set to 3, then all forums are stored in the category with the ID 3.</td>
</tr>
<tr>
<td>linkURL</td>
<td>URL used to link users to the discussions server’s Admin Console. Only required if it is different to the url argument specified using the createDiscussionForumConnection or setDiscussionForumConnection command. For example, when SSO or HTTPS is configured. Use the following format to specify an alternative public external URL: protocol://host:port For example: <a href="http://example.com:7777">http://example.com:7777</a></td>
</tr>
</tbody>
</table>

10.7.3.2 Syntax
setDiscussionForumConnectionProperty(appName, name, key, value, [secure, server, applicationVersion])
10.7.3.3 Example

The following example configures the category used to store discussion forums for a discussions server connection named `MyDiscussionServer`:

```wls:/weblogic/serverConfig> setDiscussionForumConnectionProperty
(appName='webcenter', name='MyDiscussionServer',
key='application.root.category.id', value='3')```

The following example configures an alternative, public external URL so users can access the discussions server’s Admin Console:

```wls:/weblogic/serverConfig> setDiscussionForumConnectionProperty
(appName='webcenter', name='MyDiscussionServer', key='linkURL',
value='http://example.com:7777')```

The following example adds a custom discussions server connection property called `myProperty1` with a value `propertyValue1`:

```wls:/weblogic/serverConfig> setDiscussionForumConnectionProperty
(appName='webcenter', name='MyDiscussionServer', key='myProperty1',
value='propertyValue1')```

The following example adds a secured discussions server connection property called `securedProperty` with the value `secureValue`.

```wls:/weblogic/serverConfig> setDiscussionForumConnectionProperty
(appName='webcenter', name='MyDiscussionServer', key='securedProperty',
value='secureValue', secure=1)```

10.7.4 `deleteDiscussionForumConnectionProperty`

Module: Oracle WebCenter Portal

Use with WLST: Online
10.7.4.1 Description
Deletes a discussions server connection property. Take care when deleting connection properties because the connection may not work as expected if the configuration becomes invalid as a result.

This command can only delete additional connection properties added using the setDiscussionForumConnectionProperty command.

10.7.4.2 Syntax
deleteDiscussionForumConnectionProperty(appName, name, key, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing discussions server connection.</td>
</tr>
<tr>
<td>key</td>
<td>Name of the connection property you want to delete.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.7.3 Example
The following example deletes a discussions server connection property named `myProperty1`.

```
$ wls:/weblogic/serverConfig> deleteDiscussionForumConnectionProperty
(appName='webcenter', name='MyDiscussionServer', key='myProperty1')
```

10.7.5 listDiscussionForumConnections
Module: Oracle WebCenter Portal
Use with WLST: Online

10.7.5.1 Description
Lists all the discussions server connections that are configured for a named application.

10.7.5.2 Syntax
listDiscussionForumConnections(appName, [verbose, name, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
</tbody>
</table>
10.7.5.3 Examples

The following example lists the names of all of the discussions server connections that are currently configured for WebCenter Portal:

```
wls:/weblogic/serverConfig>listDiscussionForumConnections(appName='webcenter')
```

The following example lists connection names and details for all of the discussions server connections currently configured for WebCenter Portal:

```
wls:/weblogic/serverConfig>listDiscussionForumConnections(appName='webcenter', verbose=1)
```

The following example lists connection details for a discussions server connection named myDiscussionsServer.

```
wls:/weblogic/serverConfig>listDiscussionForumConnections(appName='webcenter', name='myDiscussionsServer')
```

10.7.6 listDefaultDiscussionForumConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

10.7.6.1 Description

Names the discussions server connection that discussions and announcements service are using, in a named application. While you can register multiple discussions server connections for an application, discussions and announcements only uses one connection—known as the default (or active) connection.

10.7.6.2 Syntax

```
listDefaultDiscussionForumConnection(appName, [verbose, server, applicationVersion])
```
Discussions and Announcements

WebCenter Portal Custom WLST Commands

10.7.6.3 Examples

The following example names the discussions server connection that discussions and announcements are using in WebCenter Portal:

```
wlst:/weblogic/serverConfig>listDefaultDiscussionForumConnection(appName='webcenter ')
```

The following example lists the name and details of the discussions server connection that discussions and announcements are using in WebCenter Portal:

```
wlst:/weblogic/serverConfig>
listDefaultDiscussionForumConnection(appName='webcenter', verbose=1)
```

10.7.7 setDefaultDiscussionForumConnection

Module: Oracle WebCenter Portal
Use with WLST: Online

10.7.7.1 Description

Specifies the default discussions server connection for discussions and announcements in a named application.

While you can register multiple discussions server connections with an application, discussions and announcements only uses one connection—this is known as the default (or active) connection.

10.7.7.2 Syntax

```
setDefaultDiscussionForumConnection(appName, name, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Valid options are 1 (true) and 0 (false). When set to 1, the name and details of the discussions server connections are listed. When set to 0, only the connection name displays. This argument defaults to 0.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.7.7.3 Example
The following example makes a connection named myDiscussionServer the default (or active) connection for discussions and announcements in WebCenter Portal:

```
wls:/weblogic/serverConfig> setDefaultDiscussionForumConnection
(appName='webcenter', name='myDiscussionServer')
```

10.7.8 setDiscussionForumServiceProperty
Module: Oracle WebCenter Portal
Use with WLST: Online

10.7.8.1 Description
Specifies default values for discussions, for a named application.

Configurable properties for discussions are listed in Table 10–10, ”Discussions - Configurable Properties”.

<table>
<thead>
<tr>
<th>Configuration Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>topics.fetch.size</td>
<td>Maximum number of topics fetched by discussions and displayed in the topics view.</td>
</tr>
<tr>
<td>forums.fetch.size</td>
<td>Maximum number of forums fetched by discussions and displayed in the forums view.</td>
</tr>
<tr>
<td>recentTopics.fetch.size</td>
<td>Maximum number of topics fetched by discussions and displayed in the recent topics view.</td>
</tr>
<tr>
<td>watchedTopics.fetch.size</td>
<td>Maximum number of topics fetched by discussions and displayed in the watched topics view.</td>
</tr>
<tr>
<td>watchedForums.fetch.size</td>
<td>Maximum number of forums fetched by discussions and displayed in the watched forums view.</td>
</tr>
<tr>
<td>application.root.category.id</td>
<td>Application root category ID on the discussions server under which all discussion forums are stored. For example, if set to 3, all forums are stored inside category 3.</td>
</tr>
<tr>
<td>ForumGatewayManager.AUTO_START</td>
<td>Communication through mail distribution lists can be published as discussion forum posts. This parameter starts or stops the gateway for this communication. For WebCenter Portal, the default value is 1, which means that as soon as you configure mail server settings through administration, the gateway starts. Set this to 0, and restart the managed server, to stop the gateway and disable this feature. For Portal Framework applications, the default value is 0. Set this to 1, and restart the managed server, to start the gateway and enable this feature.</td>
</tr>
</tbody>
</table>
10.7.8.2 Syntax

```java
setDiscussionForumServiceProperty(appName, property, value, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td>property</td>
<td>Name of the configuration property.</td>
</tr>
<tr>
<td>value</td>
<td>Value for the property.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.7.8.3 Example

The following example changes the default number of topics displayed in topics view.

```bash
classic/weblogic/serverConfig> setDiscussionForumServiceProperty
(appName='webcenter', property='topics.fetch.size', value='30')
```

10.7.9 removeDiscussionForumServiceProperty

Module: Oracle WebCenter Portal

Use with WLST: Online

10.7.9.1 Description

Removes the current value that is set for a discussions property. Use this command to remove any of the properties listed in Table 10-10, "Discussions - Configurable Properties".

Take care when using this command as removing values for these properties might cause unexpected behavior.

**Note:** Use this command syntax to disable the connection currently used for discussion and announcement services:

```bash
removeDiscussionForumServiceProperty('appName='webcenter', property='selected.connection')
```

This command forces the default connection argument to 0. See also, `setDiscussionForumConnection`.

10.7.9.2 Syntax

```java
removeDiscussionForumServiceProperty(appName, property, [server, applicationVersion])
```
Example
The following example clears the current `topics.fetch.size` property for discussions in WebCenter Portal:

```
wlsc:/weblogic/serverConfig> removeDiscussionForumServiceProperty
(appName='webcenter', property='topics.fetch.size')
```

### 10.7.10 listDiscussionForumServiceProperties

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.7.10.1 Description
Lists all configurable properties for discussions.

#### 10.7.10.2 Syntax

```
listDiscussionForumServiceProperties(appName, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>appName</strong></td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td><strong>property</strong></td>
<td>Name of the configuration property.</td>
</tr>
<tr>
<td><strong>server</strong></td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, <code>WC_Spaces</code>.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><strong>applicationVersion</strong></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

#### 10.7.10.3 Example
The following example lists configuration properties for discussions in WebCenter Portal:

```
wls:/weblogic/serverConfig>
listDiscussionForumServiceProperties(appName='webcenter')
```
10.7.11 setAnnouncementServiceProperty

Module: Oracle WebCenter Portal
Use with WLST: Online

10.7.11.1 Description

Specifies default values for announcements in a named application.

Configurable properties for announcements are listed in Table 10–11, "Announcements - Configurable Properties".

Table 10–11 Announcements - Configurable Properties

<table>
<thead>
<tr>
<th>Configuration Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>miniview.page_size</td>
<td>Maximum number of announcements displayed in the announcements mini view.</td>
</tr>
<tr>
<td>mainview.page_size</td>
<td>Maximum number of announcements displayed in the announcements main view.</td>
</tr>
<tr>
<td>linksview.page_size</td>
<td>Maximum number of announcements displayed in the announcements links view.</td>
</tr>
<tr>
<td>announcements.expiration.days</td>
<td>Number of days that announcements display and remain editable.</td>
</tr>
</tbody>
</table>

10.7.11.2 Syntax

setAnnouncementServiceProperty(appName, property, value, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>property</td>
<td>Name of the configuration property.</td>
</tr>
<tr>
<td>value</td>
<td>Property value.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.7.11.3 Example

The following example changes the default number of days that announcements display in WebCenter Portal:

wls:/weblogic/serverConfig>setAnnouncementServiceProperty(appName='webcenter', property='announcements.expiration.days', value='21')

10.7.12 removeAnnouncementServiceProperty

Module: Oracle WebCenter Portal
Use with WLST: Online
10.7.12.1 Description
Removes the current value that is set for an announcements property. Use this command to remove any of the properties listed in Table 10–11, “Announcements - Configurable Properties”.

Take care when using this command as removing values for these properties might cause unexpected behavior.

10.7.12.2 Syntax
removeAnnouncementServiceProperty(appName, property, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>property</td>
<td>Name of the configuration property.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.7.12.3 Example
The following example clears the announcements.expiration.days property for announcements in WebCenter Portal:

```
ws:/weblogic/serverConfig> removeAnnouncementServiceProperty
    (appName='webcenter', property='announcements.expiration.days')
```

10.7.13 listAnnouncementServiceProperties
Module: Oracle WebCenter Portal
Use with WLST: Online

10.7.13.1 Description
Lists all configurable properties for announcements, in a named application.

10.7.13.2 Syntax
listAnnouncementServiceProperties(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
</tbody>
</table>
Discussions and Announcements

10.7.13.3 Example
The following example lists configuration properties for announcements in WebCenter Portal:

```
wls:/weblogic/serverConfig>listAnnouncementServiceProperties(appName='webcenter')
```

10.7.14 addDiscussionsServerAdmin
Module: Oracle WebCenter Portal
Use with WLST: Online

10.7.14.1 Description
Grants system administrator permissions on the discussions server to a user or a group. This command is useful when you connect the discussions server to a new identity store that does not contain any of the current administrators.

10.7.14.2 Syntax
```
addDiscussionsServerAdmin(appName, name, [type, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the discussions server application in which to perform this operation. For example, owc_discussions.</td>
</tr>
<tr>
<td>name</td>
<td>Name of the user or group to add as an administrator on the discussions server.</td>
</tr>
<tr>
<td>type</td>
<td>Optional. Identifies the type of identity. Valid values are USER and GROUP. The default value is USER.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server on which the application is deployed. For example, WC_Collaboration. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.7.14.3 Example
The following example grants system administrator permissions on the discussions server to the user weblogic:

```
addDiscussionsServerAdmin(appName='owc_discussions', name='weblogic', type='USER')
```

The following example grants system administrator permissions on the discussions server to all users in the Administrators user group:

```
addDiscussionsServerAdmin(appName='owc_discussions', name='Administrators', type='GROUP')
```
addDiscussionsServerAdmin(appName='owc_discussions', name='Administrators', type='GROUP')

10.7.15 addDiscussionsCategoryAdmin

Module: Oracle WebCenter Portal
Use with WLST: Online

10.7.15.1 Description
Grants category administrator permissions on the discussions server to a user or a group for a specific category ID.

10.7.15.2 Syntax
addDiscussionsCategoryAdmin(appName, categoryId, name, [type, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the discussions server application in which to perform this operation. For example, owc_discussions.</td>
</tr>
<tr>
<td>categoryId</td>
<td>The ID (number) of a category on the discussions server.</td>
</tr>
<tr>
<td>name</td>
<td>Name of the user or group to add as an administrator for the category on the discussions server.</td>
</tr>
<tr>
<td>type</td>
<td>Optional. Identifies the type of identity. Valid values are USER and GROUP. The default value is USER.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server on which the application is deployed. For example, WC_Collaboration. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.7.15.3 Example
The following example grants category administrator permissions on the discussions server to the user weblogic for a category with an ID=2:
addDiscussionsCategoryAdmin(appName='owc_discussions', categoryId=2, name='weblogic', type='USER')

The following example grants category administrator permissions on the discussions server to all users in the Sales user group for a category with an ID=2:
addDiscussionsCategoryAdmin(appName='owc_discussions', categoryId=2, name='Sales', type='GROUP')

10.7.16 syncDiscussionServerPermissions

Module: Oracle WebCenter Portal
Use with WLST: Online
10.7.16.1 Description

(WebCenter Portal only) Synchronizes discussion server permissions for subportals that inherit security from their parent.

When you update discussions or announcement permissions for portal hierarchies in WebCenter Portal, the subportals do not automatically inherit the corresponding permission change on the discussions server. Therefore, whenever changes are made, you must run this command to synchronize discussions and announcement permissions within a portal hierarchy, such that subportals inherit the same discussions server permissions as their parent.

Note: To execute discussions server WLST commands, such as syncDiscussionServerPermissions, the user used to connect to the Admin Server must also have administrative privileges on the discussions server.

10.7.16.2 Syntax

syncDiscussionServerPermissions(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>The default value is webcenter (the application name for WebCenter Portal).</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.7.16.3 Example

The following example synchronizes permissions for discussions and announcement in WebCenter Portal, that is, subportals inherit the same discussions server permissions as their parent:

wls:/weblogic/serverConfig> syncDiscussionServerPermissions(appName='webcenter')

10.7.17 setDiscussionsServerProperty

Module: Oracle WebCenter Portal
Use with WLST: Online

10.7.17.1 Description

Sets a discussions server property. Use this command to set a system property on the discussions server.

Note: To execute discussions server WLST commands, such as setDiscussionsServerProperty, the user used to connect to the Admin Server must also have administrative privileges on the discussions server.
10.7.17.2 Syntax

```
setDiscussionsServerProperty(appName, key, value, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the discussions server application in which to perform this operation. For example, owc_discussions.</td>
</tr>
<tr>
<td><code>key</code></td>
<td>Name of the discussions server property.</td>
</tr>
<tr>
<td></td>
<td>For example, owc_discussions.sso.mode, AuthFactory.className, UserManager.className, GroupManager.className, owc_discussions.setup.complete_11.1.1.2.0, and so on.</td>
</tr>
<tr>
<td><code>value</code></td>
<td>Value for the discussions server property.</td>
</tr>
<tr>
<td><code>server</code></td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Collaboration.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><code>applicationVersion</code></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.7.17.3 Example

The following example sets properties that configures the discussions server for SSO, where example.com:8890/owc_discussions is the base URL of the webtier on which the discussions server is deployed:

```
wls:/weblogic/serverConfig>setDiscussionsServerProperty(appName='owc_discussions', key='owc_discussions.sso.mode', value='true')
wls:/weblogic/serverConfig>setDiscussionsServerProperty(appName='owc_discussions', key='jiveURL', value='example.com:8890/owc_discussions')
```

10.7.18 `getDiscussionsServerProperty`

Module: Oracle WebCenter Portal

Use with WLST: Online

10.7.18.1 Description

Returns the current value of a discussions server property.

---

**Note:** To execute discussions server WLST commands, such as `getDiscussionsServerProperty`, the user used to connect to the Admin Server must also have administrative privileges on the discussions server.

---

10.7.18.2 Syntax

```
getDiscussionsServerProperty(appName, key, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the discussions server application in which to perform this operation. For example, owc_discussions.</td>
</tr>
</tbody>
</table>
10.7.18.3 Example
The following examples return the current value for some key discussions server properties:

```
wlst:/weblogic/serverConfig>getDiscussionsServerProperty(appName='owc_discussions', key='AuthFactory.className')
wlst:/weblogic/serverConfig>getDiscussionsServerProperty(appName='owc_discussions', key='UserManager.className')
wlst:/weblogic/serverConfig>getDiscussionsServerProperty(appName='owc_discussions', key='GroupManager.className')
wlst:/weblogic/serverConfig>getDiscussionsServerProperty(appName='owc_discussions', key='owc_discussions.setup.complete_11.1.1.2.0')
```

10.7.19 removeDiscussionsServerProperty
Module: Oracle WebCenter Portal
Use with WLST: Online

10.7.19.1 Description
Removes the current value that is set for a discussions server property.

**Note:** To execute discussions server WLST commands, such as `removeDiscussionsServerProperty`, the user used to connect to the Admin Server must also have administrative privileges on the discussion server.

10.7.19.2 Syntax
```
removeDiscussionsServerProperty(appName, key, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the discussions server application in which to perform this operation. For example, <code>owc_discussions</code>.</td>
</tr>
<tr>
<td>key</td>
<td>Name of the discussions server property. For example, <code>owc_discussions.sso.mode</code>, <code>AuthFactory.className</code>, <code>UserManager.className</code>, <code>GroupManager.className</code>, <code>owc_discussions.setup.complete_11.1.1.2.0</code>, and so on.</td>
</tr>
</tbody>
</table>
Example

The following example removes the current value for the 'SSO mode' property on the discussions server:

```wls:/weblogic/serverConfig>removeDiscussionsServerProperty(appName='owc_discussions', key='owc_discussions.sso.mode')```

10.8 External Applications

Use the commands listed in Table 10–12 to manage external application connections for WebCenter Portal and Portal Framework applications.

Configuration changes made using these WLST commands are immediately available in the application, that is, you do not need to restart the managed server.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>createExtAppConnection</td>
<td>Create an external application connection, for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>setExtAppConnection</td>
<td>Edit an existing external application connection.</td>
<td>Online</td>
</tr>
<tr>
<td>listExtAppConnections</td>
<td>List individual or all external applications that are configured for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>addExtAppField</td>
<td>Add another login field for a specific external application connection.</td>
<td>Online</td>
</tr>
<tr>
<td>setExtAppField</td>
<td>Edit the value and display-to-user setting for a specific external application login field.</td>
<td>Online</td>
</tr>
<tr>
<td>removeExtAppField</td>
<td>Remove an external application login field.</td>
<td>Online</td>
</tr>
<tr>
<td>addExtAppCredential</td>
<td>Specify shared or public credentials for an external application.</td>
<td>Online</td>
</tr>
<tr>
<td>setExtAppCredential</td>
<td>Edit shared or public credentials for an external application.</td>
<td>Online</td>
</tr>
<tr>
<td>removeExtAppCredential</td>
<td>Remove shared or public credentials currently configured for an external application.</td>
<td>Online</td>
</tr>
</tbody>
</table>

10.8.1 createExtAppConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

10.8.1.1 Description

Creates an external application connection for a named application.
10.8.1.2 Syntax

```python
createExtAppConnection(appName, name, [displayName, url, authMethod, 
userFieldName, pwdFieldName, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Connection name. The name must be unique (across all connection types) within WebCenter Portal or your Portal Framework application.</td>
</tr>
<tr>
<td>displayName</td>
<td>Optional. External application display name. A user friendly name for the external application that WebCenter Portal users will recognize. The display name must be unique across all external applications within WebCenter Portal or your Portal Framework application.</td>
</tr>
<tr>
<td>url</td>
<td>Optional. External application login URL. To determine an application’s URL, navigate to the application’s login page and note down the URL for that page. For example: <code>http://login.yahoo.com/config/login</code></td>
</tr>
<tr>
<td>authMethod</td>
<td>Optional. Authentication mechanism used by the external application. Valid options are GET, POST, and BASIC. This argument defaults to POST.</td>
</tr>
<tr>
<td>userFieldName</td>
<td>Optional. Name that identifies the user name or user ID field on the external application’s login form. To find this name, look at the HTML source for the login page. This argument does not specify user credentials. Mandatory if authMethod is GET or POST and a login url is specified. Not required if BASIC authentication method is selected.</td>
</tr>
<tr>
<td>pwdFieldName</td>
<td>Optional. Name that identifies the password field on the external application’s login form. To find this name, look at the HTML source for the login page. This argument does not specify user credentials. Mandatory if authMethod is GET or POST and a login url is specified. Not required if BASIC authentication method is selected.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.8.1.3 Example

The following example creates a connection for an external application named My Yahoo!, in WebCenter Portal (`webcenter`).

```bash
wls:/weblogic/serverConfig> createExtAppConnection(appName='webcenter', 
name='yahoo', displayName='My Yahoo!', url='http://login.yahoo.com/config/login', 
authMethod='POST', userFieldName='login', pwdFieldName='password')
```

10.8.2 setExtAppConnection

Module: Oracle WebCenter Portal
Use with WLST: Online

10.8.2.1 Description
Edits an existing external application connection.

10.8.2.2 Syntax
```
setExtAppConnection(appName, name, [displayName, url, authMethod, userFieldName, pwdFieldName, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing external application connection.</td>
</tr>
<tr>
<td>displayName</td>
<td>Optional. External application display name. A user-friendly name for the external application that WebCenter Portal users will recognize. The display name must be unique across all external applications within WebCenter Portal or your Portal Framework application.</td>
</tr>
<tr>
<td>url</td>
<td>Optional. External application login URL. To determine an application's URL, navigate to the application's login page and note down the URL for that page.</td>
</tr>
<tr>
<td>authMethod</td>
<td>Optional. Authentication mechanism used by the external application. Valid options are GET, POST, and BASIC. This argument defaults to POST.</td>
</tr>
<tr>
<td>userFieldName</td>
<td>Optional. Name that identifies the user name or user ID field on the external application's login form. To find this name, look at the HTML source for the login page. This argument does not specify user credentials. Mandatory if authMethod is GET or POST and a login URL is specified but can be left blank if BASIC authentication method is selected.</td>
</tr>
<tr>
<td>pwdFieldName</td>
<td>Optional. Name that identifies the password field on the external application's login form. To find this name, look at the HTML source for the login page. This argument does not specify user credentials. Mandatory if authMethod is GET or POST, but can be left blank if BASIC authentication method is selected.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.8.2.3 Example
The following example updates the display name attribute for an external application named yahoo.

```
wls:/weblogic/serverConfig> setExtAppConnection(appName='webcenter', name='yahoo', displayName='My Favorite Yahoo!')
```
10.8.3 listExtAppConnections

Module: Oracle WebCenter Portal

Use with WLST: Online

10.8.3.1 Description

When used with only the `appName` argument, this command lists the names of all the external applications currently configured for a named WebCenter Portal or your Portal Framework application.

10.8.3.2 Syntax

```
listExtAppConnections(appName, [verbose, name, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, this is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td><code>verbose</code></td>
<td>Optional. Displays external application details in verbose mode. Valid options are 1 (true) and 0 (false). When set to 1, <code>listExtAppConnections</code> lists all of the external applications that are configured for WebCenter Portal or your Portal Framework application, along with their details. When set to 0, <code>listExtAppConnections</code> lists only the names of the external applications. This argument defaults to 0. If you set this argument to 0, do not specify the <code>name</code> argument.</td>
</tr>
<tr>
<td><code>name</code></td>
<td>Optional. Name of an existing external application connection. You can use this argument to view details about a specific connection.</td>
</tr>
<tr>
<td><code>server</code></td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><code>applicationVersion</code></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.8.3.3 Examples

The following example lists the names of all the external applications currently used by WebCenter Portal (`webcenter`):

```
$ wls:/weblogic/serverConfig> listExtAppConnections(appName='webcenter')
app1
app2
app3
```

The following example lists details for the external applications `app1`, `app2`, and `app3`.

```
$ wls:/weblogic/serverConfig> listExtAppConnections(appName='webcenter', verbose=1)
----
app1
----
Name: app1
Display Name: Application1
Login URL: http://app1
Authentication Method: POST
User Field Name: login
Password Field Name: passwd
Shared Credential: Disabled
Public Credential: Disabled

----
app2
----
Name: app2
Display Name: Application2
Login URL: http://app2
Authentication Method: POST
User Field Name: login
Password Field Name: passwd
Additional Fields: {Account1:1, Account2:DefVal:0}
Shared Credential: Disabled
Public Credential: Enabled

----
app3
----
Name: app3
Display Name: Application3
Authentication Method: POST
Shared Credential: Enabled
Public Credential: Enabled

The following example lists details for external application app1 only.

```bash
wls:/weblogic/serverConfig> listExtAppConnections(appName='webcenter', verbose=1, name='app1')
```

```
----
app1
----
Name: app1
Display Name: Application1
Login URL: http://app1
Authentication Method: POST
User Field Name: login
Password Field Name: passwd
Shared Credential: Disabled
Public Credential: Disabled
```

### 10.8.4 addExtAppField

**Module:** Oracle WebCenter Portal

**Use with WLST:** Online

#### 10.8.4.1 Description

Adds another login field for a specific external application connection. For example, in addition to user name and password, an external application may require other login criteria such as Host and MailAddress.

Optionally, additional login fields can appear on the external application's login for a user to specify.

If you add another login field and the external application uses shared or public credentials, you can use the WLST commands `addExtAppCredential` and `setExtAppCredential` to update the shared/public credentials. See Section 10.8.7,
"addExtAppCredential" and Section 10.8.8, "setExtAppCredential".

10.8.4.2 Syntax

```
addExtAppField(appName, name, fieldName, [fieldValue, displayToUser, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing external application connection.</td>
</tr>
<tr>
<td>fieldName</td>
<td>Login field name. The name that identifies the field on the HTML login form. This field is not applicable if the application uses BASIC authentication.</td>
</tr>
<tr>
<td>fieldValue</td>
<td>Optional. Login field value. Enter a default value for the login field or leave blank for a user to specify. This argument is blank by default.</td>
</tr>
<tr>
<td>displayToUser</td>
<td>Optional. Specifies whether the login field displays on the external application's login screen. Valid options are 1 (true) and 0 (false). This argument defaults to 0. Note that if you set this argument to 0, you must specify the fieldValue.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.8.4.3 Example

This example creates an additional field named Account with the default value username.default.example in an external application called ABC. This field will be displayed on ABC's login screen.

```
wls:/weblogic/serverConfig> addExtAppField(appName='webcenter', name='ABC', fieldName='Account', fieldValue='username.default.example', displayToUser=1)
```

10.8.5 setExtAppField

Module: Oracle WebCenter Portal
Use with WLST: Online

10.8.5.1 Description

Modifies the field value and display-to-user setting for one or more login fields currently configured for an external application. Either fieldValue or displayToUser must be specified along with the external application name and login field name. The fieldValue and displayToUser arguments are optional.

Using this command has implications on any shared or public credentials that you might have created for this external application. If you modify displayToUser to 1, you may also need to update existing shared user or public user credentials. See also Section 10.8.8, "setExtAppCredential".
10.8.5.2 Syntax

```
setExtAppField(appName, name, fieldName, [fieldValue, displayToUser, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing external application connection.</td>
</tr>
<tr>
<td>fieldName</td>
<td>Name of an existing login field.</td>
</tr>
<tr>
<td>fieldValue</td>
<td>Optional. New or changed login field value.</td>
</tr>
<tr>
<td>displayToUser</td>
<td>Optional. Specifies whether the login field displays on the external</td>
</tr>
<tr>
<td></td>
<td>application's login screen. Valid options are 1 (true) and 0 (false).</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if</td>
</tr>
<tr>
<td></td>
<td>more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.8.5.3 Example

The following example specifies a default value for a login field named Account and displays the field on the external application's credential provisioning screen:

```
wls:/weblogic/serverConfig> setExtAppField(appName='webcenter', name='ABC', fieldName='Account', fieldValue='admin', displayToUser=1)
```

10.8.6 removeExtAppField

Module: Oracle WebCenter Portal

Use with WLST: Online

10.8.6.1 Description

Removes a login field from an external application connection.

This command has implications on any shared or public credentials that you may have created for this external application, that is, you may need to remove the login field from shared user or public user credentials.

You can use the `setExtAppCredential` command to remove a login field, if required. For example, external application `myApp` has an additional field called `Account` and public credentials were previously specified using:

```
addExtAppCredential(appName='webcenter', name='myApp', type='PUBLIC', username='admin', password='mypublic.password', field='Account:admin@myhost.com')
```

If you remove the `Account` field, you can modify the credentials by running:

```
setExtAppCredential(appName='webcenter', name='myApp', type='PUBLIC', username='admin', password='mypublic.password')
```
For details on using setExtAppCredential, see Section 10.8.8, "setExtAppCredential"

10.8.6.2 Syntax
removeExtAppField(appName, name, fieldName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Connection name.</td>
</tr>
<tr>
<td>fieldName</td>
<td>Login field that you want to remove.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.8.6.3 Example
The following example removes the additional login field named Account from an external application named ABC.

```
wls:/weblogic/serverConfig> removeExtAppField(appName='webcenter', name='ABC', fieldName='Account')
```

10.8.7 addExtAppCredential
Module: Oracle WebCenter Portal
Use with WLST: Online

10.8.7.1 Description
Configures shared user or public user credentials for a specific external application.

When shared credentials are specified, every user accessing WebCenter Portal (or a Portal Framework application), is authenticated using the user name and password defined here. WebCenter Portal users are not presented with a login form.

Public users accessing this external application through WebCenter Portal (or the Portal Framework application) are logged in using the public credentials defined here.

If credentials already exists, a warning indicates that the setExtAppCredential command should be used instead.

10.8.7.2 Syntax
addExtAppCredential(appName, name, type, username, password, [field, server, applicationVersion])
The following example specifies public credentials for an external application named ABC. The public user name is mypublic.username, the password is mypublic.password, and there is one additional field named Account.

```wls/weblogic/serverConfig> addExtAppCredential(appName='webcenter', name='ABC', type='PUBLIC', username='mypublic.username', password='mypublic.password', field='Account:username.example')```

### 10.8.8 setExtAppCredential

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.8.8.1 Description

Modifies shared user or public user credentials currently configured for an external application. If the credential has already not been specified, then a warning indicates that `addExtAppCredential` needs to be used instead. See Section 10.8.7, "addExtAppCredential".

The arguments `username` and `password` are optional because `setExtAppCredential` only manipulates existing credentials. At least one of the parameters, `username`, `password` or `field`, must be specified.

You can use `setExtAppCredential` command to update passwords in systems that require changing passwords every few days.

#### 10.8.8.2 Syntax

```setExtAppCredential(appName, name, type, [username, password, field, server, applicationVersion])```
### External Applications

#### WebCenter Portal Custom WLST Commands

#### 10.8.8.3 Example

The following example changes the public user's login credentials for an external application named **ABC**.

```
wls:/weblogic/serverConfig> setExtAppCredential(appName='webcenter', name='ABC',
        type='PUBLIC', username='username.example', password='password.example',
        field='Account:username.example')
```

#### 10.8.9 removeExtAppCredential

Module: Oracle WebCenter Portal

Use with WLST: Online

**10.8.9.1 Description**

Removes shared user or public user credentials currently configured for an external application.

If credentials do not exist, an error displays.

**10.8.9.2 Syntax**

```
removeExtAppCredential(appName, name, type, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. winword For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing external application connection.</td>
</tr>
<tr>
<td>type</td>
<td>Credential type. Valid values are <strong>shared</strong> and <strong>public</strong>.</td>
</tr>
<tr>
<td>username</td>
<td>Optional. User name of the shared or public user.</td>
</tr>
<tr>
<td>password</td>
<td>Optional. Password for the shared or public user.</td>
</tr>
<tr>
<td>field</td>
<td>Optional. Additional login field value. Use the format <strong>FieldName:FieldValue</strong>, where <strong>FieldName</strong> names an additional login field configured with <strong>displayToUser=1</strong>.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
Example

The following example removes shared credentials specified for an external application named ABC.

```
wlsc:weblogic/serverConfig> removeExtAppCredential(appName='webcenter', name='ABC', type='SHARED')
```

### 10.9 Instant Messaging and Presence

Use the commands listed in Table 10–13, to manage instant messaging and presence server connections.

Configuration changes made using these WLST commands are only effective after your restart the Managed Server on which WebCenter Portal or your Portal Framework application is deployed. For details, see Oracle Fusion Middleware Administering Oracle WebCenter Portal.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>createIMPConnection</code></td>
<td>Create a new instant messaging and presence server connection for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td><code>setIMPConnection</code></td>
<td>Edit an existing instant messaging and presence server connection.</td>
<td>Online</td>
</tr>
<tr>
<td><code>setIMPConnectionProperty</code></td>
<td>Modify instant messaging and presence server connection properties.</td>
<td>Online</td>
</tr>
<tr>
<td><code>deleteIMPConnectionProperty</code></td>
<td>Delete an instant messaging and presence server connection property.</td>
<td>Online</td>
</tr>
<tr>
<td><code>listIMPAdapters</code></td>
<td>List which presence servers a named application supports.</td>
<td>Online</td>
</tr>
<tr>
<td><code>listIMPConnections</code></td>
<td>List all of the instant messaging and presence server connections that are configured for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td><code>listDefaultIMPConnection</code></td>
<td>List the default instant messaging and presence server connection that is configured for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td><code>setDefaultIMPConnection</code></td>
<td>Set a specified connection as the default instant messaging and presence server connection.</td>
<td>Online</td>
</tr>
<tr>
<td><code>setIMPServiceProperty</code></td>
<td>Specify defaults for instant messaging and presence.</td>
<td>Online</td>
</tr>
<tr>
<td><code>removeIMPServiceProperty</code></td>
<td>Remove defaults for instant messaging and presence.</td>
<td>Online</td>
</tr>
<tr>
<td><code>listIMPServiceProperties</code></td>
<td>List instant messaging and presence properties.</td>
<td>Online</td>
</tr>
</tbody>
</table>
10.9.1 createIMPConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

10.9.1.1 Description

Creates an instant messaging and presence server connection for a named application.

Use the listIMPAdapters command to find out which types of instant messaging and presence servers are supported. Out-of-the-box, WebCenter Portal and Portal Framework applications support Microsoft Office Live Communications Server 2005 R2 (LCS), and Microsoft Office Communications Server 2007 SP1 (OCS), and Microsoft Lync 2010.

While you can register multiple presence server connections for WebCenter Portal or your own Portal Framework applications, only one connection is used for instant messaging and presence services—the default (or active) connection.

10.9.1.2 Syntax

createIMPConnection(appName, name, adapter, url, [appId, poolName, userDomain, timeout, default, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Connection name. The name must be unique (across all connection types) within the application.</td>
</tr>
<tr>
<td>adapter</td>
<td>Adapter name. Specify the adapter that matches your instant messaging and presence server. Valid values are LCS and OCS. Choose LCS for Microsoft Live Communications Server 2005. Choose OCS2007 for Microsoft Office Communications Server 2007 and Microsoft Lync.</td>
</tr>
<tr>
<td>url</td>
<td>URL of the sever hosting instant messaging and presence services. For example: <a href="http://myocshost.com:8888">http://myocshost.com:8888</a></td>
</tr>
<tr>
<td>domain</td>
<td>Deprecated. Use the setIMPServiceProperty command to resolve IM addresses.</td>
</tr>
</tbody>
</table>
10.9.1.3 Examples

The following example creates an external application suitable for an instant messaging and presence server connection and then creates a connection named myLCSPresenceServer to a Microsoft Live Communications Server:

```wls
wls:/weblogic/serverConfig> createIMPExtApp(appName='webcenter', name='LCSExtApp', displayName='IMP Ext App')
```

```wls
wls:/weblogic/serverConfig> createIMPConnection(appName='webcenter', name='myLCSPresenceServer', adapter='LCS', url='http://mylcs/host.com/owc/lcs', appId='LCSExtApp', poolName='pool1.myhost.com', timeout=60, default=1)
```

The following example creates an instant messaging and presence server connection to a Microsoft Office Communications Server named myOCSPresenceServer:

```wls
wls:/weblogic/serverConfig> createIMPConnection(appName='webcenter', name='myOCSPresenceServer', adapter='OCS2007', url='http://myocs/host.com/owc/ocs', userDomain='Contoso.com')
```
The following example creates an instant messaging and presence server connection to a Microsoft Lync Server named myLyncServer.

```
wls:/weblogic/serverConfig> createIMPConnection(appName='webcenter',
        name='myLyncServer', adapter='OCS2007',
        url='http://mylynchost.com:8888' appId='LyncExtApp', userDomain='LYNC',
        poolName='pool05.mylynchost.com', timeout=60, default=1))
```

### 10.9.2 setIMPConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

**10.9.2.1 Description**

Edits an existing instant messaging and presence server connection. Use this command to update connection attributes.

The connection is created using the `createIMPConnection` command.

**10.9.2.2 Syntax**

```
setIMPConnection(appName, name, [adapter, url, appId, poolName,
       userDomain, timeout, default, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
</table>
| appName  | Name of the application in which to perform this operation.  
For WebCenter Portal, the application name is always `webcenter`.  
For Portal Framework applications, specify the appropriate name. |
| name     | Name of an existing presence server connection. |
| adapter  | Optional. Adapter name. Specify the adapter that matches your instant messaging and presence server. Valid values are `LCS` and `OCS2007`.  
Choose `LCS` for Microsoft Live Communications Server.  
Choose `OCS2007` for Microsoft Office Communications Server and Microsoft Lync Server. |
| url      | Optional. URL of the server hosting instant messaging and presence services. |
| domain   | Deprecated.  
Use the `setIMPServiceProperty` command to resolve IM addresses. |


### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appId</td>
<td>Optional. External application associated with the presence server connection. If specified, external application credential information is used to authenticate users against the LCS, OCS, or Lync server. This argument is mandatory for LCS, OCS and Lync server connections. The external application you configure for instant messaging and presence services must use <code>authMethod=POST</code>, and specify an additional field with <code>fieldName='Account'</code> and <code>displayToUser=1</code>. If an external application does not exist yet, use the WLST command <code>createIMPExtAppConnection</code> to create an external application that automatically has all the required additional fields. See also <code>addExtAppField</code> and <code>setExtAppField</code>.</td>
</tr>
<tr>
<td>poolName</td>
<td>Optional. (LCS, OCS, and Lync) Pool name that is required to create an LCS, OCS, or Lync connection. Refer to Microsoft Live Communications Server, Microsoft Office Communications Server or Microsoft Lync Server documentation for details on pool names. This argument is mandatory for LCS, OCS, and Lync server connections.</td>
</tr>
<tr>
<td>userDomain</td>
<td>Optional. (OCS and Lync only.) Active Directory domain on the OCS server. This argument is mandatory for OCS/Lync server connections.</td>
</tr>
<tr>
<td>timeout</td>
<td>Optional. Length of time (in seconds) that instant messaging and presence waits for a response from the presence server before issuing a connection timeout message. This argument defaults to <code>-1</code>. When set to <code>-1</code>, the service default (10 seconds) applies.</td>
</tr>
</tbody>
</table>
| default        | Optional. Indicates whether this connection is the default connection for instant messaging and presence services. Valid values are `1` (true) and `0` (false). The default for this argument is `0`. To specify that instant messaging and presence uses this connection, change the value from `0` to `1`. To disable this connection, use the `removeIMPServiceProperty` command:  
`removeIMPServiceProperty('appName='webcenter', property='selected.connection')`  
While you can register multiple presence server connections for an application, only one connection is used for instant messaging and presence services—the default (or active) connection. |
| server         | Optional. Name of the managed server where the application is deployed. For example, `WC_Spaces`. Required when applications with the same name are deployed to different servers and also when you have a cluster. |
| applicationVersion | Optional. Version number of the deployed application. Required if more than one version of the application is deployed. |

### 10.9.2.3 Examples

The following example sets attributes on an existing instant messaging and presence server connection.

```
wlst:/weblogic/serverConfig> setIMPConnection(appName='webcenter', name='myOCSPresenceServer', adapter='OCS2007', url='http://myocshost.com/owc/ocs', timeout=120, default=1)
```
The following example sets attributes on an existing instant messaging and presence server connection.

```
wlst:/weblogic/serverConfig>setIMPConnection(appName='webcenter',
    name='myLCSPresenceServer', adapter='LCS', url='http://mylcshost.com/owc/lcs',
    appId='LCSExtApp', poolName='pool3.myhost.com', default=0)
```

### 10.9.3 setIMPConnectionProperty

**Module:** Oracle WebCenter Portal

**Use with WLST:** Online

#### 10.9.3.1 Description

Sets an instant messaging and presence server connection property. Use this command if additional parameters are required to connect to your presence server. This is an extensible way to add any connection property using a key and a value. (You are not limited to connection properties specified by `createIMPConnection` and `setIMPConnection`.)

All known, additional connection properties are listed in Table 10–14, "Additional Instant Messaging and Presence Connection Properties".

#### Table 10–14 Additional Instant Messaging and Presence Connection Properties

<table>
<thead>
<tr>
<th>Additional Connection Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>presence.url</code></td>
<td>URL to the presence server. This must be supplied if presence is deployed on a separate server.</td>
</tr>
<tr>
<td><code>contacts.url</code></td>
<td>URL to the contact management service. This must be supplied if the contact management service is deployed on a separate server.</td>
</tr>
<tr>
<td><code>call.url</code></td>
<td>URL for the third-party call server. If no value is supplied, then this uses the same value as <code>base.connection.url</code>.</td>
</tr>
</tbody>
</table>
| `call.method`                 | Supports values `sip` and `pstn`:

  - When set to `sip`, the IMP service forwards the user’s SIP address to the third-party call service. The third-party call service must decide on the routing of the call.
  - When set to `pstn`, the user’s phone number is based on the user’s profile attribute (BUSINESS_PHONE). You can use the connection property `call.number.attribute` to change this default profile attribute (BUSINESS_PHONE) to any other attribute. |

| `call.domain`                 | The domain name of the `pstn` gateway. If no domain name is supplied, then this uses the domain value specified when the connection was created. Supply a domain name only when `call.method` is set to `pstn`. |
| `contact.number.attribute`   | The attribute used to read users’ phone numbers from the user profile. The default is BUSINESS_PHONE. Supply this attribute value only when `call.method` is set to `pstn`. |
10.9.3.2 Syntax

```
setIMPConnectionProperty(appName, name, key, value, [secure, server, applicationVersion])
```

### Argument | Definition
--- | ---
appName | Name of the application in which to perform this operation. For WebCenter Portal, the application name is always `webcenter`. For Portal Framework applications, specify the appropriate name.
name | Name of an existing presence server connection.
key | Name of the connection property. See Table 10–14, "Additional Instant Messaging and Presence Connection Properties".
value | Value for the property. Allows any property to be modified on the connection with a key and value.
secure | Optional. Indicates whether the property value must be stored securely using encryption. Valid options are 1 (true) and 0 (false). When 1, the value is encrypted. The default option is 0.
server | Optional. Name of the managed server where the application is deployed. For example, `WC_Spaces`. Required when applications with the same name are deployed to different servers and also when you have a cluster.
applicationVersion | Optional. Version number of the deployed application. Required if more than one version of the application is deployed.

### Example

The following example adds a custom instant messaging and presence server connection property called `admin.user` with a default value `admin`:

```
wls:/weblogic/serverConfig> setIMPConnectionProperty(appName='webcenter', name='MyLCSPresenceServer', key='admin.user', value='admin')
```

### deleteIMPConnectionProperty

Module: Oracle WebCenter Portal

Use with WLST: Online

Note: Do not use the `setIMPConnectionProperty` to set connection properties available through `createIMPConnection` or `setIMPConnection`. Attempting to do so has no effect.
10.9.4.1 Description
Deletes an instant messaging and presence server connection property. Use caution when deleting connection properties because the connection might not work as expected if the configuration becomes invalid as a result.

This command can only delete additional connection properties added using the setIMPConnectionProperty command.

10.9.4.2 Syntax
deleteIMPConnectionProperty(appName, name, key, [server, applicationVersion])

10.9.4.3 Example
The following example deletes an instant messaging and presence server connection property named admin.user.

wls:/weblogic/serverConfig>deleteIMPConnectionProperty(appName='webcenter', name='MyLCSPresenceServer', key='admin.user')

10.9.5 listIMPAdapters
Module: Oracle WebCenter Portal
Use with WLST: Online

10.9.5.1 Description

10.9.5.2 Syntax
listIMPAdapters()

10.9.5.3 Example
The following example lists which presence servers are supported:

wls:/weblogic/serverConfig>listIMPAdapters()}
10.9.6 listIMPConnections

Module: Oracle WebCenter Portal

Use with WLST: Online

10.9.6.1 Description

Lists all of the instant messaging and presence server connections that are configured for a named application.

10.9.6.2 Syntax

listIMPConnections(appName,[verbose, name, server, applicationVersion])

10.9.6.3 Examples

The following example lists all the instant messaging and presence server connections that are configured for WebCenter Portal (webcenter):

```
wls:/weblogic/serverConfig>listIMPConnections(appName='webcenter')
```

The following example lists all the instant messaging and presence server connections that are configured for WebCenter Portal in verbose mode:

```
wls:/weblogic/serverConfig>listIMPConnections(appName='webcenter', verbose=1)
```

The following example lists connection details for an instant messaging and presence server connections named impConnection1.

```
wls:/weblogic/serverConfig> listIMPConnections(appName='webcenter', name='impConnection1')
```
10.9.7 listDefaultIMPConnection

Module: Oracle WebCenter Portal
Use with WLST: Online

10.9.7.1 Description
Lists the connection that instant messaging and presence is using, in a named application. While you can register multiple presence server connections for WebCenter Portal or your own Portal Framework applications, instant messaging and presence only uses one connection—the default (or active) connection.

If only one presence server connection is available, that connection is assumed to be the default connection.

10.9.7.2 Syntax
listDefaultIMPConnection(appName, verbose, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Displays the default presence server connection in verbose mode, if available. Valid options are 1 (true) and 0 (false). When set to 1, the name and details of the presence server connection are listed. When set to 0, only the connection name displays. This argument defaults to 0.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.9.7.3 Example
The following example lists the name and details of the connection that instant messaging and presence is using in WebCenter Portal (webcenter):

wls:/weblogic/serverConfig>listDefaultIMPConnection(appName='webcenter', verbose=1)

10.9.8 setDefaultIMPConnection

Module: Oracle WebCenter Portal
Use with WLST: Online

10.9.8.1 Description
Specifies the default connection for instant messaging and presence in a named application. While you can register multiple presence server connections with an application, instant messaging and presence only uses one connection—the default (or active) connection.
If only one presence server connection is available, that connection is assumed to be the default connection.

10.9.8.2 Syntax

```java
setDefaultIMPConnection(appName, name, [server, applicationVersion])
```

10.9.8.3 Example

The following example makes a connection named `myPresenceServer` the default (or active) connection for instant messaging and presence in WebCenter Portal (`webcenter`):

```
wls:/weblogic/serverConfig>setDefaultIMPConnection(appName='webcenter',
name='myPresenceServer')
```

10.9 setIMPServiceProperty

Module: Oracle WebCenter Portal

Use with WLST: Online

10.9.9.1 Description

Specifies default values for instant messaging and presence.

Configurable properties for instant messaging and presence are listed in Table 10–15, "Instant Messaging and Presence - Configurable Properties".

<table>
<thead>
<tr>
<th>Configuration Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>selected.connection</code></td>
<td>Connection used by instant messaging and presence.</td>
</tr>
<tr>
<td><code>rtc.cache.time</code></td>
<td>Cache timeout for instant messaging and presence data. The default is 60 seconds.</td>
</tr>
</tbody>
</table>
### Table 10–15 (Cont.) Instant Messaging and Presence - Configurable Properties

<table>
<thead>
<tr>
<th>Configuration Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resolve.display.name.from.user.profile</td>
<td>Determines what to display if user display names are missing. When set to 0, and display name information is unavailable, only the user name displays in the application. When set to 1, and display name information is unavailable, display names are read from user profile data. Setting this option to 1 will impact performance. The default setting is 0. Display names are not mandatory in presence data. If WebCenter Portal or the Portal Framework application does not always provide display names by default and you consider this information important, set resolve.display.name.from.user.profile to 1 so that display names always display.</td>
</tr>
</tbody>
</table>
| im.address.resolver.class                      | Resolver implementation used to map user names to IM addresses and IM addresses to user names. The default setting is oracle.webcenter.collab.rtc.IMAddressResolverImpl. This implementation looks for IM addresses in the following places and in the order specified:  
■ User Preferences  
■ User Credentials  
■ User Profiles |
| im.address.profile.attribute                   | User profile attribute used to determine a user's IM address. The default setting is BUSINESS_EMAIL.                                                                                                       |

### 10.9.9.2 Syntax

`setIMPServiceProperty(appName, property, value, [server, applicationVersion])`

### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>property</td>
<td>Name of the configuration property.</td>
</tr>
<tr>
<td>value</td>
<td>Value for the property.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

### 10.9.9.3 Example

The following example changes the default cache timeout for instant messaging and presence data in WebCenter Portal (`webcenter`):

```wls:/weblogic/serverConfig> setIMPServiceProperty(appName='webcenter', property='rtc.cache.time', value='30')```
10.9.10 removeIMPServiceProperty

Module: Oracle WebCenter Portal

Use with WLST: Online

10.9.10.1 Description

Removes the current value that is set for an instant messaging and presence property. Use this command to remove any of the properties listed in Table 10–15, "Instant Messaging and Presence - Configurable Properties".

Take care when using this command as removing values for these properties might cause unexpected behavior.

**Note:** Use this command syntax to disable the connection currently used by instant messaging and presence:

```
removeIMPServiceProperty('appName='webcenter',
property='selected.connection')
```

This command forces the default connection argument to 0. See also, setIMPConnection.

10.9.10.2 Syntax

```
removeIMPServiceProperty(appName, property, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>property</td>
<td>Name of the configuration property.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different</td>
</tr>
<tr>
<td></td>
<td>servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more</td>
</tr>
<tr>
<td></td>
<td>than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.9.10.3 Example

The following example clears the cache expiration value for instant messaging and presence in WebCenter Portal:

```
wls:/weblogic/serverConfig>removeIMPServiceProperty(appName='webcenter',
property='rtc.cache.time')
```

10.9.11 listIMPServiceProperties

Module: Oracle WebCenter Portal

Use with WLST: Online

10.9.11.1 Description

Lists all configurable properties for instant messaging and presence.
10.9.11.2 Syntax

```
listIMPServiceProperties(appName, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.9.11.3 Example

The following example lists configuration properties for instant messaging and presence in WebCenter Portal (`webcenter`):

```
wls:/weblogic/serverConfig>listIMPServiceProperties(appName='webcenter')
```

10.9.12 createIMPExtAppConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

10.9.12.1 Description

Creates an external application suitable for instant messaging and presence server connections. The external application is configured with the required additional properties: `authMethod=POST`, and additional fields `fieldName='Account'` and `displaytoUser=1`.

10.9.12.2 Syntax

```
createIMPExtAppConnection(appName, name, [displayName, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Connection name. The name must be unique (across all connection types) within the application.</td>
</tr>
<tr>
<td>displayName</td>
<td>Optional. External application display name. A user friendly name for the application that users will recognize. The display name must be unique across all external applications within the application.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
</tbody>
</table>
Example

The following example creates an external application named IMPxApp suitable for instant messaging and presence server connections:

```
createIMPExtAppConnection(appName='webcenter', name='IMPxApp', displayName='IMP Ext App')
```

10.10 Identity Store

Use the commands listed in Table 10–16 to configure options for searching an application’s identity store.

### Table 10–16 WebCenter Portal Identity Store WLST Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>setWebCenterIdStoreSearchConfig</td>
<td>Modify configuration options for searching a named applications's identity store.</td>
<td>Online</td>
</tr>
<tr>
<td>listWebCenterIdStoreSearchConfig</td>
<td>List current configuration options for searching a named application's identity store.</td>
<td>Online</td>
</tr>
</tbody>
</table>

10.10.1 setWebCenterIdStoreSearchConfig

Module: Oracle WebCenter Portal

Use with WLST: Online

**Description**

Modifies configuration options for searching a named application’s identity store. Use these settings to optimize identity store searches (for users and roles) in WebCenter Portal or a Portal Framework application.

Identity store search parameters are stored in `adf-config.xml`. If a search parameter is not specified, it is not modified.

**Syntax**

```
setWebCenterIdStoreSearchConfig(appName, [narrowSearchTimeout, broadSearchTimeout, maxSearchFilters, maxFetchRecords, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>narrowSearchTimeout</td>
<td>Optional. Maximum time allowed (in ms) for small, simple searches, such as fetching a single user from the identity store.</td>
</tr>
<tr>
<td></td>
<td>Out-of-the-box, the default is 30000ms.</td>
</tr>
</tbody>
</table>
The following example increases both identity store search timeouts.

```
wls:/weblogic/serverConfig>setWebCenterIdStoreSearchConfig(appName='webcenter',
narrowSearchTimeout=60000, broadSearchTimeout=100000);
```

The following example limits the maximum number of records returned to 100.

```
wls:/weblogic/serverConfig>setWebCenterIdStoreSearchConfig(appName='webcenter',
maxFetchRecords=100);
```

### listWebCenterIdStoreSearchConfig

**Module:** Oracle WebCenter Portal  
**Use with WLST:** Online

#### 10.10.2.1 Description

Lists current configuration options for searching the identity store for a named application (WebCenter Portal or a Portal Framework application).

Identity store search parameters are stored in `adf-config.xml`.

#### 10.10.2.2 Syntax

```
listWebCenterIdStoreSearchConfig(appName, [server, applicationVersion])
```
### 10.10.2.3 Example

The following example displays identity store search configuration information for WebCenter Portal (webcenter).

```
wlsc:/weblogic/serverConfig> listWebCenterIdStoreSearchConfig(appName='webcenter');
```

User role search configuration parameters
-----------------------------
Narrow search timeout : 30000
Broad search timeout : 60000
Maximum search filters : 100
Maximum records to fetch : 200

### 10.11 Lifecycle

Use the commands listed in Table 10–17 to perform lifecycle operations for WebCenter Portal and Portal Framework applications.

**Table 10–17  Lifecycle WLST Commands**

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>deployWebCenterPortal</td>
<td>(WebCenter Portal only) Deploy a portal from a stage environment to a production environment.</td>
<td>Online</td>
</tr>
<tr>
<td>propagateWebCenterPortal</td>
<td>(WebCenter Portal only) Propagate metadata for a named portal, from a stage environment to a production environment.</td>
<td>Online</td>
</tr>
<tr>
<td>exportWebCenterPortals</td>
<td>(WebCenter Portal only) Export one or more named portals to a portal archive (.par file).</td>
<td>Online</td>
</tr>
<tr>
<td>exportWebCenterPortalTemplates</td>
<td>(WebCenter Portal only) Export one or more named portal templates to a portal archive (.par file).</td>
<td>Online</td>
</tr>
<tr>
<td>importWebCenterPortals</td>
<td>(WebCenter Portal only) Import one or more portals or portal templates from a portal archive (.par file).</td>
<td>Online</td>
</tr>
</tbody>
</table>
### Table 10–17 (Cont.) Lifecycle WLST Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>listWebCenterPortalArchive</td>
<td>(WebCenter Portal only) List the content of a portal archive and extract the portal archive to a specified location</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>exportWebCenterPortalChanges</td>
<td>(WebCenter Portal only) Export metadata changes for a portal to a portal archive (par file).</td>
<td>Online</td>
</tr>
<tr>
<td>importWebCenterPortalChanges</td>
<td>(WebCenter Portal only) Import metadata changes for a portal from a portal archive (par file).</td>
<td>Online</td>
</tr>
<tr>
<td>exportWebCenterPortalConnections</td>
<td>(WebCenter Portal only) Export connection configuration information from a source WebCenter Portal environment to a named file.</td>
<td>Online</td>
</tr>
<tr>
<td>importWebCenterPortalConnections</td>
<td>(WebCenter Portal only) Import new WebCenter Portal connections from a named connection properties file.</td>
<td>Online</td>
</tr>
<tr>
<td>setSpaceState</td>
<td>(WebCenter Portal only) Take a portal offline or bring a portal online.</td>
<td>Online</td>
</tr>
<tr>
<td>exportWebCenterResource</td>
<td>Export a portal resource to an export archive (ear file).</td>
<td>Online</td>
</tr>
<tr>
<td>importWebCenterResource</td>
<td>Import a portal resource from an export archive (ear file)</td>
<td>Online</td>
</tr>
<tr>
<td>importWebCenterTranslations</td>
<td>Import translations for WebCenter Portal.</td>
<td>Online</td>
</tr>
<tr>
<td>exportWebCenterApplication</td>
<td>Export the WebCenter Portal application to an export archive (ear file).</td>
<td>Online</td>
</tr>
<tr>
<td>importWebCenterApplication</td>
<td>Import the WebCenter Portal application from an export archive (ear file).</td>
<td>Online</td>
</tr>
<tr>
<td>exportPortletClientMetadata</td>
<td>Export portlet client metadata and producer customizations and personalizations to an export archive.</td>
<td>Online</td>
</tr>
<tr>
<td>importPortletClientMetadata</td>
<td>Import portlet client metadata and producer customizations and personalizations from an export archive.</td>
<td>Online</td>
</tr>
<tr>
<td>showProducerImportFailures</td>
<td>Display names of producers where metadata imports have failed and reasons for those failures.</td>
<td>Online</td>
</tr>
<tr>
<td>retryAllFailedProducerImports</td>
<td>Attempt to import outstanding producer metadata</td>
<td>Online</td>
</tr>
<tr>
<td>cloneWebCenterManagedServer</td>
<td>Clone a Managed Server used by WebCenter Portal or a Portal Framework application.</td>
<td>Online</td>
</tr>
</tbody>
</table>

### Table 10–18 Life Cycle WLST Commands Deprecated in 11.1.1.8.0

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>exportGroupSpaces (Deprecated)</td>
<td>Export one or more spaces to an export archive.</td>
<td>Online</td>
</tr>
</tbody>
</table>
10.11.1 deployWebCenterPortal

Module: Oracle WebCenter Portal

Use with WLST: Online

10.11.1.1 Description

(WebCenter Portal only) Deploys a portal from a stage environment to a production environment.

Notes:
- The name of the managed server must be the same in both the stage and production environments.
- A connection to the production environment must be defined on the stage instance.
- You must have at least the WebLogic Monitor role and the WebCenter Portal permission Portals - Manage All.

10.11.1.2 Syntax

```
deployWebCenterPortal(appName, portalName, targetConnectionName,
                       deployCustomizations, deployPortalContent, deploySecurity, deployData,
                       deployActivities, overwrite, savePortal, deployLog, server, applicationVersion)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>portalName</td>
<td>Name of the portal that you want to deploy. For example, portalName='myPortal'. Note: Do not enter the portal's display name here. You must enter the portal name that appears in the portal URL. If you are not sure, obtain the portal name from the About Portal dialog.</td>
</tr>
<tr>
<td>targetConnectionName</td>
<td>Name of a connection on the source stage instance that specifies how to connect to the target production instance. Note: Use Enterprise Manager or the WLST command adf_CREATEDHTTPURLCONNECTION to configure the connection if it does not exist.</td>
</tr>
</tbody>
</table>
**Lifecycle**

### WebCenter Portal Custom WLST Commands

#### deployCustomizations
Optional. Indicates whether to deploy customizations associated with the portal on the target. Valid values are 1 and 0.
- 1 - Deploy portal customizations. This includes:
  - **Portal-level customizations** (to system pages, assets, task flows, and so on)
  - **User-level customizations** (sometimes referred to as user personalizations)

  deployCustomizations does not completely overwrite existing portal customizations on the target (if any). Only customizations included in the source portal are overwritten on deployment, leaving any other customizations on the target unchanged.
- 0 - Exclude portal customizations, for example, deploy default task flows without any customizations and the default portal settings. If you are redeploying a portal, all existing customizations on the target are preserved.

This argument defaults to 1.

#### deployPortalContent
Optional. Specifies whether to deploy the portal's content folder on the target. Valid values are 1 and 0.
- 1 - Deploys the portal's content folder.
- 0 - Excludes portal content.

This argument defaults to 0.

#### deploySecurity
Optional. Indicates whether to import portal member details on the target. Valid values are 1 and 0.
- 1 - Roles and permissions for the portal, as well as member details and their role assignments, are deployed on the target.
- 0 - Only portal roles and permissions are deployed. This option is useful when migrating between stage and production environments and where member details, added during the testing phase, are no longer required.

This argument defaults to 1.

Note: Always set deploySecurity=1 when importing a brand new portal as you cannot import a new portal without a security policy.

#### deployData
Optional. Indicates whether to deploy portal data on the target. Valid values are 1 and 0.
- 1 - Portal-related data stored in the WebCenter Portal database for activity streams, events, feedback, lists, links, message boards, people connections, polls, profiles, polls, and surveys is deployed.
- 0 - Portal data is not deployed and data on the target (if any) is preserved. This option is useful when migrating between stage and production environments and where test data is no longer required.

This argument defaults to 1.
The following example deploys a portal named myPortal without its content folder but includes the portal’s customizations, security, and data:

```
wlsh:weblogic/serverConfig> deployWebCenterPortal(appName='webcenter',
portName='myPortal', targetConnectionName='MyProductionConnection')
```

### 10.11.1.3 Examples

The following example deploys a portal named myPortal without its content folder but includes the portal’s customizations, security, and data:

```
wls:/weblogic/serverConfig> deployWebCenterPortal(appName='webcenter',
portName='myPortal', targetConnectionName='MyProductionConnection')
```

**Note:** You can use the `adf_createHttpURLConnection` WLST command to create MyProductionConnection. For example:

```bash
adf_createHttpURLConnection(appName='webcenter',
name='MyProductionConnection',
url='http://example.com:7777', user='myuser',
password='mypass', realm='ProductionRealm')
```
The following example deploys a portal named `myPortal` with all its associated content, customizations, security, and data, and also specifies a name and location for the deploy log file:

```java
wls:/weblogic/serverConfig> deployWebCenterPortal(appName='webcenter', portalName='myPortal', targetConnectionName='MyProductionConnection', deployPortalContent=1, deployLog='/mydeploylogs/myPortal_deploy.log')
```

The following example backs up `myPortal` on the target before redeploying "myPortal" and all its associated content, customizations, security, and data on the source:

```java
wls:/weblogic/serverConfig> deployWebCenterPortal(appName='webcenter', portalName='myPortal', targetConnectionName='MyProductionConnection', deployPortalContent=1, savePortal=1, overwrite=1)
```

---

### 10.11.2 propagateWebCenterPortal

**Module:** Oracle WebCenter Portal  
**Use with WLST:** Online

#### 10.11.2.1 Description

(WebCenter Portal only) Propagates metadata changes for a named portal, from a stage environment to a production environment. Metadata changes propagated:

- **Includes:**
  - Portal-level customizations (metadata changes) for portal pages, system pages, portlets, assets, task flows
  - User-level customizations (metadata changes) for portal pages, portlets, task flow instances
- **Excludes:** subportals, security, any changes to content and data, portal state (online/offline)

---

**Important:** You can only propagate portals that were previously deployed (from stage to production) using the `deployWebCenterPortal` command.

---

To migrate other changes, consider using `exportWebCenterPortals` and `importWebCenterPortals` or `deployWebCenterPortal`.

**Notes:**

- The name of the managed server must be the same in both the stage and production environments.
- A connection to the production environment must be defined on the stage instance.
- You must have at least the WebLogic Monitor role and the WebCenter Portal permission `Portals - Manage All`.

#### 10.11.2.2 Syntax

```java
propagateWebCenterPortal(appName, portalName, targetConnectionName, [savePortal, propagateLog, server, applicationVersion])
```
### Argument | Definition
--- | ---
app\_Name | Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter.
portal\_Name | Name of the portal that you want to deploy. For example, portal\_Name='myPortal'.
Note: Do not enter the portal’s display name here. You must enter the portal name that appears in the portal URL. If you are not sure, obtain the portal name from the About Portal dialog.
target\_Connection\_Name | Name of a connection on the source stage instance that specifies how to connect to the target production instance.
Note: Use Enterprise Manager or the WLST command adf\_createHttpUrlConnection to configure the connection if it does not exist.
save\_Portal | Optional. Specifies whether to back up the portal on the target instance to an archive before propagating changes. Valid values are 1 and 0:
- 1 - Backs up the target portal to an archive before propagation. If created, the backup archive is saved to the /tmp directory on the target machine.
- 0 - Do not back up the target portal before propagating changes.
The default value is 0.
propagate\_Log | Optional. Name and location of a local log file containing detailed information about the propagation operation.
If not specified, a propagation log file named PortalPropagation_\_<timestamp>.log is generated in the temporary directory.
server | Optional. Name of the managed server where WebCenter Portal is deployed. For example, WC_Spaces.
Required when applications with the same name are deployed to different servers and also when you have a cluster.
application\_Version | Optional. Version number of the deployed application. Required if more than one version of the WebCenter Portal application is deployed.

### 10.11.2.3 Example
The following example propagates metadata changes for a portal named myPortal:

```bash
wls:/weblogic/serverConfig> propagateWebCenterPortal(appName='webcenter', portalName='myPortal', targetConnectionName='MyProductionConnection')
```

**Note:** You can use the adf\_createHttpUrlConnection WLST command to create MyProductionConnection. For example:

```bash
adf_createHttpUrlConnection(appName='webcenter', name='MyProductionConnection', url='http://example.com:7777', user='myuser', password='mypassword', realm='ProductionRealm')
```

The following example backs up a portal named myPortal and then propagates metadata changes for the portal:

```bash
wls:/weblogic/serverConfig> propagateWebCenterPortal(appName='webcenter', portalName='myPortal', savePortal=1, targetConnectionName='MyProductionConnection')
```
portalName='myPortal', targetConnectionName='MyProductionConnection', savePortal=1)

The following example propagates metadata changes for a portal named myPortal, and also specifies a name and location for the propagation log file:

```
wlst:/weblogic/serverConfig> propagateWebCenterPortal(appName='webcenter', portalName='myPortal', targetConnectionName='MyProductionConnection', propagateLog='/mypropagationlogs/myPortal_propagation.log')
```

### 10.11.3 exportWebCenterPortals

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.11.3.1 Description

(WebCenter Portal only) Exports one or more named portals to a portal archive (.par file), using the filename specified.

The portal archive file contains one or more portal data archives (.pdr files) -- one for each portal (or portal hierarchy) that you export. Each portal data archive includes portal-specific data, portal-level customizations, user-level customizations, security information, and optionally, the portal content folder.

The export operation continues if one of the portals fail to export properly. Status messages are shown in the WLST console indicating the success or failure of the export operation for each specified portal. Portals that could not be exported are excluded from the .par file.

**Note:** To run this command you must have at least the WebLogic Monitor role and the WebCenter Portal permission Portals - Manage All.

#### 10.11.3.2 Syntax

```
exportWebCenterPortals(appName, fileName, [names, offlineDuringExport, exportPortalContent, exportConnections, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>fileName</td>
<td>Name and location of the export archive (.par file). For example,</td>
</tr>
<tr>
<td></td>
<td>/myExports/myPortalExport.par. If you do not specify a location, the</td>
</tr>
<tr>
<td></td>
<td>archive is created in the same directory that you run the WLST command.</td>
</tr>
</tbody>
</table>
names

Optional. Name of one or more portals that you want to export. Separate multiple portal names with a comma. For example:

```
names='sales,finance'
```

Note: Do not enter the portal's display names here. You must enter the name that is specified in the portal URL. The portal name is available from the About Portal dialog.

If you leave the names argument blank, all the portals are exported (up to a maximum of ten portals). If your instance contains more than ten portals you must name each portal individually, using the names argument.

offlineDuringExport

Optional. Indicates whether the portals you want to export must be offline before starting the export process. If users are allowed to access the portal, any changes made to the portal once the export starts are not exported.

Valid values are 1 and 0.

- 1 - Takes online portals (if any) offline before starting the export process and at the end of the export process, returns those portals back to their original online state.
  
  To prevent data loss during the export process, Oracle recommends that you set offlineDuringExport=1.
  
- 0 - Exports the portals in their current state.

The default value is 0.

exportPortalContent

Optional. Specifies whether to export each portal's content folder on WebCenter Content Server.

A content folder is automatically created on Content Server for any portal using document services to create, manage, and store portal documents (files, folders, wikis, blogs). Only content that is stored in this folder can be exported with the portal. The export does not, for example, include web content or pages displayed through Content Presenter since this information is not stored in the portal's content folder.

Valid values are 1 and 0.

- 1 - Exports the portal’s content folder.
  
- 0 - Excludes portal content.

The default value is 0.

exportConnections

Optional. Specifies whether to export WebCenter Portal connections into a file named connection.properties and include it in the portal archive (.par file).

In addition, a copy of connection.properties is included outside the archive at the same location, that is, the location specified using the fileName option.

Valid values are 1 and 0.

- 1 - Export WebCenter Portal connections
  
- 0 - Do not export WebCenter Portal connections.

The default value is 0.

See also, "Understanding Connection Property Files" in Oracle Fusion Middleware Administering Oracle WebCenter Portal.

server

Optional. Name of the managed server where WebCenter Portal is deployed. For example, WC_Spaces.

Required when applications with the same name are deployed to different servers and also when you have a cluster.
WebCenter Portal Custom WLST Commands

10.11.3 Examples

The following example exports myPortal1 and myPortal2 to myPortalExport.par:

```
wls:/weblogic/serverConfig> exportWebCenterPortals(appName='webcenter', fileName='myPortalExport.par', names='myPortal1, myPortal2')
```

The following example exports myPortal1 to myPortalExport.par. To ensure that myPortal1 is offline while the export operation takes place, offlineDuringExport is set to 1:

```
wls:/weblogic/serverConfig> exportWebCenterPortals(appName='webcenter', fileName='myPortalExport.par', names='myPortal1', offlineDuringExport=1)
```

The following example exports all the portals, including portal content folders, to exportAllMyPortals.par:

```
wls:/weblogic/serverConfig> exportWebCenterPortals(appName='webcenter', fileName='exportAllMyPortals.par', exportPortalContent=1)
```

10.11.4 exportWebCenterPortalTemplates

Module: Oracle WebCenter Portal

Use with WLST: Online

10.11.4.1 Description

(WebCenter Portal only) Exports one or more named portal templates to a portal archive (.par file), using the filename specified. The portal archive file contains a portal data archive (.pdr file) for each portal template you export.

Each portal data archive includes template-specific data (pages, discussions, and lists), customizations, security information, and optionally, portal template content.

The export operation continues if one of the portal templates fails to export properly. Status messages are shown in the WLST console indicating the success or failure of the export operation for each specified portal. Portal templates that could not be exported are excluded from the .par file.

Note: To run this command you must have at least the WebLogic Monitor role and the WebCenter Portal permission Portal Templates - Manage All.

10.11.4.2 Syntax

```
exportWebCenterPortalTemplates(appName, fileName, [names, exportPortalTemplateContent, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
</tbody>
</table>
The following example exports myPortalTemplate1 and myPortalTemplate2 to myPortalTemplateExport.par:

```
   wls:/weblogic/serverConfig> exportWebCenterPortalTemplates(appName='webcenter',
   fileName='myPortalTemplateExport.par', names='myPortalTemplate1,
   myPortalTemplate2')
```

The following example exports all the portal templates, including portal template content, to exportAllMyPortalTemplates.par:

```
   wls:/weblogic/serverConfig> exportWebCenterPortalTemplates(appName='webcenter',
   fileName='exportAllMyPortalTemplates.par', exportPortalTemplateContent=1)
```

### 10.11.5 importWebCenterPortals

**Module:** Oracle WebCenter Portal

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>fileName</td>
<td>Name and location of the export archive (.par file). For example, /myExports/myPortalTemplateExport.par. If you do not specify a location, the archive is created in the same directory that you run the WLST command.</td>
</tr>
<tr>
<td>names</td>
<td>Optional. Name of one or more portals templates that you want to export. Separate multiple template names with a comma. For example: names='mySalesTemplate,myHRTemplate'</td>
</tr>
<tr>
<td>exportPortalTemplateContent</td>
<td>Optional. Specifies whether to export each template's content folder on WebCenter Content Server. A content folder is automatically created on Content Server for any template using document services to create, manage, and store template documents (files, folders, wikis, blogs). Only content that is stored in this folder can be exported with the portal template. The export does not, for example, include web content or pages displayed through Content Presenter since this information is not stored in the portal template's content folder. Valid values are 1 and 0.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where WebCenter Portal is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the WebCenter Portal application is deployed.</td>
</tr>
</tbody>
</table>
Use with WLST: Online

10.11.5.1 Description
(WebCenter Portal only) Imports one or more portals or portal templates from a portal archive (.par file).

**Note:** To run this command you must have at least the WebLogic Monitor role and either the WebCenter Portal permission Portals - Manage All or Portal Templates - Manage All.

10.11.5.2 Syntax

```
importWebCenterPortals(appName, fileName, names, [parentPortal,
importCustomizations, importPortalContent, importSecurity, importData,
importActivities, overwrite, savePortals, forceOffline, importLog, server,
applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>appName</strong></td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td><strong>fileName</strong></td>
<td>Name of the portal archive (.par file) containing the portals or portal</td>
</tr>
<tr>
<td></td>
<td>templates you want to import. For example, mySalesPortal.par.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If you want to import a space archive (.ear file) that was</td>
</tr>
<tr>
<td></td>
<td>generated from a previous WebCenter Portal release, specify the name of the</td>
</tr>
<tr>
<td></td>
<td>.ear file. For example, myOldFinanceSpace.ear.</td>
</tr>
<tr>
<td><strong>names</strong></td>
<td>Name of one or more portals or portal templates that you want to</td>
</tr>
<tr>
<td></td>
<td>import. For example: names='sales'.</td>
</tr>
<tr>
<td></td>
<td>Separate multiple names with a comma. For example:</td>
</tr>
<tr>
<td></td>
<td>names='sales,finance'</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> Do not enter display names here. You must enter the name that is</td>
</tr>
<tr>
<td></td>
<td>specified in the portal or portal template URL:</td>
</tr>
<tr>
<td></td>
<td>■ Portal names are available from the About Portal dialog.</td>
</tr>
<tr>
<td></td>
<td>■ Portal template names are available from the About Portal Template</td>
</tr>
<tr>
<td></td>
<td>dialog.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The names argument is ignored if the archive that you want to</td>
</tr>
<tr>
<td></td>
<td>import was generated to an .ear file from an earlier WebCenter Portal</td>
</tr>
<tr>
<td></td>
<td>version, that is, version 11.1.1.7.0 or earlier.</td>
</tr>
<tr>
<td><strong>parentPortal</strong></td>
<td>Optional. Name of the parent portal under which to place portals in the</td>
</tr>
<tr>
<td></td>
<td>archive. If specified, imported portals become subportals of the parent</td>
</tr>
<tr>
<td></td>
<td>portal. This argument defaults to null. When no parent is specified,</td>
</tr>
<tr>
<td></td>
<td>archived portals are imported as root portals.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If the archive contains portal templates, this argument is</td>
</tr>
<tr>
<td></td>
<td>ignored.</td>
</tr>
</tbody>
</table>
### importCustomizations
Optional. Indicates whether to import customizations from the portal archive. Valid values are 1 and 0.

- **1** - Imports the portal's customizations. This includes:
  - **Portal-level customizations** (to system pages, assets, task flows, and so on)
  - **User-level customizations** (sometimes referred to as user personalizations)

  Import does not completely overwrite existing portal customizations on the target (if any). Only customizations included in the portal archive are overwritten on import, leaving any other customizations on the target unchanged.

- **0** - New portals are imported without customizations, for example, default task flows and default portal settings are imported.

  If you are importing a portal that already exists on the target, existing customizations on the target are preserved.

This argument defaults to 1.

**Note:** Portlet and page customizations are always imported.

**Note:** If the archive contains portal templates, this argument is ignored.

### importPortalContent
Optional. Specifies whether to import content associated with the portal or portal template, that is, import documents in the archive (if any) to the target content repository.

Valid values are 1 and 0.

- **1** - Imports content into the target portal or portal template's content folder. Existing folders are overwritten.

- **0** - Excludes portal (or portal template) content. This option is useful when migrating between stage and production environments where test content is no longer required.

This argument defaults to 0.

**Note:** Portal archives that contain large content folders may exceed the maximum upload size for files (2 GB by default). If necessary, you can increase this setting. For details, see "Changing the Maximum File Upload Size" in Oracle Fusion Middleware Administering Oracle WebCenter Portal.

### importSecurity
Optional. Indicates whether or not to import portal member details from the export archive. Valid values are 1 and 0.

- **1** - Imports roles and permissions for the portal, as well as member details and their role assignments

- **0** - Imports only portal roles and permissions. This option is useful when migrating between stage and production environments and where member details, added during the testing phase, are no longer required.

This argument defaults to 1.

**Note:** If the archive contains portal templates, this argument is ignored.
<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>importData</strong></td>
<td>Optional. Indicates whether to import data from the export archive.</td>
</tr>
<tr>
<td></td>
<td>Valid values are 1 and 0.</td>
</tr>
<tr>
<td></td>
<td>■ 1 - Portal-related data stored in the WebCenter Portal database</td>
</tr>
<tr>
<td></td>
<td>for activity streams, events, feedback, lists, links, message boards,</td>
</tr>
<tr>
<td></td>
<td>people connections, polls, profiles, and surveys is imported.</td>
</tr>
<tr>
<td></td>
<td>■ 0 - Portal data is not imported and data on the target (if any) is</td>
</tr>
<tr>
<td></td>
<td>preserved. This option is useful when migrating between stage</td>
</tr>
<tr>
<td></td>
<td>and production environments and where test data is no longer</td>
</tr>
<tr>
<td></td>
<td>required.</td>
</tr>
<tr>
<td></td>
<td>This argument defaults to 1.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> If the archive contains portal templates, this argument is</td>
</tr>
<tr>
<td></td>
<td>ignored.</td>
</tr>
</tbody>
</table>

| **importActivities** | Optional. Indicates whether to import activity messages from the export archive. Valid values are 1 and 0. |
|                     | ■ 1 - Activity messages stored in the portal archive are imported.        |
|                     | ■ 0 - Activity messages are not imported.                                 |
|                     | This argument defaults to 1.                                             |
|                     | **Note:** If the archive contains portal templates, this argument is     |
|                     |   ignored.                                                                |

| **overwrite**       | Optional. Specifies whether to overwrite portals or portal templates     |
|                     | that exist in the target.                                                |
|                     | Valid values are 1 and 0.                                                 |
|                     | ■ 1 - Overwrites existing portals (or portal templates).                  |
|                     | ■ 0 - Only imports new portals (or portal templates). Do not import      |
|                     |   portals or portal templates that exist on the target.                  |
|                     | The default is 0.                                                        |

| **savePortals**     | Optional. Specifies whether to save existing portals or portal templates |
|                     | to an archive before starting the import operation.                      |
|                     | Valid values are 1 and 0:                                                 |
|                     | ■ 1 - Before importing, back up existing portals or portal templates     |
|                     |   to an archive. If created, the archive is copied to the /tmp directory| |
|                     |   on the target machine.                                                 |
|                     | ■ 0 - Do not back up existing portals or portal templates.                |
|                     | The default value is 0.                                                  |

| **forceOffline**    | Optional. Specifies whether to take existing portals offline before      |
|                     | starting the import process.                                             |
|                     | Valid values are 1 and 0.                                                 |
|                     | ■ 1 - Takes the portals offline before starting the import process.      |
|                     | ■ 0 - Attempts to import the portals. If one or more portals are         |
|                     |   currently online, you are prompted to take the portals offline.       |
|                     | The default value is 0.                                                  |
|                     | **Note:** If the archive contains portal templates, this argument is     |
|                     |   ignored.                                                                |

| **importLog**       | Optional. Specifies the name of a local log file containing detailed     |
|                     | information about the import operation. If not specified, a log file     |
|                     |   named PortalImport_< timestamp>.log is generated in the temporary      |
|                     |   directory.                                                             |
10.11.5.3 Example

The following example imports a new version of the sales portal archived in myPortalExport.par and specifies a name and location for the import log file. To ensure that the existing sales portal is offline while the import operation takes place, forceOffline is set to 1:

wls:/weblogic/serverConfig> importWebCenterPortals(appName='webcenter', fileName='myPortalExport.par', names='sales', forceOffline=1, importLog='sales_import.log')

The following example saves a copy of the sales portal before re-importing a new version of the sales portal from myPortalExport.par:

wls:/weblogic/serverConfig> importWebCenterPortals(appName='webcenter', fileName='myPortalExport.par', names='sales', savePortals=1)

The following example imports the newHire portal template archived in myPortalTemplateExport.par and specifies a name and location for the import log file:

wls:/weblogic/serverConfig> importWebCenterPortals(appName='webcenter', fileName='myPortalTemplateExport.par', names='newHire', importLog='newHireTemplate_import.log')

10.11.6 listWebCenterPortalArchive

Module: Oracle WebCenter Portal

Use with WLST: Online or Offline

10.11.6.1 Description

(WebCenter Portal only) Lists the content of a portal archive and optionally, extracts the portal archive to a specified location.

10.11.6.2 Syntax

listWebCenterPortalArchive(appName, fileName, [extractDir, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>fileName</td>
<td>Name and location of a portal archive.</td>
</tr>
</tbody>
</table>
10.11.6.3 Examples

The following example lists the content of myPortal.par and extracts archive content to the directory /myPortalArchives/myPortalContent:

wls:/weblogic/serverConfig> listWebCenterPortalArchive(appName='webcenter', fileName='/myPortalArchives/myPortal.par', extractDir='/myPortalArchives/myPortalContent')

10.11.7 exportWebCenterPortalChanges

Module: Oracle WebCenter Portal
Use: Online (IBM WebSphere only)

10.11.7.1 Description

(WebCenter Portal only) Exports portal metadata changes for a named portal to a portal archive (.par file), using the filename specified.

Portal metadata changes exported:

- **Includes:**
  - Portal-level customizations (metadata changes) for portal pages, system pages, portlets, assets, task flows
  - User-level customizations (metadata changes) for portal pages, portlets, task flow instances

- **Excludes:** subportals, security, any changes to content and data, portal state (online/offline)

**Notes:**

- You can only export changes for a portal that was previously exported using the exportWebCenterPortals WLST command.
- You must have at least the WebLogic Monitor role and the WebCenter Portal permission Portals - Manage All to run this command.

10.11.7.2 Syntax

exportWebCenterPortalChanges(appName, fileName, portalName, [server,
The following example exports metadata changes for a portal named \textit{myPortal} to \textit{myPortalChangesExport.par}:

\begin{verbatim}
wsadmin> WebCenter.exportWebCenterPortalChanges(appName='webcenter', fileName='myPortalChangesExport.par', portalName='myPortal')
\end{verbatim}

### 10.11.8 \texttt{importWebCenterPortalChanges}

Module: Oracle WebCenter Portal  
Use: Online (IBM WebSphere only)

#### 10.11.8.1 Description

(WebCenter Portal only) Imports metadata changes from a portal archive (.par file), using the filename specified.

\begin{itemize}
  \item You can only import changes for a portal that was previously exported using the \texttt{importWebCenterPortals} WLST command.
  \item You must have at least the WebLogic Monitor role and the WebCenter Portal permission Portals - Manage All to run this command.
\end{itemize}

#### 10.11.8.2 Syntax

\begin{verbatim}
importWebCenterPortalChanges\{appName, fileName, [importLog, server, applicationVersion]\}
\end{verbatim}
The following example imports metadata changes for a portal from myPortalChangesExport.par:

```plaintext
wsadmin> WebCenter.importWebCenterPortalChanges(appName='webcenter',
fileName='myPortalChangesExport.par')
```

The following example imports metadata changes for a portal from myPortalChangesExport.par and specifies a name and location for the import log file:

```plaintext
wsadmin> WebCenter.importWebCenterPortalChanges(appName='webcenter',
fileName='myPortalChangesExport.par', importLog='myPortalImport.log')
```

### 10.11.9 exportWebCenterPortalConnections

**Module:** Oracle WebCenter Portal  
**Use with WLST:** Online  

#### 10.11.9.1 Description

(WebCenter Portal only) Exports connection configuration information from the source WebCenter Portal environment to a named file.

The following connections are exported:

- Analytics collector connections
- Content repository connections (Oracle WebCenter Content, Oracle Portal, file system location)
- Discussions server connections
- Events server connections
- External application connections
- Mail server connections
- Oracle Secure Enterprise Search connections
- Portlet producer connections (WSRP and PDK-Java)
- Pagelet producer connections
- Presence server connections
- URL connections
- Web service connections (used by data controls)
- Worklist / BPEL server connections

---

**Note:**

- Personalization provider connections (for CMIS, Activity Graph, and People Connections) are not exported.
- You must have at least the WebLogic Admin role to run the `exportWebCenterPortalConnections` command.

---

**10.11.9.2 Syntax**

```
exportWebCenterPortalConnections(appName, fileName, [connectionType, [connectionName,] logFile, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td><code>fileName</code></td>
<td>Specifies a name and location for the connection properties file. If you do not specify a location, the file is saved in the current directory.</td>
</tr>
<tr>
<td>Argument</td>
<td>Definition</td>
</tr>
<tr>
<td>---------------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| connectionType | Optional. Specifies the type of connections you want to export. Separate multiple connection types with a comma. For example:  
|               | connectionType='wsrpProducerConnection,  
|               | webServiceConnection'  
|               | If left blank, all connection types are exported. Valid connection types are:  
|               | ■ analyticsCollectorConnection (Analytics collector)  
|               | ■ webcenterContentServerConnection (Oracle WebCenter Content)  
|               | ■ oraclePortalConnection (Oracle Portal)  
|               | ■ fileSystemConnection (File system)  
|               | ■ discussionConnection (Discussions server)  
|               | ■ personalEventConnection (Presence server)  
|               | ■ externalAppConnection (External application)  
|               | ■ mailConnection (Mail server)  
|               | ■ sesConnection (Oracle Secure Enterprise Search)  
|               | ■ wsrpProducerConnection (WSRP portlet producer)  
|               | ■ jpdkProducerConnection (PDK-Java portlet producer)  
|               | ■ pageletProducerConnection (Pagelet producer)  
|               | ■ impConnection (Presence server)  
|               | ■ urlConnection (URL)  
|               | ■ webServiceConnection (Web service)  
|               | ■ bpe1Connection (Worklist / BPEL server)  
|               | **Note:** This argument is mandatory if you specify connectionName. |
| connectionName | Optional. Names specific connections you want to export for a single connectionType. Separate multiple connection names with a comma. For example, if connectionType='wsrpProducerConnection', you can specify to export one or more connections of this type:  
|               | connectionName='myWSRPProducer1,myWSRPProducer2'  
|               | When you specify a value for connectionName, you must specify the connectionType argument as well. For example:  
|               | exportWebCenterPortalConnections(appName='webcenter',  
|               | fileName='/scratch/conn.properties',  
|               | connectionType='wsrpProducerConnection',  
|               | connectionName='MyWSRPConn')  
|               | **Note:** If no names are specified, all connections are exported for the specified connectionType. |
| logFile        | Specifies a name of a local log file containing detailed information about the export connection operation.  
|               | If not specified, a log file named ConnectionExport_<timestamp>.log is generated in the temporary directory. |
| server         | Optional. Name of the managed server where WebCenter Portal is deployed. For example, WC_Spaces.  
|               | Required when applications with the same name are deployed to different servers and also when you have a cluster. |
| applicationVersion | Optional. Version number of the deployed application. Required if more than one version of the WebCenter Portal application is deployed. |
10.11.9.3 Examples

The following example exports connection configuration information for all WSRP producer and web service connections to a file named connection.properties located at /myConnections:

```shell
wls:/weblogic/serverConfig> exportWebCenterPortalConnections(appName='webcenter',
               fileName='/myConnections/connection.properties',
               connectionType='wsrpProducerConnection,webServiceConnection')
```

The following example exports connection configuration information for two WSRP producer connections named myWSRP1 and myWSRP2 to a file named connection.properties located at /myConnections:

```shell
wls:/weblogic/serverConfig> exportWebCenterPortalConnections(appName='webcenter',
               fileName='/myConnections/connection.properties',
               connectionType='wsrpProducerConnection', connectionName='myWSRP1,myWSRP2')
```

The following example exports all connection configuration information to a file named connection.properties located at /myConnections:

```shell
wls:/weblogic/serverConfig> exportWebCenterPortalConnections(appName='webcenter',
               fileName='/myConnections/connection.properties')
```

The following example exports connection configuration information to a file named connection.properties located at /myConnections. Detailed information about the export operation is also logged to exportConnections.log located at /myExportLogs:

```shell
wls:/weblogic/serverConfig> exportWebCenterPortalConnections(appName='webcenter',
               fileName='/myConnections/connection.properties',
               logFile='/myExportLogs/exportConnections.log')
```

10.11.10 importWebCenterPortalConnections

Module: Oracle WebCenter Portal

Use with WLST: Online

10.11.10.1 Description

(WebCenter Portal only) Imports new WebCenter Portal connections from a named connection properties file. Connections that do not exist on the target are imported. Connections that exist on the target are ignored.

The following connections can be imported:

- Analytics collector connections
- Content repository connections (Oracle WebCenter Content, Oracle Portal, file system location)
- Discussions server connections
- Events server connections
- External application connections
- Mail server connections
- Oracle Secure Enterprise Search connections
- Portlet producer connections (WSRP and PDK-Java)
- Pagelet producer connections
- Presence server connections
- URL connections
- Web service connections (used by data controls)
- Worklist / BPEL server connections

---

**Note:**
- You must have at least the WebLogic Admin role to run the `importWebCenterPortalConnections` command.
- You can only import connection information that was previously exported using the `exportWebCenterPortalConnections` command.
- Newly imported portlet producer, external application, URL, and web service connections are immediately available in the target. For the other connection types, you are prompted to restart the managed server on which WebCenter Portal is deployed to make new connections available.

---

### 10.11.10.2 Syntax

```
importWebCenterPortalConnections(appName, fileName, [promptForPassword, logFile, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code></td>
</tr>
<tr>
<td><code>fileName</code></td>
<td>Specifies the name and location of a connection properties file.</td>
</tr>
</tbody>
</table>
| `promptForPassword` | Optional. Specifies whether to prompt the user for a password if credentials are required to create a new connection during the import operation. For example, you must enter a password to create a new Oracle-Secure Enterprise Search (SES) connection. Valid values are 1 and 0:   
  - 1 - Prompts the user to enter credentials if a password is required.   
  - 0 - Do not prompt the user to enter passwords if required to create a new connection. Always set this argument to 0 if you run this command within a script. The default is 1. |
| `logFile`           | Optional. Specifies the name of a local log file in which to record detailed information about the import connection operation. If no value is specified, a log file named `ConnectionImport_<timestamp>.log` is generated in the temporary directory. |
| `server`            | Optional. Name of the managed server where WebCenter Portal is deployed. For example, `WC_Spaces`. Required when applications with the same name are deployed to different servers and also when you have a cluster. |
| `applicationVersion`| Optional. Version number of the deployed application. Required if more than one version of the WebCenter Portal application is deployed. |
10.11.10.3 Examples

The following example imports connections defined in a file named connection.properties, located at /myConnections:

```
wls:/weblogic/serverConfig> importWebCenterPortalConnections(appName='webcenter', fileName='/myConnections/connection.properties')
```

The following example imports connections defined in a file named connection.properties, located at \myConnections. Detailed information about the import operation is also logged to importConnection.log:

```
wls:/weblogic/serverConfig> importWebCenterPortalConnections(appName='webcenter', fileName='/myConnections/connection.properties', logFile='importConnection.log')
```

The following example imports connections defined in a file named connection.properties, located at \myConnections. Do not prompt the user to enter credentials (if required):

```
wls:/weblogic/serverConfig> importWebCenterPortalConnections(appName='webcenter', fileName='/myConnections/connection.properties', promptForPassword=0)
```

10.11 setSpaceState

Module: Oracle WebCenter Portal

Use with WLST: Online

10.11.11.1 Description

(WebCenter Portal only) Takes a portal offline or brings a portal online.

10.11.11.2 Syntax

```
spaceState(appName, spaceName, offline, [server, applicationVersion])
```

10.11.11.3 Example

The following example takes MyPortal offline:

```
wls:/weblogic/serverConfig> setSpaceState(appName='webcenter', spaceName='MyPortal', offline=1)
```
10.11.12 exportWebCenterResource

Module: Oracle WebCenter Portal
Use with WLST: Online

10.11.12.1 Description

(WebCenter Portal and Portal Framework applications) Exports a single asset, device, or device group to an export archive (.ear file), using the filename specified.

When you export an asset, you can specify either resourceGUID or resourceName.
When you export a device or device group, you can only specify resourceName.

---

Notes:

- To run this command you must have at least the WebLogic Monitor role, as well as the appropriate Create/Edit/Delete permission for the type of asset, device or device group you want to export.
  
  For more information, see "About Permissions Required to Import (or Export) Assets" in Oracle Fusion Middleware Administering Oracle WebCenter Portal.

- You cannot export out-of-the-box assets, devices, or device groups.

---

10.11.12.2 Syntax

exportWebCenterResource(appName, fileName, resourceType, [resourceGUID, resourceName, spaceName, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>fileName</td>
<td>Name of the local file to which the export will be written.</td>
</tr>
<tr>
<td>Argument</td>
<td>Definition</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>resourceType</td>
<td>Type of resource to export (an asset, device, or device group). Valid values include: pageTemplate, contentPresenter, pageStyle, navigation, resourceCatalog, skin, taskFlow, mashupStyle, dataControl, device, deviceGroup.</td>
</tr>
<tr>
<td></td>
<td>Where:</td>
</tr>
<tr>
<td></td>
<td>- pageTemplate - page template</td>
</tr>
<tr>
<td></td>
<td>- contentPresenter - Content presenter display template</td>
</tr>
<tr>
<td></td>
<td>- pageStyle - page style</td>
</tr>
<tr>
<td></td>
<td>- navigation - navigation models</td>
</tr>
<tr>
<td></td>
<td>- resourceCatalog - resource catalog</td>
</tr>
<tr>
<td></td>
<td>- skin - skin</td>
</tr>
<tr>
<td></td>
<td>- taskFlow - task flow</td>
</tr>
<tr>
<td></td>
<td>- taskflowStyle - task flow style</td>
</tr>
<tr>
<td></td>
<td>- dataControl - data control</td>
</tr>
<tr>
<td></td>
<td>- device - device</td>
</tr>
<tr>
<td></td>
<td>- deviceGroup - device group</td>
</tr>
<tr>
<td></td>
<td>You cannot export or import out-of-the-box assets, devices, or device groups.</td>
</tr>
<tr>
<td></td>
<td>Note: In this release, taskFlowStyle replaces the asset type mashupStyle. The mashupStyle option is deprecated but continues to work in this release for backward compatibility.</td>
</tr>
<tr>
<td>resourceGUID</td>
<td>Optional. Unique ID (GUID) of an asset to export. Internal IDs are available from the About dialog for the asset. This argument is not used when resourceType is set to device or deviceGroup.</td>
</tr>
<tr>
<td>resourceName</td>
<td>Optional. Display name of an asset, device, or device group to export. Asset display names are available from the About dialog for the asset. Device and device group names are available from the Edit dialog for the device or device group. Note: You cannot export out-of-the-box assets, devices, or device groups.</td>
</tr>
<tr>
<td>spaceName</td>
<td>Optional. (WebCenter Portal only) Name of the portal containing the asset to export. Use this argument to export portal assets, that is, assets that are owned by a particular portal. Omit this argument if you want to export shared assets for WebCenter Portal. This argument defaults to null (shared assets are exported). This argument is not used when resourceType is set to device or deviceGroup. This argument is not required when exporting assets for a Portal Framework application.</td>
</tr>
<tr>
<td>exportContentDirectory</td>
<td>Deprecated. You can no longer include MDS content associated with assets in export archives. Use the standard MDS WLST command exportMetadata to migrate legacy MDS content, if required. Note: In 11.1.1.8.0 or later, Oracle recommends that you store asset-related content on your content server rather than MDS.</td>
</tr>
</tbody>
</table>
The following example exports a page template owned by a portal named MyPortal (in WebCenter Portal) to a local file named myPageTemplateExport.ear:

```bash
wls:/weblogic/serverConfig> exportWebCenterResource(appName='webcenter',
fileName='myPageTemplateExport.ear', resourceType='pageTemplate',
resourceGUID='gsr47d9a5ac_7398_439a_97d2_8b54ce905f7e, spaceName='MyPortal')
```

The following example exports the same page template owned by a portal named MyPortal but specifies the template's display name rather than the GUID:

```bash
wls:/weblogic/serverConfig> exportWebCenterResource(appName='webcenter',
fileName='myPageTemplateExport.ear', resourceType='pageTemplate',
resourceName='MyPageTemplate, spaceName='MyPortal')
```

The following example exports a device named MyMobileDevice from WebCenter Portal:

```bash
wls:/weblogic/serverConfig> exportWebCenterResource(appName='webcenter',
fileName='myDeviceExport.ear', resourceType='device',
resourceName='MyMobileDevice')
```

The following example exports a skin from a Portal Framework application named myPortalApp to a local file named mySkinExport.ear:

```bash
wls:/weblogic/serverConfig> exportWebCenterResource(appName='myPortalApp',
fileName='mySkinExport.ear', resourceType='skin',
resourceGUID='gsr47d9a5ac_7398_439a_97d2_8b54ce905f7e)
```

### 10.11.13 importWebCenterResource

**Module:** Oracle WebCenter Portal  
**Use with WLST:** Online

#### 10.11.13.1 Description

(WebCenter Portal and Portal Framework applications) Imports a single asset, device, or device group, from an export archive (.ear file).
10.11.13.2 Syntax
importWebCenterResource(appName, fileName, [resourceType, spaceName, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>fileName</td>
<td>Name of the archive file that you want to import.</td>
</tr>
<tr>
<td>resourceType</td>
<td>Optional. Type of resource to import (an asset, device, or device group). Valid values include: pageTemplate, contentPresenter, pageStyle, navigation, resourceCatalog, skin, taskFlow, mashupStyle, dataControl, device, deviceGroup</td>
</tr>
<tr>
<td></td>
<td>Where:</td>
</tr>
<tr>
<td></td>
<td>■ pageTemplate - page template</td>
</tr>
<tr>
<td></td>
<td>■ contentPresenter - Content presenter display template</td>
</tr>
<tr>
<td></td>
<td>■ pageStyle - page style</td>
</tr>
<tr>
<td></td>
<td>■ navigation - navigation models</td>
</tr>
<tr>
<td></td>
<td>■ resourceCatalog - resource catalog</td>
</tr>
<tr>
<td></td>
<td>■ skin - skin</td>
</tr>
<tr>
<td></td>
<td>■ taskFlow - task flow</td>
</tr>
<tr>
<td></td>
<td>■ taskflowStyle - task flow style</td>
</tr>
<tr>
<td></td>
<td>■ dataControl - data control</td>
</tr>
<tr>
<td></td>
<td>■ device - device</td>
</tr>
<tr>
<td></td>
<td>■ deviceGroup - device group</td>
</tr>
</tbody>
</table>

If the archive (.ear file) contains one or more devices or device groups, then this argument is mandatory:

■ When resourceType='device', all devices in the archive are imported.

■ When resourceType='deviceGroup', all device groups in the archive and their associated devices are imported.

Note: In this release, taskFlowStyle replaces the asset type mashupStyle. The mashupStyle option is deprecated but continues to work in this release for backward compatibility.
The following example imports a page template from an archive named myPageTemplateExport.ear to MyPortal in WebCenter Portal:

```bash
wls:/weblogic/serverConfig> importWebCenterResource(appName='webcenter',
fileName='myPageTemplateExport.ear', spaceName='MyPortal',
resourceType='pageTemplate')
```

The following example imports a device from an archive named myDeviceExport.ear to WebCenter Portal:

```bash
wls:/weblogic/serverConfig> importWebCenterResource(appName='webcenter',
fileName='myDeviceExport.ear', resourceType='device')
```

The following example imports a skin from an archive named mySkinExport.ear to a Portal Framework application named myPortalApp:

```bash
wls:/weblogic/serverConfig> importWebCenterResource(appName='myPortalApp',
fileName='mySkinExport.ear', resourceType='skin')
```

### 10.11.13.3 Example

The following example imports a page template from an archive named myPageTemplateExport.ear to MyPortal in WebCenter Portal:

```bash
wls:/weblogic/serverConfig> importWebCenterResource(appName='webcenter',
fileName='myPageTemplateExport.ear', spaceName='MyPortal',
resourceType='pageTemplate')
```

The following example imports a device from an archive named myDeviceExport.ear to WebCenter Portal:

```bash
wls:/weblogic/serverConfig> importWebCenterResource(appName='webcenter',
fileName='myDeviceExport.ear', resourceType='device')
```

The following example imports a skin from an archive named mySkinExport.ear to a Portal Framework application named myPortalApp:

```bash
wls:/weblogic/serverConfig> importWebCenterResource(appName='myPortalApp',
fileName='mySkinExport.ear', resourceType='skin')
```

### 10.11.14 importWebCenterTranslations

Module: Oracle WebCenter

Use with WLST: Online

#### 10.11.14.1 Description

10.11.14.2 Syntax

importWebCenterTranslations(appName, server, mdsRootDir, [applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the name is always webcenter.</td>
</tr>
<tr>
<td>server</td>
<td>Name of the target managed server on which WebCenter Portal is deployed. For example, WC_Spaces.</td>
</tr>
<tr>
<td>mdsRootDir</td>
<td>MDS root directory on the file system that contains translated XLF files.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of WebCenter Portal is deployed.</td>
</tr>
</tbody>
</table>

10.11.14.3 Example

The following example imports translated content in the directory /scratch/shared/newmd to MDS and the WebCenter Portal repository:

```
wlsc:/weblogic/serverConfig> importWebCenterTranslations(appName='webcenter', server='WC_Spaces', mdsRootDir='/scratch/shared/newmd')
```

10.11.15 exportWebCenterApplication

Module: Oracle WebCenter Portal

Use with WLST: Online

10.11.15.1 Description

(WebCenter Portal only) Exports an entire WebCenter Portal application to an export archive (.ear file) using the filename provided.

10.11.15.2 Syntax

exportWebCenterApplication(appName, fileName, [exportCustomizations, exportData, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the name is always webcenter.</td>
</tr>
<tr>
<td>fileName</td>
<td>Name of the export archive (.ear file) to which you want the export to be written.</td>
</tr>
<tr>
<td>exportCustomizations</td>
<td>Optional. Specifies whether to export global customizations and user customizations. Valid values are 1 and 0:</td>
</tr>
<tr>
<td></td>
<td>1 - Export global customizations and user customizations.</td>
</tr>
<tr>
<td></td>
<td>0 - Exclude customizations. On import, existing customizations on the target to system pages, task flows, assets, and so on, remain intact.</td>
</tr>
<tr>
<td>This argument defaults to 1.</td>
<td></td>
</tr>
</tbody>
</table>
### 10.11.15.3 Examples

The following example exports WebCenter Portal with customizations and all possible data to a file named `myAppExport.ear`.

```plaintext
wls:/weblogic/serverConfig> exportWebCenterApplication(appName='webcenter', fileName='myAppExport.ear', exportCustomizations=1, exportData=1)
```

The following example exports a test WebCenter Portal instance. In this case, data created during testing (such as lists, events, links, tags, and so on) is not required.

```plaintext
wls:/weblogic/serverConfig> exportWebCenterApplication(appName='webcenter', fileName='myTestAppExport.ear', exportCustomizations=1, exportData=0)
```

### 10.11.16 importWebCenterApplication

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.11.16.1 Description

(WebCenter Portal only) Imports an entire WebCenter Portal application from an export archive file to a managed server.

After importing WebCenter Portal you must restart the managed server on which you deployed the application.

#### 10.11.16.2 Syntax

`importWebCenterApplication(appName, fileName, [server, applicationVersion])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td>fileName</td>
<td>Name of the export archive that you want to import.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where WebCenter Portal is deployed.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of WebCenter Portal is deployed.</td>
</tr>
</tbody>
</table>
Example

The following example imports WebCenter Portal from the export archive myAppExport.ear.

```
> wls:/weblogic/serverConfig> importWebCenterApplication(appName='webcenter', fileName='myAppExport.ear')
```

### 10.11.17 exportPortletClientMetadata

**Module:** Oracle WebCenter Portal

**Use with WLST:** Online

#### 10.11.17.1 Description

(WebCenter Portal and Portal Framework applications) Exports portlet client metadata and producer customizations and personalizations, for WebCenter Portal or a Portal Framework application. This command exports metadata for all the application’s producers to a named export archive (.ear file). You cannot opt to export metadata for specific producers.

#### 10.11.17.2 Syntax

```
exportPortletClientMetadata(appName, fileName, [exportPersonalizations, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>fileName</td>
<td>Name of the export archive (.ear file) to which you want the export to be written.</td>
</tr>
<tr>
<td>exportPersonalizations</td>
<td>Optional. Valid values are 1 (true) and 0 (false). For WebCenter Portal, it is always 1. For Portal Framework applications, it is optional. 1 - Personalizations for all producers are exported. 0 - Personalizations are not exported. This argument defaults to 1.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.11.17.3 Example
The following example exports portlet client metadata and producer customizations to an export archive named myExport.ear. Personalizations are not exported.

```bash
wls:/weblogic/serverConfig> exportPortletClientMetadata(appName='myApp', fileName='myExport.ear', exportPersonalizations=0)
```

The following example exports portlet client metadata for an application with the version number V2.0 deployed on the server WC_CustomPortal1.

```bash
wls:/weblogic/serverConfig> exportPortletClientMetadata(appName='myApp', fileName='myExport.ear', server='WC_CustomPortal1', applicationVersion='V2.0')
```

10.11.18 importPortletClientMetadata
Module: Oracle WebCenter Portal
Use with WLST: Online

10.11.18.1 Description
(WebCenter Portal and Portal Framework applications) Imports portlet client metadata and producer customizations and personalizations from a named export archive.

Producer personalizations are optional on export. Producer personalizations are imported if the export archive specified includes personalizations.

10.11.18.2 Syntax
```bash
importPortletClientMetadata(appName, fileName, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>fileName</td>
<td>Name of the export archive that you want to import.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.11.18.3 Example
The following example imports portlet client metadata and producer customizations and personalizations from an export archive named myExport.ear.

```bash
wls:/weblogic/serverConfig> importPortletClientMetadata(appName='myApp', fileName='myExport.ear')
```

10.11.19 showProducerImportFailures
Module: Oracle WebCenter Portal
Use with WLST: Online
10.11.19.1 Description
(WebCenter Portal and Portal Framework applications) Lists outstanding producer imports for a named application.

Producer import fails if a producer used by the application is not available when the application first starts after deployment or an import operation.

10.11.19.2 Syntax
showProducerImportFailures(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Name of the managed server on which the application is deployed.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if</td>
</tr>
<tr>
<td></td>
<td>more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.11.19.3 Example
The following example shows producer import failures for WebCenter Portal (webcenter):

```
wlsh:/weblogic/serverConfig> showProducerImportFailures(appName='webcenter')
```

10.11.20 retryAllFailedProducerImports

Module: Oracle WebCenter Portal

Use with WLST: Online

10.11.20.1 Description
(WebCenter Portal and Portal Framework applications) Imports outstanding producer metadata.

Producer import can fail if a producer used by the application is not available when the application first starts after deployment or an import operation. Use this command to import metadata for any producers for which metadata import previously failed.

10.11.20.2 Syntax
retryAllFailedProducerImports(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Name of the managed server on which the application is deployed.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if</td>
</tr>
<tr>
<td></td>
<td>more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.11.20.3 Example
The following example imports missing producer metadata for WebCenter Portal (webcenter):

```java
wls:/weblogic/serverConfig> retryAllFailedProducerImports(appName='webcenter')
```

Importing metadata from the following producers failed for application webcenter.

Producer Description : Serialised stack trace:

```
[[
  oracle.portlet.client.container.PortletHttpException: HTTP <unknown method>
  request to URL...
  ...
  Failure Id : /oracle/adf/portlet/producerImportFailures/producerImportFailure2
  Producer Id : /oracle/adf/portlet/JSR286FilePref
  Producer Name : JSR286FilePref
```

Tried to re-import producer metadata for application webcenter.

**Attempt to re-import producer metadata succeeded.**

---

**Note:** Errors and exceptions that occurred during a previous attempt to import producers display so you can see which failed producers the command is attempting to re-import.

10.11.21 cloneWebCenterManagedServer

Module: Oracle WebCenter Portal
Use with WLST: Online

10.11.21.1 Description

(WebCenter Portal and Portal Framework applications) Creates a new managed server with the same resources as a specified, base managed server.

10.11.21.2 Syntax

```
cloneWebCenterManagedServer(baseManagedServer, newManagedServer, newManagedServerPort, [verbose])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseManagedServer</td>
<td>Name of the base managed server.</td>
</tr>
<tr>
<td>newManagedServer</td>
<td>Name for the new, cloned managed server.</td>
</tr>
<tr>
<td>newManagedServerPort</td>
<td>Port number for the new managed server.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Creates the managed server in verbose mode. Valid values are 1 and 0. When set to 1, additional progress information displays during the creation process which is useful for diagnostic purposes. The default is 0.</td>
</tr>
</tbody>
</table>

10.11.21.3 Example

The following example creates a clone of the WC_CustomPortal managed server. The new managed server is named WC_CustomPortal2:

```
wls:/weblogic/serverConfig> cloneWebCenterManagedServer(baseManagedServer='WC_
10.11.22 exportGroupSpaces (Deprecated)

Module: Oracle WebCenter Portal
Use with WLST: Online

10.11.22.1 Description

(Deprecated from 11.1.1.8.0. Use exportWebCenterPortals instead.)

Exports one or more named portals (previously called spaces) to an export archive (.ear file), using the filename specified.

Space-related data*, application customizations, and security information is included in the export archive.

*Only internal space-related data stored in the Spaces database is exported. For example, data associated with tool or services such as activity streams, events, feedback, lists, links, message boards, people connections, profiles, polls, surveys, and tags.

You must take the spaces offline, even if only temporarily, to prevent data conflicts during the export process.

You must have at least the WebLogic Monitor role and the WebCenter Portal permission Portals - Manage All to run this command.

---

Note: You cannot use this command to export the Home space.

10.11.22.2 Syntax

`exportGroupSpaces(appName, fileName, names, [forceOffline, server, applicationVersion])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the Spaces application in which to perform this operation—always webcenter.</td>
</tr>
<tr>
<td><code>fileName</code></td>
<td>Name of the local file to which the export will be written.</td>
</tr>
<tr>
<td><code>names</code></td>
<td>Names of the spaces that you want to export. Separate multiple space names with a comma. For example: names='sales, finance' Note: Do not enter display names here. You must enter the space name that is specified in the space URL. The space name is available from the About dialog.</td>
</tr>
<tr>
<td><code>forceOffline</code></td>
<td>Optional. Specifies whether to take the spaces offline before starting export process. Valid values are 1 and 0. 1 takes the spaces offline before starting the export process. 0 attempts to export the spaces. If one or more spaces are currently online, an information message requests that you take the spaces offline. The default is 0.</td>
</tr>
</tbody>
</table>
Example

The following example exports two spaces (mySpace1 and mySpace2):

```bash
wls:/weblogic/serverConfig> exportGroupSpaces(appName='webcenter',
        fileName='myExport.ear', names='mySpace1, mySpace2')
```

The following example takes mySpace1 and mySpace2 offline and then exports both spaces, to an archive named myExport.ear:

```bash
wls:/weblogic/serverConfig> exportGroupSpaces(appName='webcenter',
        fileName='myExport.ear', names='mySpace1, mySpace2', forceOffline=1)
```

**10.11.23 exportGroupSpaceTemplates (Deprecated)**

Module: Oracle WebCenter Portal

Use with WLST: Online

**10.11.23.1 Description**

(Deprecated from 11.1.1.8.0. Use `exportWebCenterPortalTemplates` instead.)

Exports one or more portal templates (previously called space templates) to an export archive (.ear file), using the filename specified.

You must have at least the WebLogic Monitor role and the WebCenter Portal permission Portal Templates - Manage All to run this command.

**10.11.23.2 Syntax**

```bash
exportGroupSpaceTemplates(appName, fileName, names,
        [exportContentDirectory, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the Spaces application in which to perform this operation—always webcenter.</td>
</tr>
<tr>
<td><code>fileName</code></td>
<td>Name of the local file to which the export will be written.</td>
</tr>
<tr>
<td><code>names</code></td>
<td>Names of the space templates that you want to export. Separate multiple template names with a comma.</td>
</tr>
</tbody>
</table>
10.11.24 importGroupSpaces (Deprecated)

Module: Oracle WebCenter Portal

Use with WLST: Online

10.11.24.1 Description

(Deprecated from 11.1.1.8.0. Use importWebCenterPortals instead.)

Imports one or more portals or portal templates (previously called spaces and space templates) from an export archive.

You must have at least the WebLogic Monitor role and either the WebCenter Portal permission Portals - Manage All or Portal Templates - Manage All to run this command.

Note: You must take existing spaces offline, even if only temporarily, to prevent data conflicts during the import process.

10.11.24.2 Syntax

importGroupSpaces {appName, fileName, [importCustomizations, importSecurity, importData, parentSpace, forceOffline, server, applicationVersion]}

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the Spaces application in which to perform this operation—always webcenter.</td>
</tr>
<tr>
<td>fileName</td>
<td>Name of the archive file that you want to import.</td>
</tr>
</tbody>
</table>

Note: In 11.1.1.8.0 or later, Oracle recommends that you store asset-related content on your content server rather than MDS.
<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
</table>
| importCustomizations | Optional. Indicates whether to import space customizations from the export archive. Valid values are 1 and 0.  
  * 1 - Import space customizations.  
  * 0 - New spaces are imported without customizations (that is, default task flows are imported without any customizations and the default space settings are used).  
    
  If you are importing a space that already exists on the target, existing customizations on the target are preserved.  
    
  This argument defaults to 1.  
  
  Note: Portlet and page customizations are always imported.  
  
  Note: If the archive contains portal templates, this argument is ignored. |
| importSecurity    | Optional. Indicates whether to import space member details from the export archive. Valid values are 1 and 0.  
  * 1 - Roles and permissions for the space, as well as member details and their role assignments are imported.  
  * 0 - Only the roles and permissions are imported. This option is useful when migrating between stage and production environments and where member details, added during the testing phase, are no longer required.  
    
  This argument defaults to 1.  
  
  Note: If the archive contains portal templates, this argument is ignored. |
| importData        | Optional. Indicates whether to import data from the export archive. Valid values are 1 and 0.  
  * 1 - Space-related data stored in the Spaces database for activity streams, events, feedback, lists, links, message boards, people connections, profiles, polls, surveys, and tags is imported.  
  * 0 - Data is not imported. This option is useful when migrating between stage and production environments and where test data is no longer required.  
    
  This argument defaults to 1.  
  
  Note: If the archive contains portal templates, this argument is ignored. |
| forceOffline      | Optional. Takes the spaces offline before import. Valid values are 1 and 0.  
  * 1 - All spaces are taken offline.  
  * 0 - The current state of individual spaces (online or offline) is preserved.  
    
  This argument defaults to 0. |
| parentSpace       | Optional. Name of the parent space under which to place spaces in the archive. If specified, imported spaces become children of the parent space.  
    
  This argument defaults to null. When no parent is specified, archived spaces are imported as root spaces.  
  
  Note: If the archive contains space templates, this argument is ignored. |
The following example imports spaces or space templates from an archive named `myExport.ear`:

```bash
wls:/weblogic/serverConfig> importGroupSpaces(appName='webcenter',
    fileName='myExport.ear')
```

The following example takes all existing spaces in the target in `myExport.ear` offline and then imports all the spaces in `myExport.ear` under the "Sales" space. Space customizations are imported too. Test data and security details are not imported:

```bash
wls:/weblogic/serverConfig> importGroupSpaces(appName='webcenter',
    fileName='myExport.ear',
    importCustomizations=1, importSecurity=0, importData=0,
    parentSpace="Sales", forceOffline=1)
```

### 10.12 Mail

Use the commands listed in Table 10–19 to manage mail server connections for a named application.

You can register multiple mail server connections:

- **WebCenter Portal** supports multiple mail connections. The mail connection configured with `default=1` is the default connection for mail services in WebCenter Portal. All additional connections are offered as alternatives; WebCenter Portal users can choose which one they want to use through user preferences.
Portal Framework applications only use one mail connection—the connection configured with default=1. Any additional connections are ignored.

Configuration changes made using these WLST commands are only effective after your restart the Managed Server on which the application is deployed. For details, see Oracle Fusion Middleware Administering Oracle WebCenter Portal.

**Table 10–19  Mail WLST Commands**

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>createMailConnection</td>
<td>Create a mail server connection for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>setMailConnection</td>
<td>Edit an existing mail server connection.</td>
<td>Online</td>
</tr>
<tr>
<td>setMailConnectionProperty</td>
<td>Set mail server connection properties.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteMailConnectionProperty</td>
<td>Delete a mail server connection property.</td>
<td>Online</td>
</tr>
<tr>
<td>listMailConnections</td>
<td>List all of the mail server connections that are configured for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>listDefaultMailConnection</td>
<td>List the default mail server connection that is configured for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>setDefaultMailConnection</td>
<td>Set a specified connection as the default mail server connection.</td>
<td>Online</td>
</tr>
<tr>
<td>setMailServiceProperty</td>
<td>Specify defaults for mail.</td>
<td>Online</td>
</tr>
<tr>
<td>removeMailServiceProperty</td>
<td>Remove defaults for mail.</td>
<td>Online</td>
</tr>
<tr>
<td>listMailServiceProperties</td>
<td>List properties for mail.</td>
<td>Online</td>
</tr>
<tr>
<td>createMailExtApp</td>
<td>Create an external application suitable for mail connections.</td>
<td>Online</td>
</tr>
</tbody>
</table>

### 10.12.1 createMailConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

**10.12.1.1 Description**

Creates a mail server connection for a named application.

WebCenter Portal and Portal Framework applications support the Microsoft Exchange Server or any mail server that supports IMAP4 and SMTP. The most important mail server connection attributes are: `imapHost`, `imapPort`, `imapSecured`, `smtpHost`, `smtpPort`, and `smtpSecured`

You can register multiple mail server connections:

- **WebCenter Portal** supports multiple mail connections. The mail connection configured with `default=1` is the default connection for mail services in WebCenter Portal. All additional connections are offered as alternatives; WebCenter Portal users can choose which one they want to use through user preferences.

- **Portal Framework applications** only use one mail connection—the connection configured with `default=1`. Any additional connections are ignored.
## 10.12.1.2 Syntax

```java
createMailConnection(appName, name, [imapHost, imapPort, smtpHost, smtpPort, imapSecured, smtpSecured, appId, timeout, default, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td><code>name</code></td>
<td>Connection name. The name must be unique (across all connection types) within the application.</td>
</tr>
<tr>
<td><code>imapHost</code></td>
<td>Optional. Host name of the machine on which the IMAP service is running.</td>
</tr>
<tr>
<td><code>imapPort</code></td>
<td>Optional. Port on which the IMAP service listens.</td>
</tr>
<tr>
<td><code>smtpHost</code></td>
<td>Optional. Host name of the machine where the SMTP service is running.</td>
</tr>
<tr>
<td><code>smtpPort</code></td>
<td>Optional. Port on which the SMTP service listens.</td>
</tr>
<tr>
<td><code>imapSecured</code></td>
<td>Optional. Specifies whether the mail server connection to the IMAP server is SSL-enabled. Valid values are 1 (true) and 0 (false). The default for this argument is 0.</td>
</tr>
<tr>
<td><code>smtpSecured</code></td>
<td>Optional. Specifies whether the SMTP server is secured. Valid values are 1 (true) and 0 (false). The default for this argument is 0.</td>
</tr>
<tr>
<td><code>appId</code></td>
<td>External application associated with the mail server connection. External application credential information is used to authenticate users against the IMAP and SMTP servers. The same credentials are supplied to authenticate the user on both the IMAP and SMTP servers. The external application you configure for mail must use <code>authMethod=POST</code>, and specify several additional login fields: <code>fieldName='Email Address' and displayToUser=1</code> <code>fieldName='Your Name' and displayToUser=1</code> <code>fieldName='Reply-To Address' and displayToUser=1</code> If an external application does not exist yet, use the WLST command <code>createMailExtApp</code> to create an external application that automatically has all the required additional fields. See also <code>createExtAppConnection</code>.</td>
</tr>
<tr>
<td><code>timeout</code></td>
<td>Optional. Length of time (in seconds) that the mail waits to acquire a connection before terminating. This argument defaults to -1. When set to -1, the service default (10 seconds) applies.</td>
</tr>
</tbody>
</table>
The following example creates an external application suitable for a mail server connection, and then creates a mail server connection named `myMailConnection` for WebCenter Portal (`webcenter`):

```
<weblogic/serverConfig>
createMailExtApp(appName='webcenter', name='extApp_Mail', displayName='Mail Ext App')
<weblogic/serverConfig>
```

```
<weblogic/serverConfig>
createMailConnection(appName='webcenter', name='myMailConnection', imapHost='myimaphost.com', imapPort=143, smtpHost='mysmtphost.com', smtpPort=25, imapSecured=0, smtpSecured=0, appId='extApp_Mail', timeout=60, default=1)
<weblogic/serverConfig>
```

**10.12.2 setMailConnection**

Module: Oracle WebCenter Portal

Use with WLST: Online

**10.12.2.1 Description**

Edits an existing mail connection. Use this command to update connection attributes.

The connection is created using the `createMailConnection` command.

(WebCenter Portal only.) This command enables you to set additional, optional, LDAP server attributes that cannot be set using `createMailConnection`. When LDAP details are defined, mail creates, edits, and deletes portal distribution lists in WebCenter Portal. Portal distribution lists are named after their portal (excluding non-java identifiers) and assigned a domain (derived from the `domain` attribute, for example, `@mycompany.com`). If LDAP details are not provided, portal distribution lists are not created or maintained. The mail server must be a Microsoft Exchange Server.

**10.12.2.2 Syntax**

```
setMailConnection(appName, name, [imapHost, imapPort, smtpHost, smtpPort, imapSecured, smtpSecured, appId, default, ldapHost, ldapPort, ldapBaseDN, ldapAdminUser, ldapAdminPassword, ldapSecured, domain, defaultUser, timeout,
```
Mail

```
server, applicationVersion}
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>appName</strong></td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <em>webcenter</em>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td><strong>name</strong></td>
<td>Name of an existing mail server connection.</td>
</tr>
<tr>
<td><strong>imapHost</strong></td>
<td>Optional. Host name of the machine on which the IMAP service is running.</td>
</tr>
<tr>
<td><strong>imapPort</strong></td>
<td>Optional. Port on which the IMAP service listens.</td>
</tr>
<tr>
<td><strong>smtpHost</strong></td>
<td>Optional. Host name of the machine where the SMTP service is running.</td>
</tr>
<tr>
<td><strong>smtpPort</strong></td>
<td>Optional. Port on which the SMTP service listens.</td>
</tr>
<tr>
<td><strong>imapSecured</strong></td>
<td>Optional. Specifies whether the connection to the IMAP server is secured (SSL-enabled). Valid values are 1 (true) and 0 (false). The default for this argument is 0.</td>
</tr>
<tr>
<td><strong>smtpSecured</strong></td>
<td>Optional. Specifies whether the connection to the SMTP server is secured (SSL-enabled). Valid values are 1 (true) and 0 (false). The default for this argument is 0.</td>
</tr>
<tr>
<td><strong>appId</strong></td>
<td>Optional. External application associated with the mail server connection. External application credential information is used to authenticate users against the IMAP and SMTP servers. The same credentials are supplied to authenticate the user on both the IMAP and SMTP servers. The external application you configure for mail must use <code>authMethod=POST</code>, and specify several additional login fields: <code>fieldName='Email Address' and displaytoUser=1</code> <code>fieldName='Your Name' and displaytoUser=1</code> <code>fieldName='Reply-To Address' and displaytoUser=1</code> If an external application does not exist yet, use the WLST command <code>createMailExtApp</code> to create an external application that automatically has all the required additional fields. See also <code>createExtAppConnection</code>.</td>
</tr>
<tr>
<td><strong>ldapHost</strong></td>
<td>Optional. Host name of the machine where the LDAP directory server is running.</td>
</tr>
<tr>
<td><strong>ldapPort</strong></td>
<td>Optional. Port on which the LDAP directory server listens.</td>
</tr>
<tr>
<td><strong>ldapBaseDN</strong></td>
<td>Optional. Base distinguished name for the LDAP schema. For example, <em>CN=Users,DC=oracle,DC=com</em>.</td>
</tr>
<tr>
<td><strong>ldapAdminUser</strong></td>
<td>Optional. User name of the LDAP directory server administrator. A valid administrator with privileges to make entries into the LDAP schema.</td>
</tr>
<tr>
<td><strong>ldapAdminPassword</strong></td>
<td>Optional. Password for the LDAP directory server administrator. This password will be stored in a secured store.</td>
</tr>
<tr>
<td><strong>ldapSecured</strong></td>
<td>Optional. Specifies whether the connection to the LDAP server is secured (SSL enabled). Valid values are 1 (true) and 0 (false). The default for this argument is 0. Set this to 1 for all LDAP communications over SSL.</td>
</tr>
</tbody>
</table>
Mail

WebCenter Portal Custom WLST Commands

Examples

The following example sets individual attributes for a mail server connection configured for WebCenter Portal (webcenter):

```
wlst:/weblogic/serverConfig> setMailConnection(appName='webcenter',
name='myMailConnection', imapHost='myimaphost.com', imapPort=143,
smtpHost='mysmtphost.com', smtpPort=25, imapSecured=0, smtpSecured=0,
appId='extApp_Mail', timeout=60, default=1)
```

The following example sets individual attributes for a mail server connection:

```
wlst:/weblogic/serverConfig> setMailConnection(appName='webcenter',
name='myMailConnection', imapPort=993, imapSecured=1, smtpPort=465,
smtpSecured=1)
```

The following example sets LDAP attributes for a mail server connection:

```
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>domain</td>
<td>Optional. Domain name appended to portal distribution lists.</td>
</tr>
<tr>
<td></td>
<td>For example, if the domain attribute is set to mycompany.com, the Finance Project portal will maintain a distribution list named <a href="mailto:FinanceProject@mycompany.com">FinanceProject@mycompany.com</a>.</td>
</tr>
<tr>
<td>defaultUser</td>
<td>Optional. Comma-delimited list of user names to whom you want to grant moderation capabilities. These users become members of every portal distribution list that is created. The users specified must exist in the Base LDAP schema (specified in the ldapBaseDN argument).</td>
</tr>
<tr>
<td>timeout</td>
<td>Optional. Length of time (in seconds) that mail waits to acquire a connection before terminating. This argument defaults to -1. When set to -1, the service default (10 seconds) applies.</td>
</tr>
<tr>
<td>default</td>
<td>Optional. Indicates whether this connection is the default (or active) connection for mail. Valid values are 1 (true) and 0 (false). This argument defaults to 0. 1 specifies that this connection is the default connection for mail.</td>
</tr>
<tr>
<td></td>
<td>■ WebCenter Portal supports multiple mail connections. The mail connection configured with default=1 is the default connection for mail services in WebCenter Portal. Additional connections, configured with default=0, are offered as alternatives; WebCenter Portal users can choose which one they want to use through user preferences.</td>
</tr>
<tr>
<td></td>
<td>■ Portal Framework application only use one mail connection—the connection configured with default=1. Any additional connections are ignored.</td>
</tr>
<tr>
<td></td>
<td>A connection does not cease to be the default connection for mail if you change the default value from 0 to 1.</td>
</tr>
<tr>
<td></td>
<td>To stop using a default connection, use the removeMailServiceProperty command as follows:</td>
</tr>
<tr>
<td></td>
<td>removeMailServiceProperty('appName='webcenter', property='selected.connection')</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.12.3 setMailConnectionProperty

Module: Oracle WebCenter Portal

Use with WLST: Online

10.12.3.1 Description

Sets a mail server connection property. Use this command if additional parameters are required to connect to your mail server. This is an extensible way to add any connection property using a key and a value. (You are not limited to connection properties specified by createMailConnection and setMailConnection.)

All known, additional connection properties are listed in Table 10–20, "Additional Mail Connection Properties".

<table>
<thead>
<tr>
<th>Additional Connection Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>charset</td>
<td>Characterset used on the connection. The default charset is UTF-8. To use a different characterset, such as ISO-8859-1, set the charset connection property.</td>
</tr>
<tr>
<td>Various IMAP properties</td>
<td>Any valid IMAP connection property. For example, mail.imap.connectionpoolsize. A list of valid IMAP properties are listed in documentation for the com.sun.mail.imap package at: <a href="http://javamail.java.net/nonav/docs/api">http://javamail.java.net/nonav/docs/api</a></td>
</tr>
<tr>
<td>Various SMTP properties</td>
<td>Any valid SMTP connection property. For example, mail.smtp.timeout. A list of valid SMTP properties are listed in the documentation for the com.sun.mail.smtp package at: <a href="http://javamail.java.net/nonav/docs/api">http://javamail.java.net/nonav/docs/api</a></td>
</tr>
</tbody>
</table>

**Note:** Do not use the setMailConnectionProperty to set connection properties available through createMailConnection or setMailConnection. Attempting to do so, has no effect.

10.12.3.2 Syntax

setMailConnectionProperty(appName, name, key, value, [secure, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
</tbody>
</table>
Example

The following example adds a custom mail server connection property called `myProperty1` with a default value `propertyValue1`:

```
wlst:/weblogic/serverConfig> setMailConnectionProperty(appName='webcenter',
name='myMailServer', key='myProperty1', value='propertyValue1')
```

### 10.12.4 deleteMailConnectionProperty

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.12.4.1 Description

Deletes a mail server connection property. Take care when deleting connection properties because the connection may not work as expected if the configuration becomes invalid as a result.

This command can only delete additional connection properties added using the `setMailConnectionProperty` command.

#### 10.12.4.2 Syntax

```
deleteMailConnectionProperty(appName, name, key, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>appName</strong></td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td><strong>name</strong></td>
<td>Name of an existing mail server connection.</td>
</tr>
<tr>
<td><strong>key</strong></td>
<td>Name of the connection property you want to delete.</td>
</tr>
<tr>
<td><strong>server</strong></td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, <code>WC_Spaces</code>.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
</tbody>
</table>
Example
The following example deletes a mail server connection property named mailProperty1:

```
wls:/weblogic/serverConfig>deleteMailConnectionProperty(appName='webcenter', name='myMailServer', key='mailProperty1')
```

### 10.12.5 listMailConnections

**Module:** Oracle WebCenter Portal

**Use with WLST:** Online

#### 10.12.5.1 Description

Lists all of the mail server connections that are configured for a named application.

#### 10.12.5.2 Syntax

```
listMailConnections(appName, [verbose, name, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Displays mail server connection details in verbose mode.</td>
</tr>
<tr>
<td></td>
<td>Valid options are 1 (true) and 0 (false).</td>
</tr>
<tr>
<td></td>
<td>When set to 1, listMailConnections lists all of the mail server connections</td>
</tr>
<tr>
<td></td>
<td>that are configured for an application, along with their details.</td>
</tr>
<tr>
<td></td>
<td>When set to 0, only connection names are listed. This argument defaults to 0.</td>
</tr>
<tr>
<td>name</td>
<td>Optional. Name of an existing mail server connection. Use this argument to</td>
</tr>
<tr>
<td></td>
<td>view connection details for a specific mail server connection.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, <code>WC_Spaces</code>.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different</td>
</tr>
<tr>
<td></td>
<td>servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if</td>
</tr>
<tr>
<td></td>
<td>more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

#### 10.12.5.3 Example

The following example lists the names of mail server connections that are currently configured for WebCenter Portal:

```
wls:/weblogic/serverConfig>listMailConnections(appName='webcenter')
```

The following example lists connection names and details for all of the mail server connections that are currently configured for WebCenter Portal:
The following example lists connection details for a mail server connection named mailConnection1:

```bash
ws:/weblogic/serverConfig>listMailConnections(appName='webcenter', verbose=1)
```

### 10.12.6 listDefaultMailConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.12.6.1 Description

Lists the default mail server connection that mail is using in a named application.

You can register multiple mail server connections but there can only be one default connection:

- **WebCenter Portal** supports multiple mail connections. The mail connection configured with `default=1` is the default connection for mail services in WebCenter Portal. All additional connections are offered as alternatives; WebCenter Portal users can choose which one they want to use through user preferences.

- **Portal Framework applications** only use one mail connection—the connection configured with `default=1`. Any additional connections are ignored.

#### 10.12.6.2 Syntax

```bash
listDefaultMailConnection(appName,[verbose, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Displays the default mail server connection in verbose mode, if</td>
</tr>
<tr>
<td></td>
<td>available. Valid options are 1 (true) and 0 (false). When set to 1, the</td>
</tr>
<tr>
<td></td>
<td>name and details of the mail server connection are listed.</td>
</tr>
<tr>
<td></td>
<td>When set to 0, only the connection name displays. This argument defaults to</td>
</tr>
<tr>
<td></td>
<td>0.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, <code>WC_Spaces</code>.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different</td>
</tr>
<tr>
<td></td>
<td>servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more</td>
</tr>
<tr>
<td></td>
<td>than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

#### 10.12.6.3 Example

The following example lists the name and details of the mail server connection that mail is using in WebCenter Portal:

```bash
ws:/weblogic/serverConfig>listDefaultMailConnection(appName='webcenter',
```
10.12.7 setDefaultMailConnection

Module: Oracle WebCenter Portal
Use with WLST: Online

10.12.7.1 Description
Specifies the default mail server connection for mail in a named application.

You can register multiple mail server connections but there can only be one default connection:

- **WebCenter Portal** supports multiple mail connections. The mail connection configured with `default=1` is the default connection for mail services in WebCenter Portal. All additional connections are offered as alternatives; WebCenter Portal users can choose which one they want to use through user preferences.

- **Portal Framework applications** only use one mail connection—the connection configured with `default=1`. Any additional connections are ignored.

10.12.7.2 Syntax
```
setDefaultMailConnection(appName, name, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing mail connection.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.12.7.3 Example
The following example configures mail to use a connection named `myMailServer` for the default connection in WebCenter Portal:

```
wls:/weblogic/serverConfig>setDefaultMailConnection(appName='webcenter', name='myMailServer')
```

10.12.8 setMailServiceProperty

Module: Oracle WebCenter Portal
Use with WLST: Online

10.12.8.1 Description
Specifies default values for mail.
Configurable properties for mail are listed in Table 10–21, "Mail - Configurable Properties".

<table>
<thead>
<tr>
<th>Configuration Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address.delimiter</td>
<td>Defines the delimiter that is used to separate multiple mail addresses. A comma is used by default.</td>
</tr>
<tr>
<td></td>
<td>Some mail servers require mail addresses in the form lastname, firstname and, in such cases, a semi-colon is required.</td>
</tr>
<tr>
<td>mail.emailgateway.polling.frequency</td>
<td>The frequency, in seconds, that portal distribution lists are checked for new incoming emails. The default is 1800 seconds (30 minutes).</td>
</tr>
<tr>
<td></td>
<td>Email communication through portal distribution lists can be published as discussion forum posts on a discussions server. For details, see &quot;Publishing Portal Mail in a Discussion Forum&quot; in Oracle Fusion Middleware Building Portals with Oracle WebCenter Portal.</td>
</tr>
<tr>
<td>mail.messages.fetch.size</td>
<td>Maximum number of messages displayed in mail inboxes.</td>
</tr>
<tr>
<td>resolve.email.address.to.name</td>
<td>Determines whether user email addresses are resolved to WebCenter Portal user names when LDAP is configured. Valid values are 1 (true) and 0 (false). The default value is 0.</td>
</tr>
<tr>
<td></td>
<td>When set to 1, WebCenter Portal user names display instead of email addresses in mail task flows.</td>
</tr>
<tr>
<td></td>
<td>Set this property to 1 if the instant messaging and presence requires user names to obtain presence status because presence information cannot be obtained when mail provides email addresses. Setting this value to 1 does impact application performance so you must take this into consideration when setting this property.</td>
</tr>
<tr>
<td>mail.recipient.limit</td>
<td>Restricts the number of recipients to a message. For example, setting this value to '500' limits the number of recipients to 500.</td>
</tr>
</tbody>
</table>

10.12.8.2 Syntax

setMailServiceProperty(appName, property, value, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>property</td>
<td>Name of the configuration property</td>
</tr>
<tr>
<td>value</td>
<td>Value for the property.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.12.8.3 Example
The following example increases the default number of messages displayed in mail inboxes to 100, in WebCenter Portal (webcenter):

```
WLS:/weblogic/serverConfig> setMailServiceProperty(appName='webcenter',
property='mail.messages.fetch.size', value='100')
```

10.12.9 removeMailServiceProperty

Module: Oracle WebCenter Portal
Use with WLST: Online

10.12.9.1 Description
Removes the current value that is set for a mail property. Use this command to remove any of the properties listed in Table 10–21, "Mail - Configurable Properties".

Take care when using this command as removing values for these properties might cause unexpected behavior.

**Note:** Use this command syntax to stop mail from using the current default connection:

```
removeMailServiceProperty('appName='webcenter',
property='selected.connection')
```

This command forces the default connection argument to 0. See also, setMailConnection.

10.12.9.2 Syntax
```
removeMailServiceProperty(appName, property, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td>property</td>
<td>Name of the configuration property.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more</td>
</tr>
</tbody>
</table>

10.12.9.3 Example
The following example clears the current mail.messages.fetch.size setting for mail, in WebCenter Portal:

```
WLS:/weblogic/serverConfig> removeMailServiceProperty(appName='webcenter',
property='mail.messages.fetch.size')
```
10.12.10 listMailServiceProperties

Module: Oracle WebCenter Portal
Use with WLST: Online

10.12.10.1 Description
Lists all configurable properties for mail.

10.12.10.2 Syntax
listMailServiceProperties(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.12.10.3 Example
The following example lists configurable properties for mail in WebCenter Portal:
wls:/weblogic/serverConfig> listMailServiceProperties(appName='webcenter')

10.12.11 createMailExtApp

Module: Oracle WebCenter Portal
Use with WLST: Online

10.12.11.1 Description
Creates an external application suitable for mail server connections. The external application is configured with all the required additional properties: authMethod=POST, and specify several additional login fields:

fieldName='Email Address' and displaytoUser=1
fieldName='Your Name' and displaytoUser=1
fieldName='Reply-To Address' and displaytoUser=1

10.12.11.2 Syntax
createMailExtAppConnection(appName, name, [displayName, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
</tbody>
</table>
The following example creates an external application named MailxApp suitable for mail server connections.

```
wlst:/weblogic/serverConfig> createMailExtAppConnection(appName='webcenter',
name='MailxApp', displayName='Mail Ext App')
```

### 10.13 Migration

Use the commands listed in **Table 10–22** when migrating a Portal Framework application to WebCenter Portal 11.1.1.9.0.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>migrateFrameworkApplicationConnections</td>
<td>Migrate connections of a Portal Framework application to WebCenter Portal.</td>
<td>Online</td>
</tr>
<tr>
<td>migrateFrameworkApplicationToWebCenter</td>
<td>Migrate a Portal Framework application to a portal in WebCenter Portal.</td>
<td>Online</td>
</tr>
<tr>
<td>migrateFrameworkApplicationDocumentsToWebCenter</td>
<td>Migrate documents and metadata of a Portal Framework application to WebCenter Portal.</td>
<td>Online</td>
</tr>
<tr>
<td>migrateFrameworkApplicationDiscussionsToWebCenter</td>
<td>Updates discussion IDs used in the migrated portal pages with new discussion IDs provisioned for the portal.</td>
<td>Online</td>
</tr>
</tbody>
</table>

#### 10.13.1 migrateFrameworkApplicationConnections

Module: Oracle WebCenter Portal

Use with WLST: Online

**10.13.1.1 Description**

Migrates connections of a Portal Framework application to WebCenter Portal.

**10.13.1.2 Syntax**

```
migrateFrameworkApplicationConnections(appName, fwkAppName, server, fwkAppServer, |
```
migrationDirectory, [spacesRoot, securityGroup, primaryConnectionName, adminUserName, applicationVersion])

**Note:** If the Portal Framework application contains any Content Server connections, you must provide values for the spacesRoot, securityGroup, primaryConnectionName and adminUserName arguments.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the WebCenter Portal application into which you want to migrate your Portal Framework application. For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>fwkAppName</td>
<td>Name of the Portal Framework application whose connections you want to migrate.</td>
</tr>
<tr>
<td>server</td>
<td>Name of the managed server where the WebCenter Portal application is deployed. For example, WC_Spaces. In a clustered environment where applications share the same Content Server connection (that is, applications connect to the same Content Server and share the same root folder and security group) run this command against only one managed server.</td>
</tr>
<tr>
<td>fwkAppServer</td>
<td>Name of the managed server where the Portal Framework application is deployed. For example, WC_CustomPortal.</td>
</tr>
<tr>
<td>migrationDirectory</td>
<td>Path to the folder where migration logs and data related to migration are stored. In case of a multi-node setup, this must be a shared directory that can be accessed and written onto from all nodes. Also, the directory must be accessible using the same path from all nodes.</td>
</tr>
<tr>
<td>spacesRoot</td>
<td>Optional. Root folder under which WebCenter Portal content is stored. The value for this argument must use the format /foldername. For example, /WebCenter or /WebCenterSpaces. The spacesRoot value cannot be / (the root itself) and it must be unique across applications. If the specified folder does not exist, it gets created.</td>
</tr>
<tr>
<td>securityGroup</td>
<td>Optional. Identifies the documents related to WebCenter Portal. A security group is used to separate data when multiple WebCenter Portal applications share the same Content Server repository. This name must be unique across applications. The value for this argument must begin with an alphabetical character, followed by any combination of alphanumeric characters or an underscore. The string must be less than or equal to 30 characters.</td>
</tr>
<tr>
<td>primaryConnectionName</td>
<td>Optional.  Name of the Content Server connection in your Portal Framework application that will be set as the default connection in WebCenter Portal.</td>
</tr>
<tr>
<td>adminUserName</td>
<td>Optional. User name of the Content Server repository administrator. For example, sysadmin.</td>
</tr>
</tbody>
</table>
10.13.1.3 Example
The following example migrates connections from the Portal Framework application FinanceApplication deployed on the WC_CustomPortal managed server to the WebCenter Portal application deployed on the WC_Spaces managed server. Migration logs are stored under the folder /scratch/migrate.

```
wlsc:/weblogic/serverConfig>migrateFrameworkApplicationConnections(appName='webcenter',fwkAppName='FinanceApplication',server='WC_Spaces',fwkAppServer='WC_CustomPortal',migrationDirectory='/scratch/migrate')
```

The following example migrates connections from the Portal Framework application HRApplication deployed on WC_CustomPortal to the WebCenter Portal application deployed on WC_Spaces. After the connections are migrated, ucm-conn is set as the default Content Server connection for WebCenter Portal. Migration logs are stored under the folder /scratch/migrate. The Portal Framework application contains a Content Server connection, and therefore the values for the spacesRoot, securityGroup, primaryConnectionName and adminUserName arguments are provided.

```
wlsc:/weblogic/serverConfig>migrateFrameworkApplicationConnections(appName='webcenter',fwkAppName='HRApplication',server='WC_Spaces',fwkAppServer='WC_CustomPortal',migrationDirectory='/scratch/migrate',spacesRoot='/WebCenter',securityGroup='webcenter',primaryConnectionName='ucm-conn',adminUserName='weblogic')
```

10.13.2 migrateFrameworkApplicationToWebCenter
Module: Oracle WebCenter Portal
Use with WLST: Online

10.13.2.1 Description
Migrates a Portal Framework application to a portal in WebCenter Portal.

10.13.2.2 Syntax
migrateFoldersGDataToFrameworkFolders(appName, fwkAppName, server, fwkAppServer, adminUrl, adminUser, targetPortal, migrationDirectory, [pagePath, navigationFile, properties, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the WebCenter Portal application in which you are want to migrate your Portal Framework application as a portal. For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>fwkAppName</td>
<td>Name of the Portal Framework application that you want to migrate.</td>
</tr>
</tbody>
</table>
The following example migrates the Portal Framework application HRApplication to the portal HRPortal on the WebCenter Portal application deployed on the WC_Spaces managed server. The migration logs are stored under the folder /scratch/migrate.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>server</td>
<td>Name of the managed server where the WebCenter Portal application is deployed. For example, WC_Spaces. In a clustered environment where applications share the same Content Server connection (that is, applications connect to the same Content Server and share the same root folder and security group) run this command against only one managed server.</td>
</tr>
<tr>
<td>fwkAppServer</td>
<td>Name of the managed server where the Portal Framework application is deployed. For example, WC_CustomPortal.</td>
</tr>
<tr>
<td>adminUrl</td>
<td>Connection URL for the WebLogic domain's admin server. The connection URL format is t3://host:port.</td>
</tr>
<tr>
<td>adminUser</td>
<td>User name of the WebLogic Server administrator.</td>
</tr>
<tr>
<td>targetPortal</td>
<td>Name of the portal in the WebCenter Portal application where the Portal Framework application has to be migrated.</td>
</tr>
<tr>
<td>migrationDirectory</td>
<td>Absolute directory path (with the write permission) where logs and data related to migration are written. In case of a multi-node setup, this must be a shared directory that can be accessed and written onto from all nodes. Also, the directory must be accessible using the same path from all nodes. While migrating your Portal Framework application, you can choose to migrate static resources by providing a list of old and new resource mappings in the migratedResources.properties file. You need to manually create this file and place it under the resources folder in the location specified in the migrationDirectory argument. Alternatively, you can manually deploy static resources as a shared library on the portal server after migrating your application.</td>
</tr>
<tr>
<td>pagePath</td>
<td>Optional. List of custom directory paths from which pages of your Portal Framework application need to be migrated. By default, the pages created at design time are stored at the path: \oracle\webcenter\portalapp\pages. The pages created at runtime are stored at \oracle\webcenter\portalapp\pagehierarchy. If your application pages are stored at other locations, specify the comma separated list of the paths. For example, /oracle/webcenter/portalapp/mypages/oracle/webcenter/portalapp/salespages.</td>
</tr>
<tr>
<td>navigationFile</td>
<td>Optional. The navigation file to be used to build target portal's navigation. If not specified, the default navigation configured for your Portal Framework application is used.</td>
</tr>
<tr>
<td>Properties</td>
<td>Optional. Comma separated list of properties specified as key-value pairs that you want to pass while running the command. The format is key1=value1,key2=value1;value2.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the WebCenter Portal application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
The following example migrates the Portal Framework application FinanceApplication to the portal FinancePortal in the WebCenter Portal application deployed on the WC_Spaces managed server. During migration, the navigation file custom-navigation-model.xml is used to build the navigation for FinancePortal. Migration logs are stored under the folder /scratch/migrate.

```
```

### 10.13.3 `migrateFrameworkApplicationDocumentsToWebCenter`

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.13.3.1 Description

Migrates Portal Framework application documents to WebCenter Portal Server and updates the security metadata as per the target portal.

---

**Note:** The following are the prerequisites for running the `migrateFrameworkApplicationDocumentsToWebCenter` command:

- This command can be run only after migrating a Portal Framework application to WebCenter Portal.
- The Documents service must be provisioned for the target portal.
- The Portal Framework application’s contents must be moved manually to the target portal’s folder using the WebCenter Content Server user interface.

---

#### 10.13.3.2 Syntax

`migrateFrameworkApplicationDocumentsToWebCenter(appName, server, targetPortal, migrationFolderMappingFile, migrationDirectory, [connectionName, applicationVersion])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the WebCenter Portal application that contains the portal where you migrated your Portal Framework application. The WebCenter Portal application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td><code>targetPortal</code></td>
<td>Name of the portal where the Portal Framework application has been migrated.</td>
</tr>
</tbody>
</table>
### 10.13.3 Example

The following example migrates the documents of a Portal Framework application to the target portal named HR Portal in the WebCenter Portal application deployed on the WC_Spaces managed server. The default Content Server connection is used. The MigrationMappingFile.csv file is used to update the folder path references and security metadata for the documents in the target portal. Migration logs are stored in the /tmp/migrate folder.

```wls:
/serverConfig>migrateFrameworkApplicationDocumentsToWebCenter(appName = 'webcenter', server = 'WC_Spaces', targetPortal = 'HR Portal', migrationFolderMappingFile = '/tmp/migrationmap/MigrationMappingFile.csv', migrationDirectory = '/tmp/migrate')
```

### 10.13.4 migrateFrameworkApplicationDiscussionsToWebCenter

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.13.4.1 Description

Updates announcement and discussion IDs used in the migrated portal pages with the new IDs provisioned for the target portal.
10.13.4.2 Syntax

migrateFrameworkApplicationDiscussionsToWebCenter(appName, server, targetPortal, discussionsMappingFile, migrationDirectory, [applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the WebCenter Portal application where you migrated your Portal Framework application. For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>server</td>
<td>Name of the managed server where the WebCenter Portal application is deployed. For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>In a clustered environment where applications share the same Content Server connection (that is, the applications connect to the same Content Server and share the same root folder and security group) run this command against only one managed server.</td>
</tr>
<tr>
<td>targetPortal</td>
<td>Name of the portal where the Portal Framework application has been migrated.</td>
</tr>
<tr>
<td>discussionsMappingFile</td>
<td>Comma separated values (CSV) file containing mapping entries for discussions and announcements migrated.</td>
</tr>
<tr>
<td></td>
<td>You must create this file manually. The values contain IDs of the forums or categories before and after migration on WebCenter Discussion Server.</td>
</tr>
<tr>
<td>migrationDirectory</td>
<td>Path to the folder where logs and data related to migration are stored.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.13.4.3 Example

The following example migrates a Portal Framework application’s discussions and announcements to the category or forum provisioned for the HR Portal portal in the WebCenter Portal application deployed on the WC_Spaces managed server. The DiscussionsMappingFile.csv file is used to map the old forum and category IDs with the new ones.

wls:/weblogic/serverConfig>migrateFrameworkApplicationDiscussionsToWebCenter(appName='webcenter', server='WC_Spaces', targetPortal='HR Portal', discussionsMappingFile='~/tmp/DiscussionsMappingFile.csv', migrationDirectory='~/scratch/migrate')
10.14 Notifications

Use the commands listed in Table 10–23 to manage settings for the notifications in a named application.

Configuration changes made using these WLST commands are only effective after your restart the Managed Server on which the application is deployed. For details, see Oracle Fusion Middleware Administering Oracle WebCenter Portal.

Table 10–23 Notifications WLST Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>setNotificationsConfig</td>
<td>Specify the connection used for routing notifications raised in a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>getNotificationsConfig</td>
<td>Return details about the connection that is used to send notifications raised in a named application.</td>
<td>Online</td>
</tr>
</tbody>
</table>

10.14.1 setNotificationsConfig

Module: Oracle WebCenter Portal

Use with WLST: Online

10.14.1.1 Description

Specifies the connection used for routing notifications raised in a named application.

Use an existing mail server or BPEL server connection. If the application is connected to a BPEL server, the Oracle User Messaging Service (UMS) is available for routing notifications through multiple messaging channels, including mail, worklists, and SMS. If you configure notifications to use a BPEL server connection, you may specify a sender 'From' address for each available messaging channel. That is, you can specify a sender mail address and an SMS address.

Alternatively, you can route notification messages through a mail server. If you configure notifications to use a mail server connection, the external application associated with the mail server connection must contain shared credentials. Shared credentials are required for routing application-wide notifications.

10.14.1.2 Syntax

setNotificationsConfig(appName, type, name, [senderMailAddress, senderSMSAddress, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>type</td>
<td>Type of connection used to send notifications. Valid values are MAIL and BPEL.</td>
</tr>
</tbody>
</table>
Example

The following example specifies that notifications in WebCenter Portal (webcenter) use a BPEL server connection named WebCenter-Worklist and also defines the mail address and SMS address from which all notifications are sent:

```
<wls:weblogic/serverConfig>setNotificationsConfig(appName='webcenter',
  type='BPEL',
  name='WebCenter-Worklist',
  senderMailAddress='WebCenter
Notification<notifications@webcenter.com',
  senderSMSAddress='28734')
```

10.14.2 getNotificationsConfig

Module: Oracle WebCenter Portal

Use with WLST: Online

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of an existing connection.</td>
</tr>
<tr>
<td>senderMailAddress</td>
<td>Optional. Mail address from which all mail notifications are sent.</td>
</tr>
<tr>
<td>senderSMSAddress</td>
<td>Optional. SMS number from which all SMS notifications are sent.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.14.2.1 Description
Returns details about the connection that is used to send notifications raised in a named application.

10.14.2.2 Syntax
getNotificationsConfig(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.14.2.3 Example
The following example returns details about the connection used by notifications service in WebCenter Portal:

```wls
wls:/weblogic/serverConfig>getNotificationsConfig(appName='webcenter')
```

ConnectionType: BPEL
ConnectionName: WebCenter-Worklist
SenderMailAddress: notifications@webcenter.com
SenderSMSAddress: 28776

10.15 People Connections

Use the commands listed in Table 10–24 to manage user profile information in WebCenter Portal or your Portal Framework application.

Configuration changes made using these WLST commands are only effective after your restart the Managed Server on which WebCenter Portal or your Portal Framework application is deployed. For details, see Oracle Fusion Middleware Administering Oracle WebCenter Portal.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>startSyncProfiles</td>
<td>Synchronize profile information in the LDAP store, with the WebCenter Portal database schema.</td>
<td>Online</td>
</tr>
<tr>
<td>syncProfile</td>
<td>Synchronize profile information for a specific user.</td>
<td>Online</td>
</tr>
<tr>
<td>stopSyncProfiles</td>
<td>Stop the profile synchronization process.</td>
<td>Online</td>
</tr>
<tr>
<td>isSyncProfilesRunning</td>
<td>Check whether profile synchronization is in progress.</td>
<td>Online</td>
</tr>
<tr>
<td>setProfileConfig</td>
<td>Set profile properties.</td>
<td>Online</td>
</tr>
</tbody>
</table>
10.15.1 startSyncProfiles
Module: Oracle WebCenter Portal
Use with WLST: Online

10.15.1.1 Description
Synchronizes profile information in the LDAP store with the WebCenter Portal database schema.

10.15.1.2 Syntax
startSyncProfiles(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.15.1.3 Example
The following example synchronizes user profiles for WebCenter Portal:

```
wls:/weblogic/serverConfig>startSyncProfiles(appName='webcenter')
```
10.15.2.1 Description
Stops the profile synchronization process, if currently in progress.

10.15.2.2 Syntax
stopSyncProfiles(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.15.2.3 Example
The following example stops the profile synchronization process for WebCenter Portal:

```
wlsh/weblogic/serverConfig>stopSyncProfiles(appName='webcenter')
```

10.15.3 isSyncProfilesRunning
Module: Oracle WebCenter Portal
Use with WLST: Online

10.15.3.1 Description
Checks whether profile synchronization is in progress.

10.15.3.2 Syntax
isSyncProfilesRunning(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.15.3.3 Example
The following example checks whether profile synchronization is in progress for WebCenter Portal:

```
```
10.15.4 syncProfile

Module: Oracle WebCenter Portal
Use with WLST: Online

10.15.4.1 Description
Synchronizes profile information for a specific user in the LDAP store with the WebCenter Portal schema.

10.15.4.2 Syntax
syncProfile(appName, userName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>userName</td>
<td>Name of the user whose profile information you want to synchronize.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.15.4.3 Example
The following example synchronizes profile information for a user named monty:
wls:/weblogic/serverConfig>syncProfile(appName='webcenter', userName='monty')

10.15.5 setProfileConfig

Module: Oracle WebCenter Portal
Use with WLST: Online

10.15.5.1 Description
Sets profile properties for a named application. If you omit a parameter, the corresponding configuration remains unchanged.

10.15.5.2 Syntax
setProfileConfig(appName, ProfilePageVersion, ProfileCacheNumberOfObjects, ProfileCacheTimeToLive, ProfileSyncLDAPReadBatchSize, ProfileSyncHourOfDay, ProfileSyncFrequencyInDays, server, applicationVersion)
### Example

The following example sets new values for the `ProfileCacheNumberOfObjects` and `ProfileCacheTimeToLive` configuration settings.

All other settings remain unchanged:

```
wls:/weblogic/serverConfig>setProfileConfig(appName='webcenter',ProfileCacheNumberOfObjects=2000,ProfileCacheTimeToLive=120)
```

### 10.15.6 `getProfileConfig`

**Module:** Oracle WebCenter Portal  
**Use with WLST:** Online

#### 10.15.6.1 Description

Returns current profile settings for a named application.
10.15.6.2 Syntax

getProfileConfig(appName, key, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>key</td>
<td>Name of a profile property. Valid values include:</td>
</tr>
<tr>
<td></td>
<td>- ProfilePageVersion</td>
</tr>
<tr>
<td></td>
<td>- ProfileCacheNumberOfObjects</td>
</tr>
<tr>
<td></td>
<td>- ProfileCacheTimeToLive</td>
</tr>
<tr>
<td></td>
<td>- ProfileSyncLDAPReadBatchSize</td>
</tr>
<tr>
<td></td>
<td>- ProfileSyncHourOfDay</td>
</tr>
<tr>
<td></td>
<td>- ProfileSyncFrequencyInDays</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.15.6.3 Example

The following examples return current settings for various profile properties:

wls:/weblogic/serverConfig> getProfileConfig(appName='webcenter', key='ProfilePageVersion')
wls:/weblogic/serverConfig> getProfileConfig(appName='webcenter', key='ProfileCacheNumberOfObjects')
wls:/weblogic/serverConfig> getProfileConfig(appName='webcenter', key='ProfileCacheTimeToLive')
wls:/weblogic/serverConfig> getProfileConfig(appName='webcenter', key='ProfileSyncLDAPReadBatchSize')
wls:/weblogic/serverConfig> getProfileConfig(appName='webcenter', key='ProfileSyncHourOfDay')
wls:/weblogic/serverConfig> getProfileConfig(appName='webcenter', key='ProfileSyncFrequencyInDays')

10.15.7 listProfileConfig

Module: Oracle WebCenter Portal
Use with WLST: Online

10.15.7.1 Description

Lists the current profile settings for a named application.

10.15.7.2 Syntax

listProfileConfig(appName, [server, applicationVersion])
The following example lists current profile settings for WebCenter Portal (webcenter):

```
  wls:/weblogic/serverConfig>listProfileConfig(appName='webcenter')
```

### 10.15.8 `setProfileCacheNumberOfObjects`

**Module:** Oracle WebCenter Portal  
**Use with WLST:** Online  

#### 10.15.8.1 Description

(Deprecated from 11.1.1.8.0. Use `setProfileConfig` instead.)  
Sets the maximum number of profile objects to cache (in the profile cache).

#### 10.15.8.2 Syntax

```
  setProfileCacheNumberOfObjects(appName, noOfObjects, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| For WebCenter Portal, the application name is always `webcenter`.  
| For Portal Framework applications, specify the appropriate name.  | |
| `noOfObjects`          | Number of profile objects to cache.                              |
|  
| Any value between 1 and 10,000. The default value is `1000`.  | |
| `server`               | Optional. Name of the managed server where the application is    |
|  
| deployed. For example, `WC_Spaces`.  
| Required when applications with the same name are deployed to  
| different servers and also when you have a cluster.  | |
| `applicationVersion`   | Optional. Version number of the deployed application. Required if  |
|  
| more than one version of the application is deployed.  | |

#### 10.15.8.3 Example

The following example increases the size of the cache to 2,000 profiles:

```
  wls:/weblogic/serverConfig>setProfileCacheNumberOfObjects(appName='webcenter', noOfObjects=2000)
```
10.15.9 setProfileSyncLDAPReadBatchSize

Module: Oracle WebCenter Portal

Use with WLST: Online

10.15.9.1 Description

(Deprecated from 11.1.1.8.0. Use setProfileConfig instead.)

Sets the profile synchronization LDAP batch read size.

10.15.9.2 Syntax

setProfileSyncLDAPReadBatchSize(appName, batchSize, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>batchSize</td>
<td>LDAP batch read size. Any value between 1 and 1000.</td>
</tr>
<tr>
<td></td>
<td>The default value is 1000.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different</td>
</tr>
<tr>
<td></td>
<td>servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more</td>
</tr>
<tr>
<td></td>
<td>than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.15.9.3 Example

The following example increases the batch size to 2000 LDAP profiles:

```
wlst:/weblogic/serverConfig>setProfileSyncLDAPReadBatchSize(appName='webcenter',
batchSize=800)
```

10.15.10 setProfileCacheTimeToLive

Module: Oracle WebCenter Portal

Use with WLST: Online

10.15.10.1 Description

(Deprecated from 11.1.1.8.0. Use setProfileConfig instead.)

Sets the time (in minutes) for a profile to live in the profile cache.

10.15.10.2 Syntax

setProfileCacheTimeToLive(appName, timeToLive, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
</tbody>
</table>
The following example decreases the length of time profile objects are cached to 30 minutes:

```wls/``weblogic/serverConfig``>setProfileCacheTimeToLive(appName='webcenter',
  timeToLive=30)
```

### 10.15.11 setProfilePhotoSync

**Module:** Oracle WebCenter Portal  
**Use with WLST:** Online

#### 10.15.11.1 Description

Specifies whether to synchronize the latest user profile photos in LDAP with WebCenter Portal's profile cache. When synchronization is enabled, profile photos are synchronized when the profile cache expires. When synchronization is disabled, profile photos are not synchronized with the WebCenter Portal's profile cache.

**Note:** The WLST command `setProfileCacheTimeToLive` defines the length of time profile objects live in the cache.

#### 10.15.11.2 Syntax

`setProfilePhotoSync(appName, enablePhotoSync, [server, applicationVersion])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
</tbody>
</table>
| `enablePhotoSync`| Specifies whether to periodically synchronize user profile photos in LDAP. Valid values are 1 and 0.  
  - 1 - Synchronize user profile photos periodically (after WebCenter Portal's profile cache expires).  
  - 0 - Profile photos in LDAP are not automatically synchronized with WebCenter Portal's profile cache. WebCenter Portal users must manually upload new photos through their profile page.  
  The default value is 1. |

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>server</code></td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><code>applicationVersion</code></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

---

**Table:**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>timeToLive</code></td>
<td>Time to live for profile objects (in minutes) in the profile cache. The default value is 60 minutes.</td>
</tr>
<tr>
<td><code>server</code></td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><code>applicationVersion</code></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.15.11.3 Example
The following example enables profile photo synchronization in WebCenter Portal:

```
wlsec/weblogic/serverConfig>setProfilePhotoSync(appName='webcenter',
enablePhotoSync=1)
```

10.15.12 printProfileConfig

Module: Oracle WebCenter Portal

Use with WLST: Online

10.15.12.1 Description
(Deprecated from 11.1.1.8.0. Use `setProfileConfig` instead.)

Prints profile cache configuration values.

10.15.12.2 Syntax

`printProfileConfig(appName, [server, applicationVersion])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td><code>server</code></td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><code>applicationVersion</code></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.15.12.3 Example
The following example displays the current profile cache configuration for WebCenter Portal:

```
wls/weblogic/serverConfig>printProfileConfig(appName='webcenter')
```

10.16 Personal Events

Use the commands listed in Table 10–25 to manage personal events server connections for a named application.
Configuration changes made using these WLST commands are only effective after your restart the Managed Server on which the application is deployed. For details, see *Oracle Fusion Middleware Administering Oracle WebCenter Portal*.

### Table 10–25  Personal Events WLST Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>createPersonalEventConnection</code></td>
<td>Create a personal events server connection for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td><code>setPersonalEventConnection</code></td>
<td>Edit an existing personal events server connection.</td>
<td>Online</td>
</tr>
<tr>
<td><code>listPersonalEventConnections</code></td>
<td>List all of the personal events server connections that are configured for a named application</td>
<td>Online</td>
</tr>
</tbody>
</table>

#### 10.16.1 `createPersonalEventConnection`

**Module:** Oracle WebCenter Portal

**Use with WLST:** Online

#### 10.16.1.1 Description

Creates a personal events server connection for a named application.


While you can register multiple personal events connections for WebCenter Portal and Portal Framework applications, only one connection is used for personal events services - the default (or active) connection.

#### 10.16.1.2 Syntax

`createPersonalEventConnection(appName, name, webServiceUrl, adapterName, appId, [default, server, applicationVersion])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td><code>name</code></td>
<td>Connection name. The name must be unique (across all connection types within the application.</td>
</tr>
<tr>
<td><code>webServiceUrl</code></td>
<td>URL of the web service exposing the event application. Use the format: protocol://host:port/appWebServiceInterface/WSName</td>
</tr>
<tr>
<td><code>adapterName</code></td>
<td>Specify the adapter that matches the personal events server. Valid values are MSEx2003 and MSEx2007. Choose MSEx2003 for Microsoft Exchange Server 2003 and MSEx2007 for Microsoft Exchange Server 2007. Each adapter has its own mechanism of authenticating and exchanging data</td>
</tr>
</tbody>
</table>
10.16.1.3 Example
The following example creates a connection named MyPEConnection for WebCenter Portal (webcenter). The connection points to a Microsoft Exchange Server 2007 and is designated as the default connection for personal event services:

```wls:/weblogic/serverConfig>createPersonalEventConnection(appName='webcenter',
name='MyPEConnection', webServiceUrl='http://myexchange.com/EWS/Services.wsdl',
adapterName='MSEx2007', appId='ExtPEApp', default=1)
```

The following example creates a connection named MyPEConnection for a WebCenter Portal. The connection points to a Microsoft Exchange Server 2003:

```wls:/weblogic/serverConfig>createPersonalEventConnection(appName='webcenter',
name='MyPEConnection', webServiceUrl='http://myexchange.com/ExchangeWS/PersonalEventsWebService.asmx',
adapterName='MSEx2003', appId='ExtPEApp')
```

10.16.2 setPersonalEventConnection
Module: Oracle WebCenter Portal
Use with WLST: Online

10.16.2.1 Description
Edits an existing personal events server connection configured for a named application.

10.16.2.2 Syntax
```
setPersonalEventConnection(appName, name, [webServiceUrl, adapterName, appId, 
default, server, applicationVersion])
```
### Argument Definitions

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing personal events server connection.</td>
</tr>
<tr>
<td>webServiceUrl</td>
<td>URL of the web service exposing the event application. Use the format: <code>protocol://host:port/appWebServiceInterface/WSName</code></td>
</tr>
<tr>
<td>adapterName</td>
<td>Optional. Specify the adapter that matches the personal events server. Valid values are <code>MSEx2003</code> and <code>MSEx2007</code>. Choose <code>MSEx2003</code> for Microsoft Exchange Server 2003 and <code>MSEx2007</code> for Microsoft Exchange Server 2007. Each adapter has its own mechanism of authenticating and exchanging data.</td>
</tr>
<tr>
<td>appId</td>
<td>Optional. External application associated with the Microsoft Exchange Server providing personal events services. If specified, external application credential information is used to authenticate users against the Microsoft Exchange Server.</td>
</tr>
<tr>
<td>default</td>
<td>Optional. Indicates whether this connection is the default connection for personal events. Valid values are 1 (true) and 0 (false). The default for this argument is 0. To specify that personal events uses this connection, set the value to 1. While you can register multiple connections for WebCenter Portal and Portal Framework applications, only one connection is used for personal event services—the default (or active) connection.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

#### 10.16.2.3 Example

The following example updates the web service URL for a connection named `MyPEConnection`.

```wls`<br>wslic:weblogic/serverConfig> setPersonalEventConnection(appName='webcenter', name='MyPEConnection', webServiceUrl='http://myexchange.com/EWS/Services.wsdl')<br>`
```

The following example makes a connection named `MyPEConnection` the default connection for personal events services in WebCenter Portal.

```wls`<br>wslic:weblogic/serverConfig> setPersonalEventConnection(appName='webcenter', name='MyPEConnection', default=1)<br>`
```

#### 10.16.3 listPersonalEventConnections

Module: Oracle WebCenter Portal  
Use with WLST: Online
10.16.3.1 Description
Lists all of the personal events server connections that are configured for a named application.

10.16.3.2 Syntax
listPersonalEventConnections(appName, [verbose, name, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Displays connection details for personal events in verbose mode.</td>
</tr>
<tr>
<td></td>
<td>Valid options are 1 (true) and 0 (false).</td>
</tr>
<tr>
<td></td>
<td>When set to 1, listPersonalEventConnections lists all of the personal</td>
</tr>
<tr>
<td></td>
<td>events server connections that are configured for a named application,</td>
</tr>
<tr>
<td></td>
<td>along with their details.</td>
</tr>
<tr>
<td></td>
<td>When set to 0, only connection names are listed. This argument defaults</td>
</tr>
<tr>
<td></td>
<td>to 0.</td>
</tr>
<tr>
<td></td>
<td>When set to 0, do not specify the name argument.</td>
</tr>
<tr>
<td>name</td>
<td>Optional. Name of an existing personal events connection. Use this</td>
</tr>
<tr>
<td></td>
<td>argument to view connection details for a specific personal events server.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different</td>
</tr>
<tr>
<td></td>
<td>servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if</td>
</tr>
<tr>
<td></td>
<td>more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.16.3.3 Example
The following example lists connection names and details for all of the personal events server connections currently configured for WebCenter Portal (webcenter):

wls:/weblogic/serverConfig>listPersonalEventConnections(appName='webcenter', verbose=1)

The following example displays connection details for a personal events server connection named MyPEConnection:

wls:/weblogic/serverConfig>listPersonalEventConnections(appName='webcenter', verbose=1, name='MyPEConnection')

10.17 Personalization
Use the commands listed in Table 10–26 to manage personalization connections for a named application.

Configuration changes made using these WLST commands are only effective after your restart the Managed Server on which personalization is deployed. For details, see Oracle Fusion Middleware Administering Oracle WebCenter Portal.
Table 10–26  Personalization WLST Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>createWCPSCMISConnection</td>
<td>Create a CMIS connection for the personalization.</td>
<td>Online</td>
</tr>
<tr>
<td>createWCPSActivityGraphConnection</td>
<td>Create an activity graph connection for the personalization.</td>
<td>Online</td>
</tr>
<tr>
<td>createWCPSPeopleConnection</td>
<td>Create a People connection for the personalization.</td>
<td>Online</td>
</tr>
<tr>
<td>createWCPSCustomConnection</td>
<td>Create a custom connection for the personalization.</td>
<td>Online</td>
</tr>
<tr>
<td>setWCPSConnectionProperty</td>
<td>Modify properties of an existing connection for the personalization.</td>
<td>Online</td>
</tr>
<tr>
<td>listWCPSCMISConnection</td>
<td>List CMIS connections configured for the personalization.</td>
<td>Online</td>
</tr>
<tr>
<td>listWCPSActivityGraphConnection</td>
<td>List activity graph connections configured for the personalization.</td>
<td>Online</td>
</tr>
<tr>
<td>listWCPSPeopleConnection</td>
<td>List People connections configured for the personalization.</td>
<td>Online</td>
</tr>
<tr>
<td>listWCPSCustomConnection</td>
<td>List custom connections configured for the personalization.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteWCPSCMISConnection</td>
<td>Create a CMIS connection for the personalization.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteWCPSActivityGraphConnection</td>
<td>Create an activity graph connection for the personalization.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteWCPSPeopleConnection</td>
<td>Create a People connection for the personalization.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteWCPSCustomConnection</td>
<td>Create a custom connection for the personalization.</td>
<td>Online</td>
</tr>
</tbody>
</table>

10.17.1 createWCPSCMISConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

10.17.1.1 Description

Creates a connection to a CMIS (Content Management Interoperability Service) provider for personalization services.

10.17.1.2 Syntax

createWCPSCMISConnection(name, repositoryId, host, port, [scheme, namespace, isDefault, path, pathPrepend, servletPathPart, rewriteUrls, pathTrim, timeoutInMilliSecs, propagateTimeoutExceptions, server])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Connection name. The name must be unique for this connection type within a namespace.</td>
</tr>
</tbody>
</table>
10.17.1.3 Example
The following example creates a connection to a CMIS data provider:

```wls:/weblogic/serverConfig>createWCPSCMISConnection(name='Repos1CMISConnection',
  repositoryId=CMIS repository ID.
  Typically, the name of the Oracle WebCenter Content repository connection.
  host=Hostname of the server that is hosting the CMIS REST service.
  Typically, the machine name of the WC_Spaces managed server.
  port=Port of the server that is hosting the CMIS REST service.
  Typically, the port number of the WC_Spaces managed server.
  scheme=Optional. HTTP scheme for accessing the CMIS REST service.
  Valid options are: http and https (for SSL)
  Defaults to http.
  namespace=Optional. Namespace for the connection. If not specified or set to
  none, the connection is available to all namespaces.
  isDefault=Optional. Indicates whether this connection is the default CMIS
  connection for personalization services.
  Valid values are 1 (true) or 0 (false).
  Defaults to 0.
  path=Optional. CMIS service URL path.
  Defaults to:/api/cmis/repository/<repositoryId>
  pathPrepend=Optional. Base CMIS service URL path to prepend to the
  servletPathPart and path.
  Defaults to /rest.
  servletPathPart=Optional. Servlet section of the CMIS service URL path.
  rewriteUrls=Optional. Specifies how to rewrite URLs returned from the CMIS
  REST service.
  Valid options are producer, consumer, and none. Defaults to none.
  For more details, see 'Managing Personalization' in the Oracle Fusion
  Middleware Administering Oracle WebCenter Portal.
  pathTrim=Optional. Specifies the path parts to trim from URLs returned from
  the CMIS REST service. Defaults to None.
  For more details, see 'Managing Personalization' in the Oracle Fusion
  Middleware Administering Oracle WebCenter Portal.
  timeoutInMillisecs=Optional. Timeout in milliseconds (as a string) to wait for CMIS calls
  to return, or None for no timeout.
  Defaults to None.
  propagateTimeoutExceptions=Optional. Valid values are 1 (true) and 0 (false).
  When set to 1, CMIS call timeouts raise an exception.
  When set to 0, exceptions are not raised.
  server=Optional. Name of the Managed Server hosting personalization.
  This parameter is only required in a nondefault deployment configuration. No value is required for a default deployment where
  the WCUtilities server hosts personalization.
repositoryId='ucm11g-server', host='myhost.com', port=8888, scheme='http', isDefault=1)

10.17.2 createWCPSActivityGraphConnection
Module: Oracle WebCenter Portal
Use with WLST: Online

10.17.2.1 Description
Creates a connection to an activity graph data provider for personalization services.

10.17.2.2 Syntax
createWCPSActivityGraphConnection(name, host, port, [scheme, namespace, isDefault, restResourceIndex, rewriteUrls, pathTrim, server)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Connection name. Must be unique for this connection type within a namespace.</td>
</tr>
<tr>
<td>host</td>
<td>Hostname of the server hosting activity graph REST services. Typically, the machine name of the WC_Spaces managed server.</td>
</tr>
<tr>
<td>port</td>
<td>Port of the server hosting activity graph services. Typically, the port number of the WC_Spaces managed server.</td>
</tr>
<tr>
<td>scheme</td>
<td>Optional. HTTP scheme for accessing activity graph. Valid options are http and https. Defaults to http.</td>
</tr>
<tr>
<td>namespace</td>
<td>Optional. Namespace for the connection. If not specified or set to none, the connection is available to all namespaces.</td>
</tr>
<tr>
<td>isDefault</td>
<td>Optional. Indicates whether this connection is the default activity graph connection. Valid values are 1 (true) or 0 (false). Defaults to 0.</td>
</tr>
<tr>
<td>restResourceIndex</td>
<td>Optional. URL path for the resourceIndex of the REST server. Defaults to /rest/api/resourceIndex.</td>
</tr>
<tr>
<td>rewriteUrls</td>
<td>Optional. Specifies how to rewrite URLs returned from activity graph REST services. Valid options are producer, consumer, and none. Defaults to none.</td>
</tr>
<tr>
<td>pathTrim</td>
<td>Optional. Specifies the path parts to trim from URLs returned from activity graph REST services. Defaults to None. For more details, see 'Managing Personalization' in Oracle Fusion Middleware Administering Oracle WebCenter Portal.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the Managed Server hosting personalization. This parameter is only required in a nondefault deployment configuration. No value is required for a default deployment where the WC_Utilities server hosts personalization services.</td>
</tr>
</tbody>
</table>

10.17.2.3 Example
The following example creates a connection to an activity graph data provider in a particular namespace:
10.17.3 createWCPSPeopleConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

10.17.3.1 Description

Creates a connection to a people connections provider for personalization services.

10.17.3.2 Syntax

createWCPSPeopleConnection(name, host, port, [scheme], [namespace], [isDefault], [restResourceIndex], [rewriteUrls], [pathTrim], [server])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Connection name. Must be unique for this connection type within a namespace.</td>
</tr>
<tr>
<td>host</td>
<td>Hostname of the server hosting the people connections REST service.</td>
</tr>
<tr>
<td>port</td>
<td>Port of the server hosting people connections.</td>
</tr>
<tr>
<td>scheme</td>
<td>Optional. HTTP scheme for accessing people connection services.</td>
</tr>
<tr>
<td>namespace</td>
<td>Optional. Namespace for the connection.</td>
</tr>
<tr>
<td>isDefault</td>
<td>Optional. Indicates whether this connection is the default for people</td>
</tr>
<tr>
<td></td>
<td>connections. Valid values are 1 (true) or 0 (false). Defaults to 0.</td>
</tr>
<tr>
<td>restResourceIndex</td>
<td>Optional. URL path for the resourceIndex of the REST server.</td>
</tr>
<tr>
<td>rewriteUrls</td>
<td>Optional. Specifies how to rewrite URLs returned from the people</td>
</tr>
<tr>
<td></td>
<td>connections REST service. Valid options are producer, consumer, and</td>
</tr>
<tr>
<td></td>
<td>none. Defaults to none. For more details, see 'Managing Personalization' in</td>
</tr>
<tr>
<td></td>
<td>Oracle Fusion Middleware Administering Oracle WebCenter Portal.</td>
</tr>
<tr>
<td>pathTrim</td>
<td>Optional. Specifies the path parts to trim from URLs returned from people</td>
</tr>
<tr>
<td></td>
<td>connections. Defaults to None. For more details, see 'Managing Personalization' in Oracle Fusion Middleware Administering Oracle WebCenter Portal.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the Managed Server hosting personalization services.</td>
</tr>
<tr>
<td></td>
<td>This parameter is only required in a nondefault deployment configuration.</td>
</tr>
<tr>
<td></td>
<td>No value is required for a default deployment where the WC_Utilities server hosts personalization services.</td>
</tr>
</tbody>
</table>
10.17.3.3 Example
The following example creates a connection to a people connections provider in the default namespace:

```
wlst:/weblogic/serverConfig> createWCPSPeopleConnection(name='PeopleConnection', host='myhost.com', port=8888)
```

10.17.4 createWCPSCustomConnection
Use with WLST: Online

10.17.4.1 Description
Creates a connection to a custom data provider for personalization services.
Custom connection types are used with custom data providers and property locators.

10.17.4.2 Syntax
```
createWCPSCustomConnection(name, type, [namespace, properties, [server]])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Connection name. Must be unique for this connection type within a namespace.</td>
</tr>
<tr>
<td>type</td>
<td>Custom connection type specific to the custom data provider or property locator implementation.</td>
</tr>
<tr>
<td>namespace</td>
<td>Optional. Namespace for the connection. If not specified or set to none, the connection is available to all namespaces.</td>
</tr>
<tr>
<td>properties</td>
<td>Optional. Dictionary of connection properties and values. The set of properties is specific to the connection type. All values in the dictionary must be strings.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the Managed Server hosting personalization. This parameter is only required in a nondefault deployment configuration. No value is required for a default deployment where the WC_Utilities server hosts personalization services.</td>
</tr>
</tbody>
</table>

10.17.4.3 Example
The following example creates a connection to a custom data provider with several properties:

```
wls:/weblogic/serverConfig> createWCPSCustomConnection(name='CustomConnection', type='my.connection.type', properties={'prop1': 'value1', 'prop2': 'value2'})
```

10.17.5 listWCPSCMISConnection
Module: Oracle WebCenter Portal
Use with WLST: Online

10.17.5.1 Description
Lists all CMIS (Content Management Interoperability Service) connections configured for personalization or lists a single connection.

10.17.5.2 Syntax
```
listWCPSCMISConnections([server], [verbose], [name], [namespace])
```
### 10.17.5.3 Example

The following example lists the names of all the CMIS connections:

```bash
wls:/weblogic/serverConfig> listWCPSCMISConnections(verbos=0)

Repos1CMISConnection
Repos2CMISConnection
```

The following example lists the details of one CMIS connection:

```bash
wls:/weblogic/serverConfig> listWCPSCMISConnections(name='Repos1CMISConnection')
```

```
----------------------
Repos1CMISConnection (type=cmis.provider.connection, namespace=*)
----------------------
host: myhost.com
isDefault: false
path: /api/cmis/repository/repo1
pathPrepend: /rest
port: 8888
repositoryId: ucm11g-server
rewriteUrls: none
scheme: http
```

### 10.17.6 listWCPSActivityGraphConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.17.6.1 Description

Lists all activity graph connections configured for the personalization or lists a single connection.

#### 10.17.6.2 Syntax

```bash
listWCPSActivityGraphConnections([server], [verbose], [name], [namespace])
```
### 10.17.6.3 Example
The following example lists the names of all activity graph data provider connections:

```
    wls:/weblogic/serverConfig> listWCPSActivityGraphConnections(verbose=0)

    AG1Connection
    AG2Connection
```

The following example lists the details of one activity graph data provider connection:

```
    wls:/weblogic/serverConfig> listWCPSActivityGraphConnections(name='AG1Connection')

    ----------------------
    AG1Connection (type=activity.provider.connection, namespace=*)
    ----------------------
    host: myhost.com
    isDefault: false
    port: 8888
    restResourceIndex: /rest/api/resourceIndex
    rewriteUrls: producer
    scheme: http
```

### 10.17.7 `listWCPSPeopleConnection`
Module: Oracle WebCenter Portal
Use with WLST: Online

#### 10.17.7.1 Description
Lists all people connections configured for personalization or lists a single connection.

#### 10.17.7.2 Syntax
```
    listWCPSPeopleConnections([server], [verbose], [name], [namespace])
```
The following example lists the names of all people connections for personalization:

```
  wls:/weblogic/serverConfig> listWCPSPeopleConnections(verbos=0)
  People1Connection
  People2Connection
```

The following example lists the details of a single connection:

```
  wls:/weblogic/serverConfig> listWCPSPeopleConnections(name='PeopleConnection')
```

```
----------------------
PeopleConnection (type=people.service.connection, namespace=*)
----------------------
  host: myhost.com
  isDefault: false
  port: 8888
  restResourceIndex: /rest/api/resourceIndex
  rewriteUrls: producer
  scheme: http
```

### 10.17.8 listWCPSCustomConnection

**Module:** Oracle WebCenter Portal  
**Use with WLST:** Online

#### 10.17.8.1 Description

Lists all connections of a particular type configured for personalization or lists a single connection.

Custom connection types are used with custom data providers and property locators.

#### 10.17.8.2 Syntax

```
listWCPSCustomConnections(type, [server], [verbose], [name], [namespace])
```
WebCenter Portal Custom WLST Commands

10.17.8.3 Example
The following example lists the names of all connections with the type my.connection.type:

```plaintext
wls:/weblogic/serverConfig> listWCPSCustomConnections(type='my.connection.type',
verbose=0)

Custom1Connection
Custom2Connection
```

The following example lists the details of one custom connection for personalization:

```plaintext
wls:/weblogic/serverConfig> listWCPSPeopleConnections(type='my.connection.type',
name='CustomConnection')

----------------------
CustomConnection (type=my.connection.type, namespace=*),
----------------------
host: myhost.com
isDefault: false
port: 8888
customConnectionProperty: someValue
scheme: http
```

10.17.9 deleteWCPSCMISConnection
Module: Oracle WebCenter Portal
Use with WLST: Online

10.17.9.1 Description
Deletes a CMIS (Content Management Interoperability Service) connection configured for personalization.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Custom connection type specific to the custom data provider or property locator implementation.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the Managed Server hosting personalization. This parameter is only required in a nondefault deployment configuration. No value is required for a default deployment where the WC_Utilities server hosts personalization services.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Controls verbose or concise output. Valid options are 1 and 0. When set to 1, this command lists the connections and their properties. When set to 0, this command lists connection names only. Defaults to 1.</td>
</tr>
<tr>
<td>name</td>
<td>Optional. Name of an existing connection. If not specified or set to None, then all connections are listed.</td>
</tr>
<tr>
<td>namespace</td>
<td>Optional. Namespace for the connection you want to list. If not specified or set to none, this command lists connections configured to be available in all namespaces.</td>
</tr>
</tbody>
</table>
10.17.9.2 Syntax
deleteWCPSCMISConnection(name, [namespace, server])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Connection name.</td>
</tr>
<tr>
<td>namespace</td>
<td>Optional. Namespace for the connection you want to delete. If not specified or set to none, this command deletes connections configured to be available in all namespaces.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the Managed Server hosting personalization. This parameter is only required in a nondefault deployment configuration. No value is required for a default deployment where the WC_Utilities server hosts personalization services.</td>
</tr>
</tbody>
</table>

10.17.9.3 Example
The following example deletes a CMIS connection configured for personalization:

```
wls:/weblogic/serverConfig>deleteWCPSCMISConnection(name='ReposCMISConnection')
```

10.17.10 deleteWCPSActivityGraphConnection
Module: Oracle WebCenter Portal

Use with WLST: Online

10.17.10.1 Description
Deletes an activity graph connection configured for personalization.

10.17.10.2 Syntax
deleteWCPSActivityGraphConnection(name, [namespace, server])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Connection name.</td>
</tr>
<tr>
<td>namespace</td>
<td>Optional. Namespace for the connection you want to delete. If not specified or set to none, this command deletes connections configured to be available in all namespaces.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the Managed Server hosting personalization. This parameter is only required in a nondefault deployment configuration. No value is required for a default deployment where the WC_Utilities server hosts personalization services.</td>
</tr>
</tbody>
</table>

10.17.10.3 Example
The following example deletes an activity graph connection configured for personalization:

```
wls:/weblogic/serverConfig>deleteWCPSActivityGraphConnection(name='AGConnection')
```

10.17.11 deleteWCPSPeopleConnection
Module: Oracle WebCenter Portal

Use with WLST: Online
10.17.11 Description
Deletes a people connection configured for personalization.

10.17.11.2 Syntax
```
deleteWCPSPeopleConnection(name, [namespace, server])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Connection name.</td>
</tr>
<tr>
<td>namespace</td>
<td>Optional. Namespace for the connection you want to delete. If not specified or set to none, this command deletes connections configured to be available in all namespaces.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the Managed Server hosting personalization. This parameter is only required in a nondefault deployment configuration. No value is required for a default deployment where the WC_Utilities server hosts personalization services.</td>
</tr>
</tbody>
</table>

10.17.11.3 Example
The following example deletes a people connection configured for personalization:
```
wls:/weblogic/serverConfig> deleteWCPSPeopleConnection(name='PeopleConnection')
```

10.17.12 deleteWCPSCustomConnection
Module: Oracle WebCenter Portal
Use with WLST: Online

10.17.12.1 Description
Deletes a custom data provider connection configured for personalization.

10.17.12.2 Syntax
```
deleteWCPSCustomConnection(name, type, [namespace, server])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Connection name.</td>
</tr>
<tr>
<td>type</td>
<td>Custom connection type.</td>
</tr>
<tr>
<td>namespace</td>
<td>Optional. Namespace for the connection you want to delete. If not specified or set to none, this command deletes connections configured to be available in all namespaces.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the Managed Server hosting personalization. This parameter is only required in a nondefault deployment configuration. No value is required for a default deployment where the WC_Utilities server hosts personalization services.</td>
</tr>
</tbody>
</table>

10.17.12.3 Example
The following example deletes a custom data provider connection configured for personalization:
```
wls:/weblogic/serverConfig> deleteWCPSCustomConnection(name='CustomConnection', type='my.connection.type')
```
10.17.13 setWCPSConnectionProperty
Module: Oracle WebCenter Portal
Use with WLST: Online

10.17.13.1 Description
Add, modify, or delete properties of an existing connection for personalization. The properties supported by a connection are specific to the connection type:

- CMIS connections support the following properties: repositoryId, host, port, scheme, path, pathPrepend, servletPathPart, rewriteUrls, pathTrim, isDefault, timeoutInMillisecs, propagateTimeoutException

See also, createWCPSCMISConnection.

- Activity graph and people connections support the following properties: host, port, scheme, restResourceIndex, rewriteUrls, pathTrim, isDefault

See also, createWCPSActivityGraphConnection and createWCPSPeopleConnection.

10.17.13.2 Syntax
setWCPSConnectionProperty{connectionName, connectionType, propertyName, propertyValue, [namespace], [server]}

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionName</td>
<td>Connection name.</td>
</tr>
<tr>
<td>connectionType</td>
<td>Connection type. Valid values are WCPS_CMIS_CONNECTION_TYPE, WCPS_AG_CONNECTION_TYPE, and WCPS_PC_CONNECTION_TYPE for CMIS, activity graph, and people connections, respectively. Alternatively, any valid, custom connection type can be specified</td>
</tr>
<tr>
<td>propertyName</td>
<td>Property name.</td>
</tr>
<tr>
<td>propertyValue</td>
<td>Property value as a string. Use None to remove a property value from the connection.</td>
</tr>
<tr>
<td>namespace</td>
<td>Optional. Namespace for the connection you want to change. If not specified or set to None, this command modifies properties of connections configured to be available in all namespaces.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the Managed Server hosting personalization. This parameter is only required in a nondefault deployment configuration. No value is required for a default deployment where the WC_Utility server hosts personalization services.</td>
</tr>
</tbody>
</table>

10.17.13.3 Example
The following example changes or adds a property to a CMIS connection:

```
wlst:/weblogic/serverConfig>
setWCPSConnectionProperty{connectionName='ReposCMISConnection',
                    connectionType='WCPS_CMIS_CONNECTION_TYPE',
                    propertyName='propagateTimeoutExceptions', propertyValue='false')
```

The following example removes a property from a custom data provider connection:

```
wlst:/weblogic/serverConfig>
setWCPSConnectionProperty{connectionName='CustomConnection',
                    connectionType='my.connection.type', propertyName='prop2', propertyValue=None)
```
10.18 Portlet Producers

Use the commands listed in Table 10–27 to manage portlet producers used in WebCenter Portal and Portal Framework applications.

All configuration changes made using these WLST commands are immediately available in the application.

**Table 10–27  Portlet Producer WLST Commands**

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>registerWSRPProducer</td>
<td>Create and register a WSRP producer.</td>
<td>Online</td>
</tr>
<tr>
<td>setWSRPProducer</td>
<td>Edit WSRP producer registration details.</td>
<td>Online</td>
</tr>
<tr>
<td>listWSRPProducers</td>
<td>List WSRP producer registration details.</td>
<td>Online</td>
</tr>
<tr>
<td>deregisterWSRPProducer</td>
<td>Deregister a WSRP producer, and delete the associated WSRP and web service connections.</td>
<td>Online</td>
</tr>
<tr>
<td>listWSRPProducerRegistrationProperties</td>
<td>List registration properties supported by a WSRP producer.</td>
<td>Online</td>
</tr>
<tr>
<td>listWSRPProducerUserCategories</td>
<td>List any user categories that the WSRP producer might support.</td>
<td>Online</td>
</tr>
<tr>
<td>mapWSRPProducerUserCategory</td>
<td>Map a role that is defined in the specified application to a user category supported by a WSRP producer.</td>
<td>Online</td>
</tr>
<tr>
<td>registerPDKJavaProducer</td>
<td>Create and register an Oracle PDK-Java producer.</td>
<td>Online</td>
</tr>
<tr>
<td>setPDKJavaProducer</td>
<td>Edit PDK-Java producer registration details.</td>
<td>Online</td>
</tr>
<tr>
<td>listPDKJavaProducers</td>
<td>List registered Oracle PDK-Java producers.</td>
<td>Online</td>
</tr>
<tr>
<td>deregisterPDKJavaProducer</td>
<td>Deregister an Oracle PDK-Java producer, deleting the associated connection.</td>
<td>Online</td>
</tr>
<tr>
<td>registerPageletProducer</td>
<td>Create and register a pagelet producer.</td>
<td>Online</td>
</tr>
<tr>
<td>setPageletProducer</td>
<td>Edit pagelet producer registration details.</td>
<td>Online</td>
</tr>
<tr>
<td>listPageletProducers</td>
<td>List pagelet producer registration details.</td>
<td>Online</td>
</tr>
<tr>
<td>deregisterPageletProducer</td>
<td>Deregister a pagelet producer, deleting the associated connection.</td>
<td>Online</td>
</tr>
<tr>
<td>refreshProducer</td>
<td>Refresh the metadata stored for the named producer to reflect the portlets currently offered by that producer.</td>
<td>Online</td>
</tr>
<tr>
<td>listPortletClientConfig</td>
<td>List portlet client configuration for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>setPortletClientConfig</td>
<td>Edit portlet client configuration settings.</td>
<td>Online</td>
</tr>
<tr>
<td>getPortletClientConfig</td>
<td>Return portlet client configuration settings.</td>
<td>Online</td>
</tr>
<tr>
<td>registerOOTBProducers</td>
<td>Register out-of-the-box producers provided with Oracle WebCenter Portal.</td>
<td>Online</td>
</tr>
<tr>
<td>deregisterOOTBProducers</td>
<td>Deregister out-of-the-box producers provided with Oracle WebCenter Portal.</td>
<td>Online</td>
</tr>
<tr>
<td>registerSampleProducers</td>
<td>Register the sample producers provided with Oracle WebCenter Portal.</td>
<td>Online</td>
</tr>
<tr>
<td>deregisterSampleProducers</td>
<td>Deregister sample producers.</td>
<td>Online</td>
</tr>
</tbody>
</table>
10.18.1 registerWSRPProducer

Module: Oracle WebCenter Portal

Use with WLST: Online

10.18.1.1 Description

Creates a connection to a WSRP portlet producer and registers the WSRP producer with a named application. When you create a WSRP producer connection, a web service connection is also created named <name>-wsconn where <name> is the value specified for the name argument.

10.18.1.2 Syntax

```java
registerWSRPProducer(appName, name, url, [proxyHost], [proxyPort], [timeout], [externalApp], [registrationProperties], [tokenType], [issuer], [defUser], [keyStorePath], [keyStorePswd], [sigKeyAlias], [sigKeyPswd], [encKeyAlias], [encKeyPswd], [recptAlias], [enforcePolicyURI], [server], [applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Connection name. The name must be unique (across all connection types) within the application. The name you specify here will appear in Composer (under the Portlets folder).</td>
</tr>
</tbody>
</table>
| url            | Producer WSDL URL. The syntax will vary according to your WSRP implementation, for example:
- http://host_name:port_number/context_root/portlets/wsrp2?WSDL
- http://host_name:port_number/context_root/portlets/wsrp1?WSDL (WSRP 1.0 for backward compatibility)
  Where:
  - host_name is the server where your producer is deployed
  - port_number is the HTTP listener port number
  - context_root is the web application's context root
  - portlets/[wsrp{1|2}]?WSDL is static text. The text entered here depends on how the producer is deployed.
  For example:
<p>| proxyHost      | Optional. Host name or IP address of the proxy server. A proxy is required when the application and the remote portlet producer are separated by a firewall and an HTTP proxy is needed to communicate with the producer. |
| proxyPort      | Optional. Port number on which the proxy server listens. |</p>
<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>timeout</strong></td>
<td>Optional. Timeout setting for communications with the producer, in seconds. For example, the maximum time the producer may take to register, deregister, or display portlets on portal pages. This argument defaults to 30. Individual portlets may define their own timeout period, which takes precedence over the value expressed here.</td>
</tr>
<tr>
<td><strong>registrationProperties</strong></td>
<td>Optional. A list of registration properties and their values. The format of this argument must be a comma-separated list of valid registration properties, each followed by an equals symbol and the value. For example: name=Producer, key=123. The registration properties for a producer can be found using listWSRPProducerRegistrationProperties. See Section 10.18.5, “listWSRPProducerRegistrationProperties”.</td>
</tr>
</tbody>
</table>
Argument| Definition
---|---
tokenType| Optional. Type of token profile to use for authentication with this WSRP producer.

When the argument `enforcePolicyURI=1`, valid values are:

- **USERNAME_WITHOUT_PASSWORD**
  (oracle/wss10_username_id_propagation_with_msg_protection_client_policy)—This policy provides message protection (integrity and confidentiality) and identity propagation for outbound SOAP requests in accordance with the WS-Security 1.0 standard. Credentials (user name only) are included in outbound SOAP request messages through a WS-Security UsernameToken header. No password is included.

  Message protection is provided using WS-Security 1.0's Basic128 suite of asymmetric key technologies. Specifically, RSA key mechanisms for message confidentiality, SHA-1 hashing algorithm for message integrity, and AES-128 bit encryption.

- **USERNAME_WITH_PASSWORD**
  (oracle/wss10_username_token_with_message_protection_client_policy)—This policy provides message protection (integrity and confidentiality) and authentication for outbound SOAP requests in accordance with the WS-Security v1.0 standard. Both plain text and digest mechanisms are supported.

  This policy uses WS-Security's Basic 128 suite of asymmetric key technologies. Specifically, RSA key mechanism for message confidentiality, SHA-1 hashing algorithm for message integrity, and AES-128 bit encryption.

  Use this token profile if the WSRP producer has a different identity store. You will need to define an external application pertaining to the producer and associate the external application with this producer.

- **SAML_TOKEN_WITH_MSG_INTEGRITY**
  (wss10_saml_token_with_message_integrity_client_policy)—This policy provides message-level integrity protection and SAML-based authentication for outbound SOAP requests in accordance with the WS-Security 1.0 standard. A SAML token, included in the SOAP message, is used in SAML-based authentication with sender vouches confirmation.

  This policy uses WS-Security's Basic 128 suite of asymmetric key technologies and SHA-1 hashing algorithm for message integrity.

  When this policy is selected, the recipient key alias (recptAlias) must be disabled.

- **SAML_TOKEN_WITH_MSG_PROTECTION**
  (oracle/wss10_saml_token_with_message_protection_client_policy)—This policy provides message-level protection (integrity and confidentiality) and SAML-based authentication for outbound SOAP requests in accordance with the WS-Security 1.0 standard. The web service consumer includes a SAML token in the SOAP header and the confirmation type is sender-vouches.

  This policy uses WS-Security's Basic 128 suite of asymmetric key technologies. Specifically, RSA key mechanisms for message confidentiality, SHA-1 hashing algorithm for message integrity, and AES-128 bit encryption. and SHA-1 hashing algorithm for message integrity.
<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
</table>
| tokenType           | **WSS11_SAML_TOKEN_WITH_MSG_PROTECTION** (oracle/wss11_saml_token_with_message_protection_client_policy)—This policy provides message-level protection (integrity and confidentiality) and SAML token population for outbound SOAP requests in accordance with the WS-Security 1.1 standard. A SAML token, included in the SOAP message, is used in SAML-based authentication with sender vouches confirmation. This policy uses the symmetric key technology for signing and encryption, and WS-Security’s Basic 128 suite of asymmetric key technologies for endorsing signatures.  
**WSS10_SAML_TOKEN_ONLY** (oracle/wss10_saml_token_client_policy)—This policy provides SAML-based authentication for outbound SOAP request messages in accordance with the WS-Security 1.0 standard. The policy propagates user identity and is typically used in intra departmental deployments where message protection and integrity checks are not required. This policy does not require any keystore configuration.  
If the argument `enforcePolicyURI=0`, you can specify any valid Oracle Web Services Manager (OWSM) policy URI for the `tokenType` argument. |
| issuer              | Optional. Name of the issuer of the token. The issuer name is the entity that vouches for the verification of the subject. For example: www.oracle.com.  
This argument only applies when the `tokenType` is: SAML_TOKEN_WITH_MSG_PROTECTION, SAML_TOKEN_WITH_MSG_INTEGRITY, WSS10_SAML_TOKEN_ONLY, WSS11_SAML_TOKEN_WITH_MSG_PROTECTION. |
| defUser             | Optional. User name to assert to the remote producer when the user is not authenticated with the portal application.  
When unauthenticated, the identity `anonymous` is associated with the application user. The value `anonymous` may be inappropriate for the remote producer, so you may need to specify an alternative identity here. Keep in mind though, that in this case, the application has not authenticated the user so the default user you specify should be a low privileged user in the remote producer. If the user has authenticated to the application, the user’s identity is asserted rather than the default user.  
This argument only applies when the `tokenType` is: USERNAME_WITHOUT_PASSWORD, SAML_TOKEN_WITH_MSG_PROTECTION, SAML_TOKEN_WITH_MSG_INTEGRITY, WSS10_SAML_TOKEN_ONLY, WSS11_SAML_TOKEN_WITH_MSG_PROTECTION. |
| extApp              | Optional. This argument applies when the `tokenType` is `USERNAME_WITHOUT_PASSWORD`. If this producer uses an external application to store and supply user credentials for authentication, use this argument to name the associated external application. |
| keyStorePath        | Optional. Full path to the key store that contains the certificate and the private key that is used for signing some parts of the SOAP message, such as the security token and SOAP message body. The selected file should be a key store created with the Java keytool. |
| keyStorePswd        | Optional. Password to the key store that was set when the key store was created. |
| sigKeyAlias         | Optional. Identifier for the certificate associated with the private key that is used for signing. |
| sigKeyPswd          | Optional. Password for accessing the key identified by the alias that is specified using the `sigKeyAlias` argument. |
Examples

The following example registers a WSRP producer named WSRPSamples with WebCenter Portal (webcenter):

```
wls:/weblogic/serverConfig> registerWSRPProducer(appName='webcenter',
name='WSRPSamples', url='http://myhost.com:9999/
portletapp/portlets/wsrp2?WSDL')
```

The following example registers a secure WSRP producer:

```
wls:/weblogic/serverConfig> registerWSRPProducer(appName='webcenter',
tokenType='WSS11_SAML_TOKEN_WITH_MSG_PROTECTION', issuer='www.oracle.com',
defUser='anonymous', keyStorePath='/keys/mykeystore.jks', keyStorePswd='Test1',
sigKeyAlias='mysigalias', sigKeyPswd='mysigpswd', encKeyAlias='myencalias',
enckKeyPswd='myencpswd', recptAlias='myrcptalias')
```

10.18.2 setWSRPProducer

Module: Oracle WebCenter Portal

Use with WLST: Online

10.18.2.1 Description

Edits registration details for an existing WSRP producer.

10.18.2.2 Syntax

```
setWSRPProducer(appName, name, [url], [proxyHost], [proxyPort], [timeout],
```
Portlet Producers

[externalApp], [tokenType], [issuer], [defUser], [keyStorePath], [keyStorePswd]
[sigKeyAlias], [sigKeyPswd], [encKeyAlias], [encKeyPswd], [recptAlias],
[enforcePolicyURI], [server], [applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing WSRP producer.</td>
</tr>
</tbody>
</table>
| url      | Optional. WSRP producer URL. The syntax will vary according to your WSRP implementation, for example: http://host_name:port_number/context_root/portlets/wsrp2?WSDL http://host_name:port_number/context_root/portlets/wsrp1?WSDL http://host_name:port_number/context_root/portlets/?WSDL (WSRP 1.0 for backward compatibility) Where:  
  - host_name is the server where your producer is deployed  
  - port_number is the HTTP listener port number  
  - context_root is the Web application’s context root  
  - portlets[/wsrp(1|2)]?WSDL is static text. The text entered here depends on how the producer is deployed. For example: http://myhost:7778/MyPortletApp/portlets/wsrp2?WSDL |
<p>| proxyHost| Optional. Host name or IP address of the proxy server. A proxy is required when the application and the remote portlet producer are separated by a firewall and an HTTP proxy is needed to communicate with the producer. |
| proxyPort| Optional. Port number on which the proxy server listens. |
| timeout  | Optional. Timeout setting for communications with the producer, in seconds. For example, the maximum time the producer may take to register, deregister, or display portlets on portal pages. This argument defaults to 30. Individual portlets may define their own timeout period, which takes precedence over the value expressed here. |
| extApp   | Optional. This argument applies when the tokenType is USERNAME_ WITH_PASSWORD. If this producer uses an external application to store and supply user credentials for authentication, use this argument to name the associated external application. |</p>
<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>tokenType</code></td>
<td>Optional. Type of token profile to use for authentication with this WSRP producer. When the argument <code>enforcePolicyURI=1</code>, valid values are:</td>
</tr>
<tr>
<td></td>
<td>- <strong>USERNAME_WITHOUT_PASSWORD</strong> (oracle/wss10_username_id_propagation_with_msg_protection_client_policy)—This policy provides message protection (integrity and confidentiality) and identity propagation for outbound SOAP requests in accordance with the WS-Security 1.0 standard. Credentials (user name only) are included in outbound SOAP request messages through a WS-Security UsernameToken header. No password is included. Message protection is provided using WS-Security 1.0’s Basic 128 suite of asymmetric key technologies. Specifically, RSA key mechanisms for message confidentiality, SHA-1 hashing algorithm for message integrity, and AES-128 bit encryption.</td>
</tr>
<tr>
<td></td>
<td>- <strong>USERNAME_WITH_PASSWORD</strong> (oracle/wss10_username_token_with_message_protection_client_policy)—This policy provides message protection (integrity and confidentiality) and authentication for outbound SOAP requests in accordance with the WS-Security v1.0 standard. Both plain text and digest mechanisms are supported. This policy uses WS-Security’s Basic 128 suite of asymmetric key technologies. Specifically, RSA key mechanism for message confidentiality, SHA-1 hashing algorithm for message integrity, and AES-128 bit encryption. Use this token profile if the WSRP producer has a different identity store. You will need to define an external application pertaining to the producer and associate the external application with this producer.</td>
</tr>
<tr>
<td></td>
<td>- <strong>SAML_TOKEN_WITH_MSG_INTEGRITY</strong> (wss10_saml_token_with_message_integrity_client_policy)—This policy provides message-level integrity and SAML-based authentication for outbound SOAP requests in accordance with the WS-Security 1.0 standard. A SAML token, included in the SOAP message, is used in SAML-based authentication with sender vouches confirmation. This policy uses WS-Security’s Basic 128 suite of asymmetric key technologies and SHA-1 hashing algorithm for message integrity. When this policy is selected, the recipient key alias (<code>recptAlias</code>) must be disabled.</td>
</tr>
<tr>
<td></td>
<td>- <strong>SAML_TOKEN_WITH_MSG_PROTECTION</strong> (oracle/wss10_saml_token_with_message_protection_client_policy)—This policy provides message-level protection (integrity and confidentiality) and SAML-based authentication for outbound SOAP requests in accordance with the WS-Security 1.0 standard. The Web service consumer includes a SAML token in the SOAP header and the confirmation type is sender-vouches. This policy uses WS-Security’s Basic 128 suite of asymmetric key technologies. Specifically, RSA key mechanisms for message confidentiality, SHA-1 hashing algorithm for message integrity, and AES-128 bit encryption. and SHA-1 hashing algorithm for message integrity.</td>
</tr>
<tr>
<td>Argument</td>
<td>Definition</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>tokenType</td>
<td>continued...</td>
</tr>
</tbody>
</table>
|                    | ■ **WSS11_SAML_TOKEN_WITH_MSG_PROTECTION**  
(oracle/wss11_saml_token_with_message_protection_client_policy)—This policy enables message-level protection (integrity and confidentiality) and SAML token population for outbound SOAP requests in accordance with the WS-Security 1.1 standard. A SAML token, included in the SOAP message, is used in SAML-based authentication with sender vouches confirmation. This policy uses the symmetric key technology for signing and encryption, and WS-Security's Basic 128 suite of asymmetric key technologies for endorsing signatures. | |
|                    | ■ **WSS10_SAML_TOKEN_ONLY**  
(oracle/wss10_saml_token_client_policy)—This policy includes SAML-tokens in outbound SOAP request messages in accordance with the WS-Security 1.0 standard. The policy propagates user identity and is typically used in intra-departmental deployments where message protection and integrity checks are not required.  
This policy does not require any keystore configuration. If the argument enforcePolicyURI=0, you can specify any valid Oracle Web Services Manager (OWSM) policy URI for the tokenType argument. | |
| issuer             | Optional. Name of the issuer of the token. The issuer name is the entity that vouches for the verification of the subject. For example: www.oracle.com.  
This argument only applies when the tokenType is: SAML_TOKEN_WITH_MSG_PROTECTION, SAML_TOKEN_WITH_MSG_INTEGRITY, WSS10_SAML_TOKEN_ONLY, WSS11_SAML_TOKEN_WITH_MSG_PROTECTION. | |
| defUser            | Optional. User name to assert to the remote producer when the user is not authenticated with the application.  
When unauthenticated, the identity anonymous is associated with the application user. The value anonymous may be inappropriate for the remote producer, so you may need to specify an alternative identity here. Keep in mind though, that in this case, the application has not authenticated the user so the default user you specify should be a low privileged user in the remote producer. If the user has authenticated to the application, the user's identity is asserted rather than the default user.  
This argument only applies when the tokenType is: USERNAME_WITHOUT_PASSWORD, SAML_TOKEN_WITH_MSG_PROTECTION, SAML_TOKEN_WITH_MSG_INTEGRITY, WSS10_SAML_TOKEN_ONLY, WSS11_SAML_TOKEN_WITH_MSG_PROTECTION. | |
| keyStorePath       | Optional. Full path to the key store that contains the certificate and the private key that is used for signing some parts of the SOAP message, such as the security token and SOAP message body. The selected file should be a key store created with the Java keytool. | |
| keyStorePswd       | Optional. Password to the key store that was set when the key store was created. | |
| sigKeyAlias        | Optional. Identifier for the certificate associated with the private key that is used for signing. | |
| sigKeyPswd         | Optional. Password for accessing the key identified by the alias that is specified using the sigKeyAlias argument. | |
| encKeyAlias        | Optional. Key alias used by the producer to encrypt the return message. A valid value is one of the key aliases that is located in the specified key store. If not specified, the producer uses the signing key for encrypting the return message. |
**Portlet Producers**

**10.18.2.3 Example**

This example increases the timeout, for the WSRPSamples producer, to 60 seconds:

```
wlst:/weblogic/serverConfig> setWSRPProducer(appName='webcenter',
name='WSRPSamples', timeout=60)
```

This example updates security properties on a secure WSRP producer:

```
wlst:/weblogic/serverConfig> setWSRPProducer(appName='webcenter',
name='WSRPSamples2', tokenType='WSS11_SAML_TOKEN_WITH_MSG_PROTECTION',
issuer='www.oracle.com', defUser='anonymous',
keyStorePath='/keys/mykeystore.jks', keyStorePswd='Test1',
sigKeyAlias='mysigalias', sigKeyPswd='mysigpswd', encKeyAlias='myencalias',
encKeyPswd='myencpswd', recptAlias='myrcptalias')
```

This example removes all the security properties set on a secure WSRP producer:

```
wlst:/weblogic/serverConfig> setWSRPProducer(appName='webcenter',
name='WSRPSamples2', tokenType='')
```

**10.18.3 listWSRPProducers**

Module: Oracle WebCenter Portal

Use with WLST: Online

**10.18.3.1 Description**

Lists WSRP producer registration details for a named application.

**10.18.3.2 Syntax**

```
listWSRPProducers(appName, [name, verbose, server, [applicationVersion]])
```
### Portlet Producers

#### WebCenter Portal Custom WLST Commands

**10.18.3.3 Example**

The following example lists all the WSRP producers registered with a Portal Framework application named `myApp`:

```
wlst:/weblogic/serverConfig> listWSRPProducers(appName='myApp', verbose=0)
```

```
----------------------
WSRPSamples-connection
----------------------
```

The following example lists detailed connection information for a WSRP producer registered as `WSRPSamples-connection` with a Portal Framework application named `myApp`:

```
wlst:/weblogic/serverConfig> listWSRPProducers(appName='myApp', name='WSRPSamples-connection', verbose=1)
```

```
----------------------
WSRPSamples-connection
----------------------
Connection Name: WSRPSamples-connection
Web Service Connection Name: WSRPSamples-connection-wsconn
Proxy Host: None
Proxy Port: None
Timeout: 0
WSDL URL: http://example.com:7777/portletapp/portlets/wsrp2?WSDL
```

### 10.18.4 deregisterWSRPProducer

Module: Oracle WebCenter Portal
Use with WLST: Online

10.18.4.1 Description
Deregisters a WSRP producer, and deletes the associated WSRP and web service connections.

10.18.4.2 Syntax

deregisterWSRPProducer(appName, name, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing WSRP producer.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces. Required when applications with the same name are</td>
</tr>
<tr>
<td></td>
<td>deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than</td>
</tr>
<tr>
<td></td>
<td>one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.18.4.3 Example
The following example deregisters the WSRPSamples producer in WebCenter Portal (webcenter):

```
ws:/weblogic/serverConfig> deregisterWSRPProducer(appName='webcenter',
name='WSRPSamples')
```

10.18.5 listWSRPProducerRegistrationProperties
Module: Oracle WebCenter Portal
Use with WLST: Online

10.18.5.1 Description
Lists registration properties supported by a WSRP portlet producer.

10.18.5.2 Syntax

listWSRPProducerRegistrationProperties(appName, url, [proxyHost, proxyPort, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
</tbody>
</table>
10.18.5.3 Example
The following example lists valid registration properties for the WSRP producer with
the WSDL URL provided:

```
wlst:/weblogic/serverConfig> listWSRPProducerRegistrationProperties
(appName='webcenter', url='http://myhost:9999/portletapp/portlets/wsrp2?WSDL')
```

Registration Property hint : hint text
Registration Property label : label text
Registration Property language : en
Registration Property name : (urn:xyz:wlp:prop:reg:registration)consumerRole
Registration Property value : None

10.18.6 listWSRPProducerUserCategories
Module: Oracle WebCenter Portal
Use with WLST: Online

10.18.6.1 Description
Lists any user categories that a WSRP producer might support. WebCenter Portal
users can use the WLST command mapWSRPProducerUserCategory to map
application roles to a producer’s user category.
10.18.6.2 Syntax

```
listWSRPProducerUserCategories(appName, name, [server, [applicationVersion]])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing WSRP producer.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.18.6.3 Example

The following example displays the categories associated with a WSRP producer named `WSRPSamples`:

```
ws://weblogic/serverConfig> listWSRPProducerUserCategories(appName='webcenter', name='WSRPSamples')
```

User Category Name : categoryTwo
User Category Description : Custom role two.
User Category Mapped Local Roles : None

User Category Name : categoryOne
User Category Description : Custom role one.
User Category Mapped Local Roles : None

10.18.7 mapWSRPProducerUserCategory

Module: Oracle WebCenter Portal

Use with WLST: Online

10.18.7.1 Description

Maps a role that is defined in the named application to a user category supported by a WSRP producer. The user categories may be found using `listWSRPProducerUserCategories`.

10.18.7.2 Syntax

```
mapWSRPProducerUserCategory(appName, name, localRole, producerUserCategory, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing WSRP producer.</td>
</tr>
</tbody>
</table>
Portlet Producers

WebCenter Portal Custom WLST Commands

10.18.7.3 Example
The following example maps the application role `admin` to the WSRP user category `wrsp-admin`:

```
wls:/weblogic/serverConfig> mapWSRPProducerUserCategory(appName='webcenter', name='WSRPProducer1', localRole='admin', producerUserCategory='wrsp-admin')
```

10.18.8 registerPDKJavaProducer
Module: Oracle WebCenter Portal
Use with WLST: Online

10.18.8.1 Description
Creates a connection to an Oracle PDK-Java portlet producer and registers the Oracle PDK-Java producer with a named application.

10.18.8.2 Syntax
```
registerPDKJavaProducer(appName, name, url, [serviceId, proxyHost, proxyPort, subscriberId, sharedKey, timeout, establishSession, externalApp, mapUser, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Connection name. The name must be unique (across all connection types) within the application.</td>
</tr>
<tr>
<td>url</td>
<td>URL for the Oracle PDK-Java producer. Use the following syntax:</td>
</tr>
<tr>
<td></td>
<td><code>http://host_name:port_number/context_root/providers</code></td>
</tr>
<tr>
<td></td>
<td>Where:</td>
</tr>
<tr>
<td></td>
<td>■ host_name is the server where the producer is deployed</td>
</tr>
<tr>
<td></td>
<td>■ port_number is the HTTP Listener port number</td>
</tr>
<tr>
<td></td>
<td>■ context_root is the web application's context root.</td>
</tr>
<tr>
<td></td>
<td>■ providers is static text. The text entered here depends on how the producer is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td><code>http://myHost:7778/myEnterprisePortlets/providers</code></td>
</tr>
</tbody>
</table>
**serviceId**  
Optional. Service ID of the producer.  
PDK-Java enables you to deploy multiple producers under a single adapter servlet. Producers are identified by their unique service ID. A service ID is required only if the service ID is not appended to the URL end point.  
For example, the following URL endpoint requires sample as the service ID:  
http://domain.example.com:7778/axyz/providers  
However, the following URL endpoint, does not require a service ID:  
http://domain.example.com:7778/axyz/providers/sample  
The service ID is used to look up a file called `<service_id>.properties`, which defines the characteristics of the producer, such as whether to display its test page. Use any value to create the service ID.

**proxyHost**  
Optional. Host name or IP address of the proxy server.  
A proxy is required if the application and the remote portlet producer are separated by a firewall and an HTTP proxy is needed for communication with the producer.

**proxyPort**  
Optional. Port number on which the proxy server listens. This argument defaults to 80.

**sharedKey**  
Optional. Shared key used for message authentication with the remote producer. Message authentication ensures that the incoming messages are sent from a host with a shared key. This argument defaults to null.  
The shared key can contain between 10 and 20 alphanumeric characters.

**subscriberId**  
Optional. Consumer’s identifier, if required.  
When a producer is registered with an application, a call is made to the producer. During the call, the consumer (WebCenter Portal or Portal Framework application in this instance) passes the value for subscriberId to the producer. The producer may be coded to use the subscriber ID.

**timeout**  
Optional. Timeout setting for communications with the producer, in seconds. For example, the maximum time the producer may take to register, deregister, or display portlets on portal pages.  
This argument defaults to 30.  
Individual portlets may define their own timeout period, which takes precedence over the value expressed here.

**establishSession**  
Optional. Enable a user session when executing portlets from this producer. Valid values are 1 (true) and 0 (false). The default for this argument is 0.  
When sessions are enabled (1), the server maintains session-specific information, such as the user name. Message authentication uses sessions, so if a shared key is specified, this option should also be enabled. For sessionless communication between the producer and the server, specify 0.

**externalApp**  
Optional. Name of the external application with which to associate the producer. Required if one of this producer’s portlets requires authentication.

**mapUser**  
Optional. Flag indicating whether the mapped user name from the external application should be passed to the producer.  
Valid values are 1 (true) and 0 (false). This argument defaults to 1.
Portlet Producers

10.18.8.3 Example

The following example creates and registers an Oracle PDK-Java producer named JPDKSamples in WebCenter Portal (webcenter):

```
wls:/weblogic/serverConfig> registerPDKJavaProducer(appName='webcenter', name='JPDKSamples', url='http://myhost:9999/jpdk/providers/sample')
```

10.18.9 setPDKJavaProducer

Module: Oracle WebCenter Portal

Use with WLST: Online

10.18.9.1 Description

Edits registration details for an existing PDK-Java producer.

10.18.9.2 Syntax

```
setPDKJavaProducer(appName, name, url, [serviceId, proxyHost, proxyPort, subscriberId, sharedKey, timeout, establishSession, externalApp, mapUser, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing PDK-Java producer.</td>
</tr>
<tr>
<td>url</td>
<td>URL for the Oracle PDK-Java producer. Use the following syntax:</td>
</tr>
<tr>
<td></td>
<td><code>http://host_name:port_number/context_root/providers</code></td>
</tr>
<tr>
<td></td>
<td>Where:</td>
</tr>
<tr>
<td></td>
<td>■ host_name is the server where the producer is deployed</td>
</tr>
<tr>
<td></td>
<td>■ port_number is the HTTP Listener port number</td>
</tr>
<tr>
<td></td>
<td>■ context_root is the Web application’s context root.</td>
</tr>
<tr>
<td></td>
<td>■ providers is static text. The text entered here depends on how</td>
</tr>
<tr>
<td></td>
<td>the producer is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example:</td>
</tr>
<tr>
<td></td>
<td><code>http://myHost:7778/myEnterprisePortlets/providers</code></td>
</tr>
</tbody>
</table>
**Argument** | **Definition**
---|---
`serviceId` | Optional. Service ID of the producer.
PDK-Java enables you to deploy multiple producers under a single adapter servlet. Producers are identified by their unique service ID. A service ID is required only if the service ID is not appended to the URL end point.

For example the following URL endpoint requires `sample` as the service ID:

http://domain.example.com:7778/axyz/providers

However, the following URL endpoint, does not require a service ID:

http://domain.example.com:7778/axyz/providers/sample

The service ID is used to look up a file called `<service_id>.properties`, which defines the characteristics of the producer, such as whether to display its test page. Use any value to create the service ID.

`proxyHost` | Optional. Host name or IP address of the proxy server.
A proxy is required if the application and the remote portlet producer are separated by a firewall and an HTTP proxy is needed for communication with the producer.

`proxyPort` | Optional. Port number on which the proxy server listens.

`subscriberId` | Optional. Consumer’s identifier, if required.
When a producer is registered with an application, a call is made to the producer. During the call, the consumer (WebCenter Portal or Portal Framework application in this instance) passes the value for Subscriber ID to the producer. If the producer does not see the expected value for Subscriber ID, it might reject the registration call.

`sharedKey` | Optional. The shared key is used for message authentication with the remote producer. Message authentication ensures that the incoming messages are sent from a host with a shared key. You should enable sessions using the `sharedKey` argument, as well as the `establishSession` argument.

`timeout` | Optional. Timeout setting for communications with the producer, in seconds. For example, the maximum time the producer may take to register, deregister, or display portlets on portal pages.
Individual portlets may define their own timeout period, which takes precedence over the value expressed here.

`establishSession` | Optional. Enable a user session when executing portlets from this producer. Valid values are 1 (true) and 0 (false). You should enable sessions using the `establishSession` argument, as well as the `sharedKey` argument.

When sessions are enabled (1), the server maintains session-specific information, such as the user name. Message authentication uses sessions, so if a shared key is specified, this option should also be enabled. For sessionless communication between the producer and the server, set to 0.

`externalApp` | Optional. Name of the external application associated with this producer.

`mapUser` | Optional. Flag indicating whether the mapped user name from the external application should be passed to the producer. Valid values are 1 (true) and 0 (false).
Portlet Producers

[489x34]WebCenter Portal Custom WLST Commands

10.18.9.3 Example
The following example changes a PDK-Java producer registered with MyApp to use a proxy server:

```wls:/weblogic/serverConfig> setPDKJavaProducer(appName='MyApp', name='MyProducer', url='http://myhost.com/jpdk/providers/sample', proxyHost='myproxy.com', proxyPort=80)
```

10.18.10 deregisterPDKJavaProducer
Module: Oracle WebCenter Portal
Use with WLST: Online

10.18.10.1 Description
Deregisters an Oracle PDK-Java producer and deletes the associated connection, for a named application.

10.18.10.2 Syntax
deregisterPDKJavaProducer(appName, name, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing PDK-Java producer.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.18.10.3 Example
The following example deregisters the `wc-WebClipping` producer, and deletes the associated connection:

```wls:/weblogic/serverConfig> deregisterPDKJavaProducer(appName='webcenter', name='wc-WebClipping')
```

Already in Domain Runtime Tree
Producer `wc-WebClipping` has been deregistered.
Already in Domain Runtime Tree
`wc-WebClipping` successfully deleted
Already in Domain Runtime Tree
"wc-WebClipping-urlconn" successfully deleted

10.18.11 listPDKJavaProducers
Module: Oracle WebCenter Portal
Use with WLST: Online

10.18.11.1 Description
Lists details for one or more Oracle PDK-Java producers registered with a named application.

10.18.11.2 Syntax
listPDKJavaProducers(appName, [name, verbose, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Optional. Name of an existing PDK-Java portlet producer. If omitted, connection details for all PDK-Java producers configured for this application are listed.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Displays PDK-Java producer connection details in verbose mode. Valid options are 1 (true) and 0 (false). When set to 1, listPDKJavaProducers lists all connection properties. When set to 0, listPDKJavaProducers lists connection names only. This argument defaults to 1. If you set this argument to 0, do not specify the name argument.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.18.11.3 Example
The following example lists all the connection properties (verbose mode) for the JPDKSamples producer:

wls:/weblogic/serverConfig> listPDKJavaProducers(appName='webcenter', name='JPDKSamples', verbose=1)

-------------------------------
wc-WebClipping
Service Id: None
Shared Key: None
External Application Id: None
Subscriber Id: None
URL: http://myhost.com:9999/portalTools/webClipping/providers/webClipping
-------------------------------
wc-OmniPortlet
10.18.12 registerPageletProducer

Module: Oracle WebCenter Portal
Use with WLST: Online

10.18.12.1 Description
Registers a pagelet producer with a named application.

10.18.12.2 Syntax
registerPageletProducer(appName, name, url, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Connection name. The name must be unique (across all connection types) within the application. The name you specify here appears in Composer under the <code>Mash-ups &gt; Pagelet Producers</code> folder (by default).</td>
</tr>
</tbody>
</table>
| url             | URL required to access the server where the Pagelet Producer is deployed. Use the syntax:
  - `protocol://host.domain:port_number/pagelets`
  - The URL must include a fully-qualified domain name. For example:
    - `http://myhost.example.com:7778/pagelets`
  - If pagelets carry secure data, the URL registered must use the `https` protocol. For example:
    - `https://myhost.com:7779/pagelets`
  - **Note:** In WebCenter Portal, if the Pagelet Producer URL is protected by Oracle Access Manager (OAM), the URL to the pagelet catalog must be excluded (mapped directly without access control), or the catalog will appear to be empty when using REST. The pagelet catalog URL is:
    - `http://proxy_host:proxy_port/api/v2/ensemble/pagelets`
| server          | Optional. Name of the managed server where the application is deployed. For example, `WC_Spaces`. Required when applications with the same name are deployed to different servers and also when you have a cluster. |
| applicationVersion | Optional. Version number of the deployed application. Required if more than one version of the application is deployed. |

10.18.12.3 Example
The following example registers a pagelet producer with WebCenter Portal (webcenter):
Portlet Producers

```
registerPageletProducer(appName='webcenter',
name='MyPageletProducer', url='http://myhost.com:7001/pagelets')
```

### 10.18.13 setPageletProducer

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.18.13.1 Description

Edits connection details for an existing pagelet producer.

#### 10.18.13.2 Syntax

```
setPageletProducer(appName, name, [url, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing pagelet producer connection.</td>
</tr>
<tr>
<td>url</td>
<td>Optional. URL required to access the server where the Pagelet Producer is deployed. Use the syntax: <code>protocol://host.domain:port_number/pagelets</code> The URL must include a fully-qualified domain name. For example: <code>http://myhost.example.com:7778/pagelets</code> <strong>Note:</strong> In WebCenter Portal, if the Pagelet Producer URL is protected by Oracle Access Manager (OAM), the URL to the pagelet catalog must be excluded (mapped directly without access control), or the catalog will appear to be empty when using REST. The pagelet catalog URL is: <code>http://proxy_host:proxy_port/api/v2/ensemble/pagelets</code></td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

#### 10.18.13.3 Example

The following example updates administrator user name and password details for an existing pagelet producer connection named `MyPageletProducer`:

```
setPageletProducer(appName='webcenter',
name='MyPageletProducer', url='http://mypagelethost.com:7778/pagelets')
```

### 10.18.14 listPageletProducers

Module: Oracle WebCenter Portal

Use with WLST: Online
10.18.14.1 Description
Lists connection details for one or all pagelet producers registered with a named application.

10.18.14.2 Syntax
listPageletProducers(appName, [name, verbose, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Optional. Name of an existing pagelet producer connection. Use this argument to view connection details for a specific pagelet producer. If omitted, connection details for all pagelet producers configured for this application are listed.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Displays pagelet producer connection details in verbose mode. Valid options are 1 (true) and 0 (false). When set to 1, <code>listPageletProducers</code> lists all connection properties. When set to 0, <code>listPageletProducers</code> lists connection names only. This argument defaults to 1. If you set this argument to 0, do not specify the name argument.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.18.14.3 Example
The following example lists connection names and details for all pagelet producers currently registered for a Portal Framework application named `MyWebCenterApp`:

```
wlst:/weblogic/serverConfig> listPageletProducers(appName='MyWebCenterApp', verbose=1)
```

```
----------------------
MyPageletProducer
----------------------
URL: http://myhost.com:7001/pagelets
----------------------
TestPageletProducer
----------------------
URL: http://testhost.com:7002/pagelets
```

The following example displays details for a single pagelet producer connection named `MyPageletProducer`:

```
wls:/weblogic/serverConfig> listPageletProducers(appName='webcenter', names='MyPageletProducer', verbose=1)
```

```
----------------------
MyPageletProducer
----------------------
```
10.18.15 deregisterPageletProducer

Module: Oracle WebCenter Portal
Use with WLST: Online

10.18.15.1 Description
Deregisters a pagelet producer currently registered with a named application.

10.18.15.2 Syntax
deregisterPageletProducer(appName, name, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing pagelet producer connection.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different</td>
</tr>
<tr>
<td></td>
<td>servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more</td>
</tr>
<tr>
<td></td>
<td>than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.18.15.3 Example
The following example deregisters a pagelet producer connection named MyPageletProducer currently configured for a Portal Framework application named MyWebCenterApp:

    wls:/weblogic/serverConfig> deregisterPageletProducer(appName='MyWebCenterApp', name='MyPageletProducer')

10.18.16 refreshProducer

Module: Oracle WebCenter Portal
Use with WLST: Online

10.18.16.1 Description
Refreshes the metadata stored for a named producer to reflect the portlets that are currently offered by that producer.

10.18.16.2 Syntax
refreshProducer(appName, producerName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
</tbody>
</table>
Portlet Producers

WebCenter Portal Custom WLST Commands

10.18.16.3 Example
The following example refreshes the WSRPSamples producer in WebCenter Portal (webcenter):

```
ws:/weblogic/serverConfig> refreshProducer(appName='webcenter',
producerName='WSRPSamples')
```

Producer WSRPSamples has been refreshed.

10.18.17 listPortletClientConfig
Module: Oracle WebCenter Portal
Use with WLST: Online

10.18.17.1 Description
Lists portlet client’s configuration for a named application.

10.18.17.2 Syntax
```
listPortletClientConfig(appName, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.18.17.3 Examples
The following example lists portlet client configuration for a Portal Framework application named myPortalApp.

```
wls:/weblogic/serverConfig> listPortletClientConfig(appName='myPortalApp')
```

-----------------------------
Portlet Client Configuration Settings
-----------------------------
Application Striping: 0
Content Cache Enabled: 0
Maximum Content Cache Objects: 2000
Maximum Content Cache Size: 10000000
Default Timeout: 30
Maximum IFrame Querystring Length: 0
Maximum Resource URL Length: 1500
Maximum Timeout: 60
Minimum Timeout: 2
Parallel Pool Size: 20
Parallel Queue Size: 40
Maximum Rendition Cache Objects: 1000
Resource Proxy Path: /resource-proxy
Supported Locales: en, de, fr

10.18.18 setPortletClientConfig
Module: Oracle WebCenter Portal
Use with WLST: Online

10.18.18.1 Description
Edits the portlet client's configuration, for a named application. If you omit a parameter, the corresponding configuration setting remains unchanged.

Note: Configuration changes made using this WLST command are only effective after your restart the Managed Server on which the application is deployed. For details, see Oracle Fusion Middleware Administering Oracle WebCenter Portal.

10.18.18.2 Syntax
setPortletClientConfig(appName, [applicationStriping, contentCacheEnabled, contentCacheMaxObjects, contentCacheMaxSize, defaultTimeout, maximumIframeQueryStringLength, maximumResourceUrlLength, maximumTimeout, minimumTimeout, parallelPoolSize, parallelQueueSize, renditionCacheMaxObjects, resourceProxyPath, supportedLocales, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>applicationStriping</td>
<td>Optional. Specifies whether to enable application striping. Valid values are 1 (true) and 0 (false).</td>
</tr>
<tr>
<td></td>
<td>The default value is 0 (false).</td>
</tr>
<tr>
<td>contentCacheEnabled</td>
<td>Optional. Specifies whether to enable the portlet client's content cache. Valid values are 1 (true) and 0 (false).</td>
</tr>
<tr>
<td></td>
<td>The default value is 1 (true).</td>
</tr>
<tr>
<td>contentCacheMaxObjects</td>
<td>Optional. Maximum number of objects to be stored in the portlet client's content cache.</td>
</tr>
<tr>
<td></td>
<td>The default value is 0 (unlimited).</td>
</tr>
<tr>
<td>contentCacheMaxSize</td>
<td>Optional. Maximum size of the portlet client's content cache in bytes.</td>
</tr>
<tr>
<td></td>
<td>The default value is 0 (unlimited).</td>
</tr>
</tbody>
</table>
The following example sets new values for the ContentCacheEnabled, MinimumTimeout, MaximumTimeout, and ParallelQueueSize configuration settings. All other settings remain unchanged:

```wls:
setPortletClientConfig(appName='myPortalApp', contentCacheEnabled=0, minimumTimeout=10, maximumTimeout=120, parallelQueueSize=25)
```

The following example sets the value of the SupportedLocales configuration setting to English, French, and Spanish. All other settings remain unchanged:

```wls:
setPortletClientConfig(appName='myPortalApp', supportedLocales='en,fr,de')
```
supportedLocales='en, fr, es')

10.18.19  getPortletClientConfig

Module: Oracle WebCenter Portal

Use with WLST: Online

10.18.19.1 Description
Returns the value of a specific portlet client configuration setting or values of all settings for a named application.

10.18.19.2 Syntax
getPortletClientConfig(appName, [configSetting, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>configSetting</td>
<td>Optional. Name of the portlet client configuration setting to return. One of the following values: ApplicationStriping, ContentCacheEnabled, ContentCacheMaxObjects, ContentCacheMaxSize, DefaultTimeout, MaximumIframeQueryStringLength, MaximumResourceUrlLength, MaximumTimeout, MinimumTimeout, ParallelPoolSize, ParallelQueueSize, RenditionCacheMaxObjects, ResourceProxyPath, SupportedLocales. Note: The values are case-sensitive. Omit this parameter to return the names and values of all settings.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.18.19.3 Examples
The following example returns the value of the DefaultTimeout configuration setting for a Portal Framework application named myPortalApp:

wls:/weblogic/serverConfig> defaultTimeout =
getPortletClientConfig(appName='myPortalApp', configSetting='DefaultTimeout')

The following example returns of the names and values of all portlet client configuration settings for a Portal Framework application named myPortalApp in a dict:

wls:/weblogic/serverConfig> settingsDict =
getPortletClientConfig(appName='myPortalApp')

10.18.20 registerOOTBProducers

Module: Oracle WebCenter Portal

Use with WLST: Online

10.18.20.1 Description

Registers several out-of-the-box producers with WebCenter Portal or a Portal Framework application: OmniPortlet, Web Clipping, and WSRP Tools

10.18.20.2 Syntax

registerOOTBProducers(producerHost, producerPort, appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>producerHost</td>
<td>Host name or IP address of the server hosting out-of-the-box producers.</td>
</tr>
<tr>
<td>producerPort</td>
<td>Port number for the server hosting out-of-the-box producers.</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the application in which the out-of-the-box producers are to be registered.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.18.20.3 Example

The following example registers out-of-the-box producers in a Portal Framework application named myApp.

wls:/weblogic/serverConfig> registerOOTBProducers(producerHost='myhost.com', producerPort=9999, appName='myApp')

Registering Out-of-the-Box Producers

Registering producers at http://myhost.com:9999

Registering Omniportlet

Created connection wc-OmniPortlet-urlconn
Created connection wc-OmniPortlet
Producer connection wc-OmniPortlet has been registered.

Registering WebClipping
Created connection wc-WebClipping-urlconn
Created connection wc-WebClipping
Producer connection wc-WebClipping has been registered.

Registering WSRP Tools
Created connection wc-WSRPTools-wsconn
Created connection wc-WSRPTools
Producer connection wc-WSRPTools has been registered.

10.18.21 deregisterOOTBProducers
Module: Oracle WebCenter Portal
Use with WLST: Online

10.18.21.1 Description
Deregisters out-of-the-box producers with WebCenter Portal or a Portal Framework application: OmniPortlet, Web Clipping, and WSRP Tools

10.18.21.2 Syntax

deregisterOOTBProducers(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which the out-of-the-box producers are currently registered. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.18.21.3 Example
The following example deregisters out-of-the-box producers, and deletes their associated connections, in a Portal Framework application named myApp.

wls:/weblogic/serverConfig> deregisterOOTBProducers(appName='myApp')

Deregistering Out-of-the-Box Producers

Deregistering OmniPortlet
Producer wc-OmniPortlet has been deregistered.
wc-OmniPortlet successfully deleted
wc-OmniPortlet-urlconn successfully deleted

Deregistering WebClipping
Producer wc-WebClipping has been deregistered.
wc-WebClipping successfully deleted
wc-WebClipping-urlconn successfully deleted
10.18.22 registerSampleProducers
Module: Oracle WebCenter Portal
Use with WLST: Online

10.18.22.1 Description
Registers the sample producers provided with Oracle WebCenter Portal with a named application. There are two sample producers—WSRP Samples and JPDK Samples.

10.18.22.2 Syntax
registerSampleProducers(producerHost, producerPort, appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>producerHost</td>
<td>Host name or IP address of the server hosting the sample producers.</td>
</tr>
<tr>
<td>producerPort</td>
<td>Port number for the server hosting the sample producers.</td>
</tr>
<tr>
<td>appName</td>
<td>Name of the application in which the sample producers are to be registered.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.18.22.3 Example
The following example registers Oracle WebCenter Portal’s sample producers in a Portal Framework application named myApp:

```
wls:/weblogic/serverConfig> registerSampleProducers(producerHost='myhost.com', producerPort=9999, appName='myApp')
```

10.18.23 deregisterSampleProducers
Module: Oracle WebCenter Portal
Use with WLST: Online

10.18.23.1 Description
Deregisters Oracle WebCenter Portal’s sample producers (WSRP Samples and JPDK Samples) from a named application.
10.18.23.2 Syntax

deregisterSampleProducers(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which the sample producers are currently</td>
</tr>
<tr>
<td></td>
<td>registered.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td></td>
<td>If a value is not specified, this argument defaults to webcenter.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different</td>
</tr>
<tr>
<td></td>
<td>servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more</td>
</tr>
<tr>
<td></td>
<td>than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.18.23.3 Example

The following example deregisters sample producers from a Portal Framework application named myApp.

```wls:/weblogic/serverConfig> deregisterSampleProducers(appName='myApp')```

10.19 Proxy Server

Use the commands listed in Table 10–28 to manage proxy server settings used by tools and services, in a named application.

Configuration changes made using these WLST commands are only effective after your restart the Managed Server on which the application is deployed. For details, see Oracle Fusion Middleware Administering Oracle WebCenter Portal.

**Table 10–28 RSS WLST Commands**

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>getWebCenterProxyConfig</td>
<td>Return the proxy host and proxy port used by the tools and services.</td>
<td>Online</td>
</tr>
<tr>
<td>setWebCenterProxyConfig</td>
<td>Specify the proxy host and proxy port used by tools and services.</td>
<td>Online</td>
</tr>
<tr>
<td>unsetWebCenterProxyConfig</td>
<td>Delete proxy host and proxy port settings.</td>
<td>Online</td>
</tr>
<tr>
<td>getRssProxyConfig</td>
<td>Deprecated.</td>
<td>Online</td>
</tr>
<tr>
<td>setRssProxyConfig</td>
<td>Deprecated.</td>
<td>Online</td>
</tr>
<tr>
<td>unsetRssProxyConfig</td>
<td>Deprecated.</td>
<td>Online</td>
</tr>
</tbody>
</table>

10.19.1 getWebCenterProxyConfig

Module: Oracle WebCenter Portal

Use with WLST: Online
10.19.1 Description
Returns the proxy host and proxy port used by RSS news feeds and activity streams in a named application. Depending on your network configuration, proxy details may be required to display external RSS news feeds and external links in activity streams in your application.

10.19.2 Syntax
getWebCenterProxyConfig(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces. Required when applications with the same name are</td>
</tr>
<tr>
<td></td>
<td>deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.19.3 Example
The following example returns the proxy host and proxy port used by tools and services in WebCenter Portal (webcenter):

```
web:/weblogic/serverConfig> getWebCenterProxyConfig(appName='webcenter')
```

10.19.2 setWebCenterProxyConfig
Module: Oracle WebCenter Portal
Use with WLST: Online

10.19.2.1 Description
Specifies the proxy host and proxy port used by RSS news feeds and activity streams, in a named application. Depending on your network configuration, proxy details may be required to display external RSS news feeds and external links in activity streams in your application.

10.19.2.2 Syntax
setWebCenterProxyConfig(appName, proxyHost, proxyPort, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>proxyHost</td>
<td>Host name of the proxy server.</td>
</tr>
<tr>
<td>proxyPort</td>
<td>Port on which the proxy server is running.</td>
</tr>
</tbody>
</table>
10.19.2.3 Example
The following example sets the proxy host and proxy port used by tools and services in WebCenter Portal (webcenter):

```
weblogic/serverConfig> setWebCenterProxyConfig(appName='webcenter',
proxyHost='www-proxy.example.com', proxyPort='80')
```

10.19.3 unsetWebCenterProxyConfig
Module: Oracle WebCenter Portal
Use with WLST: Online

10.19.3.1 Description
Deletes the current proxy host and proxy port settings configured for a named application.

10.19.3.2 Syntax
```
unsetWebCenterProxyConfig(appName, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.19.3.3 Example
The following example deletes the proxy host and proxy port settings configured for WebCenter Portal (webcenter):

```
weblogic/serverConfig> unsetWebCenterProxyConfig(appName='webcenter')
```

10.19.4 getRssProxyConfig
Module: Oracle WebCenter Portal
Use with WLST: Online
10.19.4.1 Description
(Deprecated from 11.1.1.8.0. Use getWebCenterProxyConfig instead.)

Returns the proxy host and proxy port used by tools and services. Depending on your network configuration, proxy details may be required to display external RSS news feeds and external links in activity streams in your application.

10.19.4.2 Syntax
getRssProxyConfig(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.19.4.3 Example
The following example returns the proxy host and proxy port used by tools and services in WebCenter Portal (webcenter):

```
wlst:/weblogic/serverConfig> getRssProxyConfig(appName='webcenter')
```

10.19.5 setRssProxyConfig
Module: Oracle WebCenter Portal
Use with WLST: Online

10.19.5.1 Description
(Deprecated from 11.1.1.8.0. Use setWebCenterProxyConfig instead.)

Specifies the proxy host and proxy port used by tools and services, such as RSS news feeds and activity streams. Depending on your network configuration, proxy details may be required to display external RSS news feeds and external links in activity streams in your application.

10.19.5.2 Syntax
setRssProxyConfig(appName, proxyHost, proxyPort, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>proxyHost</td>
<td>Host name of the proxy server.</td>
</tr>
<tr>
<td>proxyPort</td>
<td>Port on which the proxy server is running.</td>
</tr>
</tbody>
</table>
### Example

The following example sets the proxy host and proxy port used by tools and services in WebCenter Portal (`webcenter`):

```
wlst:/weblogic/serverConfig> setRssProxyConfig(appName='webcenter', proxyHost='www-proxy.example.com', proxyPort='80')
```

### unsetRssProxyConfig

**Module:** Oracle WebCenter Portal

**Use with WLST:** Online

**Description**

(Deprecated from 11.1.1.8.0. Use `unsetWebCenterProxyConfig` instead.)

Deletes the current proxy host and proxy port settings used by tools and services.

**Syntax**

```
unsetRssProxyConfig(appName, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

**Example**

The following example deletes the proxy host and proxy port settings used by tools and services in WebCenter Portal (`webcenter`):

```
wls:/weblogic/serverConfig> unsetRssProxyConfig(appName='webcenter')
```

### Search - Oracle SES Search

Use the commands listed in Table 10–29 to manage Oracle Secure Enterprise Search (SES) connections and other Oracle SES search related properties for WebCenter Portal and Portal Framework applications.
Configuration changes made using these WLST commands are only effective after your restart the Managed Server on which the application is deployed. For details, see *Oracle Fusion Middleware Administering Oracle WebCenter Portal*.

**Table 10–29  Search - Oracle SES WLST Commands**

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>createSESConnection</td>
<td>Create a connection to an Oracle SES instance for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>setSESConnection</td>
<td>Edit an existing Oracle SES search connection.</td>
<td>Online</td>
</tr>
<tr>
<td>listSESConnections</td>
<td>List individual or all Oracle SES search connections that are configured for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>setSearchSESConfig</td>
<td>Configure search settings for an existing Oracle SES search connection.</td>
<td>Online</td>
</tr>
<tr>
<td>listSearchSESConfig</td>
<td>List Oracle SES properties for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>createFederationTrustedEntity</td>
<td>Create a federation trusted entity on an Oracle (SES) instance.</td>
<td>Online</td>
</tr>
<tr>
<td>setSESVersion</td>
<td>Obtains and stores version information for the Oracle SES instance that is currently configured for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>listSESVersion</td>
<td>Returns version information for the Oracle SES instance configured for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>createDefaultSourceGroup</td>
<td>Creates a source group in Oracle SES instance with default document, discussion, and announcement sources.</td>
<td>Online</td>
</tr>
</tbody>
</table>

### 10.20.1 createSESConnection

**Module:** Oracle WebCenter Portal

**Use with WLST:** Online

#### 10.20.1.1 Description

Creates a connection to an Oracle Secure Enterprise Search (SES) instance for a named application.

#### 10.20.1.2 Syntax

```plaintext
createSESConnection(appName, name, url, appUser, appPassword,[default, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Connection name. The name must be unique (across all connection types) within the application.</td>
</tr>
<tr>
<td>url</td>
<td>Web services URL that Oracle Secure Enterprise Search exposes to enable Search requests. Use the format: <a href="http://host:port/search/query/OracleSearch">http://host:port/search/query/OracleSearch</a></td>
</tr>
</tbody>
</table>
Example

The following example creates a new search connection that points to the Oracle SES instance `http://myhost.com:7777/search/query/OracleSearch` and makes this connection the active SES search connection for a Portal Framework application named `myApp`.

```wls:/weblogic/serverConfig>createSESConnection(appName='myApp', name='SESConn1', url='http://myhost.com:7777/search/query/OracleSearch', appUser='wpadmin', appPassword='password', default=1)```

setSESConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

### 10.20.2.1 Description

Edits an existing Oracle Secure Enterprise Search (SES) search connection for a named application.

### 10.20.2.2 Syntax

```setSESConnection(appName, name, [url, appUser, appPassword, default, server, applicationVersion])```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Name of an existing search connection.</td>
</tr>
<tr>
<td>appUser</td>
<td>User name that the application uses to authenticate itself as a trusted</td>
</tr>
<tr>
<td></td>
<td>application to Oracle Secure Enterprise Search so that it may perform</td>
</tr>
<tr>
<td></td>
<td>searches on behalf of WebCenter Portal users.</td>
</tr>
<tr>
<td></td>
<td>The specified user must be present in both the Oracle Identity Management</td>
</tr>
<tr>
<td></td>
<td>server configured for the application and the Oracle Identity Management</td>
</tr>
<tr>
<td></td>
<td>server configured for Oracle SES.</td>
</tr>
<tr>
<td>appPassword</td>
<td>Password for the user name specified.</td>
</tr>
<tr>
<td>default</td>
<td>Optional. Configures search to actively use the search connection. Valid</td>
</tr>
<tr>
<td></td>
<td>options are 1 (true) and 0 (false). Setting to 1 replaces any other search</td>
</tr>
<tr>
<td></td>
<td>connection that is being used. Setting to 0 does not change the current</td>
</tr>
<tr>
<td></td>
<td>search configuration. This argument defaults to 0.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For</td>
</tr>
<tr>
<td></td>
<td>example, <code>WC_Spaces</code>. Required when applications with the same name are</td>
</tr>
<tr>
<td></td>
<td>deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if</td>
</tr>
<tr>
<td></td>
<td>more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

Argument | Definition |
--- | --- |
appName | Name of the application in which to perform this operation. For WebCenter Portal, the application name is always `webcenter`. For Portal Framework applications, specify the appropriate name. |
name | Name of an existing search connection. |
Example

The following example modifies the URL of a search connection named SESConn1 and makes the connection the active Oracle SES search connection for a Portal Framework application named myApp.

```wls
wls:/weblogic/serverConfig> etSESConnection(appName='myApp', name='SESConn1',
appPassword='password', default=1)
```

### listSESConnections

Module: Oracle WebCenter Portal

Use with WLST: Online

#### Description

Lists the names of all Oracle Secure Enterprise Search (SES) search connections configured for a named application.

#### Syntax

```
listSESConnections(appName, [verbose, name, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
</tbody>
</table>
### 10.20.3.3 Examples

The following example displays connection details for all Oracle SES search connections configured for a Portal Framework application named `myApp`.

```
wlst/weblogic/serverConfig> listSESConnections(appName='myApp', verbose=1)
```

The following example displays connection details for an Oracle SES search connection named `SESConn1`.

```
wlst/weblogic/serverConfig> listSESConnections(appName='myApp', verbose=1, name='SESConn1')
```

### 10.20.4 setSearchSESConfig

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.20.4.1 Description

Configures search settings for an existing Oracle Secure Enterprise Search (SES) search connection. If a parameter is not specified it is not modified.

#### 10.20.4.2 Syntax

```
setSearchSESConfig(appName, [connectionName, dataGroup, topNRows, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>connectionName</td>
<td>Optional. Names the connection used for search.</td>
</tr>
</tbody>
</table>
10.20.4.3 Example
The following example specifies that search must use the SES search connection named SESConn1, and to search the data group named group2:

```wls:/weblogic/serverConfig>setSearchSESConfig(appName='webcenter', connectionName='SESConn1', dataGroup='group2', topNRows=200);```

The following example changes the maximum number of results that search returns. No connection name is specified in this example, so this configuration change is applied to the current default (or active) search connection:

```wls:/weblogic/serverConfig>setSearchSESConfig(appName='webcenter', topNRows=500);
Already in Domain Runtime Tree
Restart is needed for the service connection changes to take effect.
```

10.20.5 listSearchSESConfig
Module: Oracle WebCenter Portal
Use with WLST: Online

10.20.5.1 Description
Lists Oracle SES search settings for a named application.

10.20.5.2 Syntax
```
listSearchSESConfig(appName, [server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different</td>
</tr>
<tr>
<td></td>
<td>servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if</td>
</tr>
<tr>
<td></td>
<td>more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.20.5.3 Example
The following example displays Oracle SES search configuration information for WebCenter Portal (webcenter):

```
wlsc:/weblogic/serverConfig> listSearchSESConfig(appName='webcenter')
Already in Domain Runtime Tree
-----------------
Search SES Config
-----------------
  connectionName:  SESConn1
  dataGroup:  group2
  topNRows:  200
```

10.20.6 createFederationTrustedEntity
Module: Oracle WebCenter Portal
Use with WLST: Online

10.20.6.1 Description
Creates a federation trusted entity on an Oracle Secure Enterprise Search (SES) instance for a given entity name and password.

10.20.6.2 Syntax
```
createFederationTrustedEntity(appName, sesUrl, sesPassword, entityName, entityPassword, desc,  [sesSchema, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API.</td>
</tr>
<tr>
<td></td>
<td>Use the format: <a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user (eqsys).</td>
</tr>
<tr>
<td>entityName</td>
<td>Entity name.</td>
</tr>
<tr>
<td>entityPassword</td>
<td>Entity password.</td>
</tr>
<tr>
<td>desc</td>
<td>Short description of the entity.</td>
</tr>
<tr>
<td></td>
<td>Alternatively, specify an empty string ‘’.</td>
</tr>
<tr>
<td>sesSchema</td>
<td>Username for the Oracle SES administrative user. The default value is eqsys.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.20.6.3 Example
The following example creates a federation trusted entity named myEntity on the Oracle SES instance http://myseshost.com:7777/search/query/OracleSearch:

```
createFederationTrustedEntity(appName='webcenter',
  sesUrl='http://myseshost.com:7777/search/api/admin/AdminService',
  sesSchema='eqsys', sesPassword='password', entityName='myEntity',
  entityPassword='password', desc='This is a my sample entity')
```

10.20.7 setSESVersion
Module: Oracle WebCenter Portal
Use with WLST: Online

10.20.7.1 Description
Obtains version information for the Oracle Secure Enterprise Search (SES) instance that is configured for a named application, and stores the version for use by search queries later on.

Version information is used when the Oracle SES connection is set as the default connection for search queries.

10.20.7.2 Syntax
```
setSESVersion(appName, sesUrl, sesSchema, sesPassword, [server,
  applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API.</td>
</tr>
<tr>
<td></td>
<td>Use the format:</td>
</tr>
<tr>
<td></td>
<td><a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesSchema</td>
<td>Username for the Oracle SES administrative user. For example, searchsys.</td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different</td>
</tr>
<tr>
<td></td>
<td>servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if</td>
</tr>
<tr>
<td></td>
<td>more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.20.7.3 Example
The following example obtains version information for the Oracle SES instance http://myhost.com:7777/search/query/OracleSearch and makes this information available to a Portal Framework application named MyApp:

```
setSESVersion(appName='myApp',
  sesUrl='http://myseshost.com:7777/search/api/admin/AdminService',
  sesSchema='searchsys', sesPassword='password')
```
10.20.8 listSESVersion

Module: Oracle WebCenter Portal

Use with WLST: Online

10.20.8.1 Description

Returns the version number of the Oracle Secure Enterprise Search (SES) instance that is being used by a named application.

10.20.8.2 Syntax

listSESVersion(appName, sesUrl, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format: <a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format: <a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesSchema</td>
<td>Username for the Oracle SES administrative user. For example, searchsys.</td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.20.8.3 Example

The following example returns version information for the Oracle SES instance http://myhost.com:7777/search/query/OracleSearch and confirms that the version information is saved in the Portal Framework application named MyApp:

```
wls:/weblogic/serverConfig>listSESVersion(appName='MyApp', sesUrl='http://myseshost.com:7777/search/api/admin/AdminService')
```

```
----------------- SES Version Config
-----------------
url: http://myhost.com:7777/search/api/admin/AdminService
version: 11.1.2.2.0
```
10.20.9 createDefaultSourceGroup

Module: Oracle WebCenter Portal
Use with WLST: Online

10.20.9.1 Description
Creates a source group in an Oracle Secure Enterprise Search (SES) instance with a default name, for a given host, port and application name, as well as the shared documents and discussions/announcement sources.

10.20.9.2 Syntax
createDefaultSourceGroup(appName, sesUrl, sesPassword, host, port, defaultHost, defaultPort, [sesSchema, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format: <a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user.</td>
</tr>
<tr>
<td>host</td>
<td>Host name of the machine where the application is running.</td>
</tr>
<tr>
<td>port</td>
<td>Port number used to access the application.</td>
</tr>
<tr>
<td>defaultHost</td>
<td>Host name of the machine where the default sources are configured.</td>
</tr>
<tr>
<td>defaultPort</td>
<td>Port number used where the default sources are configured.</td>
</tr>
<tr>
<td>sesSchema</td>
<td>Optional. Username for the Oracle SES administrative user. Default value is eqsys.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.20.9.3 Example
The following example creates a default source group in the Oracle SES instance http://myhost.com:7777/search/query/OracleSearch for the Portal Framework application named myApp at: http://webhost:8888/webcenter

The default source group also includes documents and discussions sources at: http://mydefaultHost:8989

wlsh:/weblogic/serverConfig>createDefaultSourceGroup(appName='myApp', sesUrl='http://myseshost.com:7777/search/api/admin/AdminService', sesSchema='eqsys', sesPassword='password', host='webhost', port='8888', defaultHost='mydefaultHost', defaultPort='8989')
10.21 **Search - Oracle SES Search Crawlers**

Use the commands listed in [Table 10–30](#) to manage Oracle Secure Enterprise Search (SES) crawlers for WebCenter Portal and Portal Framework applications.

There is no need to restart your application after running Oracle SES crawler WLST commands.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>createSpacesCrawler</td>
<td>Create a crawler for WebCenter Portal objects on an Oracle SES instance.</td>
<td>Online</td>
</tr>
<tr>
<td>createDocumentsCrawler</td>
<td>Create a documents crawler for a named application, on an Oracle SES instance.</td>
<td>Online</td>
</tr>
<tr>
<td>createDiscussionsCrawler</td>
<td>Create a discussions crawlers and an announcement crawler for a named application, on an Oracle SES instance.</td>
<td>Online</td>
</tr>
<tr>
<td>listSpacesCrawler</td>
<td>Return the Spaces crawler configured for WebCenter Portal, on an Oracle SES instance.</td>
<td>Online</td>
</tr>
<tr>
<td>listDocumentsCrawler</td>
<td>Return the documents crawler configured for a named application, on an Oracle SES instance.</td>
<td>Online</td>
</tr>
<tr>
<td>listDiscussionsCrawler</td>
<td>Return the discussion and announcement crawlers configured for a named application, on an Oracle SES instance.</td>
<td>Online</td>
</tr>
<tr>
<td>startSpacesCrawler</td>
<td>Start the Spaces crawler configured for WebCenter Portal, on an Oracle SES instance.</td>
<td>Online</td>
</tr>
<tr>
<td>startDocumentsCrawler</td>
<td>Start the documents crawler configured for a named application, on an Oracle SES instance.</td>
<td>Online</td>
</tr>
<tr>
<td>startDiscussionsCrawler</td>
<td>Start the discussion and announcement crawlers configured for a named application, on an Oracle SES instance.</td>
<td>Online</td>
</tr>
<tr>
<td>stopSpacesCrawler</td>
<td>Stop the Spaces crawler configured for WebCenter Portal, on an Oracle SES instance.</td>
<td>Online</td>
</tr>
<tr>
<td>stopDocumentsCrawler</td>
<td>Stop the documents crawler configured for a named application, on an Oracle SES instance.</td>
<td>Online</td>
</tr>
<tr>
<td>stopDiscussionsCrawler</td>
<td>Stop discussion and announcement crawlers configured for a named application, on an Oracle SES instance.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteSpacesCrawler</td>
<td>Delete the Spaces crawler configured for WebCenter Portal, on an Oracle SES instance.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteDocumentsCrawler</td>
<td>Delete the documents crawler configured for a named application, on an Oracle SES instance.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteDiscussionsCrawler</td>
<td>Delete discussion and announcement crawlers configured for a named application, on an Oracle SES instance.</td>
<td>Online</td>
</tr>
</tbody>
</table>

### 10.21.1 createSpacesCrawler

**Module:** Oracle WebCenter Portal

**Use with WLST:** Online
10.21.1.1 Description
Creates a crawler for WebCenter Portal objects on an Oracle SES instance. The command creates a WebCenter Portal datasource and specifies a schedule for crawling WebCenter Portal objects (such as portals, lists, pages, and people).

10.21.1.2 Syntax
createSpacesCrawler(appName,  host, port, sesUrl, sesPassword, crawlUser, crawlPassword, scratchDir, authUserIdFormat, crawlingMode, recrawlPolicy, freqType, startHour, hoursBetweenLaunches, startDayOfWeek, startDayOfMonth, daysBetweenLaunches, weeksBetweenLaunches, monthsBetweenLaunches, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>host</td>
<td>Host name of the machine where WebCenter Portal is running.</td>
</tr>
<tr>
<td>port</td>
<td>Port number used to access WebCenter Portal.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format:</td>
</tr>
<tr>
<td></td>
<td><a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user (eqsys).</td>
</tr>
<tr>
<td>crawlUser</td>
<td>Crawl administration user in WebCenter Portal.</td>
</tr>
<tr>
<td></td>
<td>This user must exist in WebCenter Portal and in your back-end identity</td>
</tr>
<tr>
<td></td>
<td>management server with appropriate permissions and roles. For example:</td>
</tr>
<tr>
<td></td>
<td>mycrawladmin</td>
</tr>
<tr>
<td>crawlPassword</td>
<td>Password for the WebCenter Portal user that is specified in the crawlUser argument.</td>
</tr>
<tr>
<td>scratchDir</td>
<td>Local directory where Oracle SES can write temporary status logs. The</td>
</tr>
<tr>
<td></td>
<td>directory must be on the system where Oracle SES is installed.</td>
</tr>
<tr>
<td>authUserIdFormat</td>
<td>Format of the user ID in the active identity plug-in. For example,</td>
</tr>
<tr>
<td></td>
<td>username, email, nickname, user_name.</td>
</tr>
<tr>
<td>crawlingMode</td>
<td>Mode for crawling URLs in the source. The default mode is ACCEPT_ALL. Valid</td>
</tr>
<tr>
<td></td>
<td>values are: ACCEPT_ALL, INDEX_ONLY, EXAMINE_URL:</td>
</tr>
<tr>
<td></td>
<td>ACCEPT_ALL—Automatically Accept All URLs for Indexing; Crawls and indexes</td>
</tr>
<tr>
<td></td>
<td>all URLs in the source. It also extracts and indexes any links found in</td>
</tr>
<tr>
<td></td>
<td>those URLs. Previously crawled URLs are only reindexed if they have changed.</td>
</tr>
<tr>
<td></td>
<td>EXAMINE_URL—Examine URLs Before Indexing: Crawls but does not index any</td>
</tr>
<tr>
<td></td>
<td>URLs in the source. It also crawls any links found in those URLs.</td>
</tr>
<tr>
<td></td>
<td>INDEX_ONLY—Index Only: Crawls and indexes all URLs in the source. It does</td>
</tr>
<tr>
<td></td>
<td>not extract any links found in those URLs. Select this option for a source</td>
</tr>
<tr>
<td></td>
<td>previously crawled using EXAMINE_URL.</td>
</tr>
<tr>
<td>recrawlPolicy</td>
<td>Specifies whether to crawl all documents or only documents that have</td>
</tr>
<tr>
<td></td>
<td>changed. Valid values are PROCESS_ALL and PROCESS_CHANGED:</td>
</tr>
<tr>
<td></td>
<td>PROCESS_ALL—All documents are crawled. Use this option to force a full</td>
</tr>
<tr>
<td></td>
<td>crawl.</td>
</tr>
<tr>
<td></td>
<td>PROCESS_CHANGED—Only crawl documents that have changed since the previous</td>
</tr>
<tr>
<td></td>
<td>crawl. This setting can significantly speed up the crawling process.</td>
</tr>
</tbody>
</table>
**Example**

The following example creates a Spaces crawler on the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal (webcenter) located at http://myhost.com:8888/webcenter/portal:

```shell
createSpacesCrawler(appName='webcenter', host='myhost.com', port='8888',
    sesUrl='http://myseshost.com:7777/search/api/admin/AdminService',
    sesPassword='sespassword', crawlUser='mycrawladmin', crawlPassword='password',
    scratchDir='/tmp', authUserIDFormat='username', crawlingMode='ACCEPT_ALL',
    recrawlPolicy='PROCESS_ALL', freqType='MANUAL', startHour=1,
    hoursBetweenLaunches=1, startDayOfWeek='MONDAY', startDayOfMonth=1,
    daysBetweenLaunches =1, weeksBetweenLaunches=1, monthsBetweenLaunches=1)
```

**Argument Definition**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>freqType</strong></td>
<td>Frequency of scheduled crawls. Valid values are: MANUAL, MONTHLY, WEEKLY, DAILY, HOURLY. To schedule crawls MONTHLY, WEEKLY, DAILY, or HOURLY, specify additional arguments as follows: <strong>MONTHLY</strong>: startHour, startDayOfTheMonth, monthsBetweenLaunches <strong>WEEKLY</strong>: startHour, startDayOfTheWeek, weeksBetweenLaunches <strong>DAILY</strong>: startHour, daysBetweenLaunches <strong>HOURLY</strong>: hoursBetweenLaunches If regular crawls are not required, choose MANUAL and then use the startSpacesCrawler command to initiate a crawl manually.</td>
</tr>
<tr>
<td><strong>startHour</strong></td>
<td>Time to start the crawl. Any number between 1 and 24. For example, enter 2 for 2:00am, 14 for 2:00pm, and so on.</td>
</tr>
<tr>
<td><strong>hoursBetweenLaunches</strong></td>
<td>Number of hours between crawls. Only valid when freqType='HOURLY'.</td>
</tr>
<tr>
<td><strong>startDayOfWeek</strong></td>
<td>Day on which to start a weekly crawl. For example, MONDAY, TUESDAY, and so on. Only valid when freqType='WEEKLY'.</td>
</tr>
<tr>
<td><strong>startDayOfMonth</strong></td>
<td>Day of the month on which to start a monthly crawl. For example, enter 1 for 1st day of the month, 2 for 2nd day of the month, and so on. Only valid when freqType='MONTHLY'.</td>
</tr>
<tr>
<td><strong>daysBetweenLaunches</strong></td>
<td>Number of days between crawls. Only valid when freqType='DAILY'.</td>
</tr>
<tr>
<td><strong>weeksBetweenLaunches</strong></td>
<td>Number of weeks between crawls. Only valid when freqType='WEEKLY'.</td>
</tr>
<tr>
<td><strong>monthsBetweenLaunches</strong></td>
<td>Number of months between crawls. Only valid when freqType='MONTHLY'.</td>
</tr>
<tr>
<td><strong>server</strong></td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><strong>applicationVersion</strong></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.21.2 createDocumentsCrawler

Module: Oracle WebCenter Portal

Use with WLST: Online

10.21.2.1 Description

Creates a documents crawler for a named application, on an Oracle SES instance.

The command creates an Oracle WebCenter Content repository datasource and specifies a schedule for crawling documents in the Oracle WebCenter Content repository.

10.21.2.2 Syntax

createDocumentsCrawler(appName, host, port, sesUrl, sesPassword, configUrl, user, password, scratchDir, httpEndpoint, displayUrl, realm, authUserIdFormat, pipelineName, crawlingMode, recrawlPolicy, freqType, startHour, hoursBetweenLaunches, startDayOfWeek, startDayOfMonth, daysBetweenLaunches, weeksBetweenLaunches, monthsBetweenLaunches, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>host</td>
<td>Host name of the machine where the application is running.</td>
</tr>
<tr>
<td>port</td>
<td>Port number used to access the application.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format:</td>
</tr>
<tr>
<td></td>
<td><a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user (eqsys).</td>
</tr>
<tr>
<td>configUrl</td>
<td>URL of the XML configuration file providing details of the source, such as</td>
</tr>
<tr>
<td></td>
<td>the data feed type, location, security attributes, and so on.</td>
</tr>
<tr>
<td></td>
<td>Use the URL format:</td>
</tr>
<tr>
<td>user</td>
<td>Administrative user for Oracle WebCenter Content's Content Server.</td>
</tr>
<tr>
<td></td>
<td>For example, sysadmin.</td>
</tr>
<tr>
<td></td>
<td>If the authentication type is Oracle SSO, then enter a user ID (and password)</td>
</tr>
<tr>
<td></td>
<td>of a user in the identity management server fronted by Oracle SSO. This user</td>
</tr>
<tr>
<td></td>
<td>must be granted the same permissions as sysadmin. If it is not possible to</td>
</tr>
<tr>
<td></td>
<td>grant those permissions, then delete the &quot;remote&quot; user corresponding to this</td>
</tr>
<tr>
<td></td>
<td>user in Content Server, and create a &quot;local&quot; version of the user (same name)</td>
</tr>
<tr>
<td></td>
<td>in Content Server.</td>
</tr>
<tr>
<td>password</td>
<td>Password for the administrative user specified.</td>
</tr>
<tr>
<td>scratchDir</td>
<td>Local directory where Oracle SES can write temporary status logs. The</td>
</tr>
<tr>
<td></td>
<td>directory must be on the system where Oracle SES is installed.</td>
</tr>
<tr>
<td>httpEndpoint</td>
<td>HTTP endpoint for Content Server authorization. For example:</td>
</tr>
<tr>
<td></td>
<td><a href="http://host:port/idc/idcplg">http://host:port/idc/idcplg</a></td>
</tr>
<tr>
<td>Argument</td>
<td>Definition</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>displayUrl</td>
<td>HTTP host information string to prefix the relative access URL to form the complete display URL. For example: <a href="http://host:port/idc">http://host:port/idc</a></td>
</tr>
<tr>
<td>realm</td>
<td>Realm of the application serving the control and data feed. This parameter is relevant when the feeds are accessed over HTTP and is mandatory when the authentication type is BASIC. For example, jazn.com</td>
</tr>
<tr>
<td>authUserIdFormat</td>
<td>Format of the user ID (in active identity plug-in) that is used by Content Server authorization API. For example, username, email, nickname, user_name.</td>
</tr>
<tr>
<td>pipelineName</td>
<td>Document service pipeline created for this source in Oracle SES.</td>
</tr>
<tr>
<td>crawlingMode</td>
<td>Mode for crawling URLs in the source. The default mode is ACCEPT_ALL. Valid values are: ACCEPT_ALL, INDEX_ONLY, EXAMINE_URL:</td>
</tr>
<tr>
<td></td>
<td>ACCEPT_ALL—Automatically Accept All URLs for Indexing: Crawls and indexes all URLs in the source. It also extracts and indexes any links found in those URLs. Previously crawled URLs are only reindexed if they have changed.</td>
</tr>
<tr>
<td></td>
<td>EXAMINE_URL—Examine URLs Before Indexing: Crawls but does not index any URLs in the source. It also crawls any links found in those URLs.</td>
</tr>
<tr>
<td></td>
<td>INDEX_ONLY—Index Only: Crawls and indexes all URLs in the source. It does not extract any links found in those URLs. Select this option for a source previously crawled using EXAMINE_URL</td>
</tr>
<tr>
<td>recrawlPolicy</td>
<td>Specifies whether to crawl all documents or only documents that have changed. Valid values are: PROCESS_ALL and PROCESS_CHANGED:</td>
</tr>
<tr>
<td></td>
<td>PROCESS_ALL—All documents are crawled. Use this option to force a full crawl.</td>
</tr>
<tr>
<td></td>
<td>PROCESS_CHANGED—Only crawl documents that have changed since the previous crawl. This setting can significantly speed up the crawling process.</td>
</tr>
<tr>
<td>freqType</td>
<td>Frequency of scheduled crawls. Valid values are: MANUAL, MONTHLY, WEEKLY, DAILY, HOURLY.</td>
</tr>
<tr>
<td></td>
<td>To schedule crawls MONTHLY, WEEKLY, DAILY, or HOURLY, specify additional arguments as follows:</td>
</tr>
<tr>
<td></td>
<td>MONTHLY: startHour, startDayOfTheMonth, monthsBetweenLaunches</td>
</tr>
<tr>
<td></td>
<td>WEEKLY: startHour, startDayOfTheWeek, weeksBetweenLaunches</td>
</tr>
<tr>
<td></td>
<td>DAILY: startHour, daysBetweenLaunches</td>
</tr>
<tr>
<td></td>
<td>HOURLY: hoursBetweenLaunches</td>
</tr>
<tr>
<td></td>
<td>If regular crawls are not required, choose MANUAL and then use the startDocumentsCrawler command to initiate a crawl manually.</td>
</tr>
<tr>
<td>startHour</td>
<td>Time to start the crawl. Any number between 1 and 24.</td>
</tr>
<tr>
<td></td>
<td>For example, enter 2 for 2:00am, 14 for 2:00pm, and so on.</td>
</tr>
<tr>
<td>hoursBetweenLaunches</td>
<td>Number of hours between crawls. Only valid when freqType='HOURLY'.</td>
</tr>
<tr>
<td>startDayOfWeek</td>
<td>Day on which to start a weekly crawl. For example, MONDAY, TUESDAY, and so on.</td>
</tr>
<tr>
<td></td>
<td>Only valid when freqType='WEEKLY'.</td>
</tr>
</tbody>
</table>
The following example creates a documents crawler on the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal (webcenter) located at http://myhost.com:8888/webcenter/portal:

createDocumentsCrawler(appName='webcenter', host='myhost.com', port='8888',
sesUrl='http://myseshost.com:7777/search/api/admin/AdminService',
sesPassword='password',
user='adminuser', password='password', scratchDir='/scratch',
httpEndpoint='http://myucmhost.com:9044/cs/idcplg',
displayUrl='http://myucmhost:9044/cs', realm='jazn.com',
authUserIdFormat='username',
pipelineName='My UCM Pipeline', crawlingMode='ACCEPT_ALL',
recrawlPolicy='PROCESS_ALL', freqType='MANUAL', startHour=1,
hoursBetweenLaunches=1, startDayOfWeek='MONDAY', startDayOfMonth=1,
daysBetweenLaunches=1, weeksBetweenLaunches=1, monthsBetweenLaunches=1)

10.21.3 createDiscussionsCrawler

Module: Oracle WebCenter Portal

Use with WLST: Online

10.21.3.1 Description
Creates a discussion forum crawler and an announcements crawler for a named application, on an Oracle Secure Enterprise Search (SES) instance.

The command creates two Oracle SES database sources (one for discussion forums and one for announcements) and specifies a crawl schedule. The discussion forums source is named <appname_host_port>_forums with a view of FORUMCRAWLER_VW, and the announcements source is named <appname_host_port>_announcements with a view of ANNOUNCEMENTS_VW.
10.21.3.2 Syntax

createDiscussionsCrawler(appName, host, port, sesUrl, sesPassword, dbConnString, user, password, authUserIdFormat, crawlingMode, recrawlPolicy, freqType, startHour, hoursBetweenLaunches, startDayOfWeek, startDayOfMonth, daysBetweenLaunches, weeksBetweenLaunches, monthsBetweenLaunches, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>host</td>
<td>Host name of the machine where the application is running.</td>
</tr>
<tr>
<td>port</td>
<td>Port number used to access the application.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format: <code>http://host:port/search/api/admin/AdminService</code></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user (eqsys).</td>
</tr>
</tbody>
</table>
| dbConnString | Connection URL for the database on which the discussions server is installed. Use the format:  
  **Oracle**: `jdbc:oracle:thin:@<host>:<port>/<oracle-sid>`  
  **IBM DB2**: `jdbc:db2://<host>:<port>/<database_name>`  
  **Microsoft SQL Server**: `jdbc:sqlserver://<host_or_IP_address>:<port>/<database_name>` |
| user        | Administrative user for the database on which the discussions server is installed.  
  **Oracle**: The user `MyPrefix_DISCUSSIONS_CRAWLER` is created during installation.  
  **IBM DB2**: The user `MyPrefix_DC` is created during installation (where `MyPrefix` is five characters)  
  **Microsoft SQL Server**: The user `MyPrefix_DISCUSSIONS_CRAWLER` is created during installation. |
| password    | Password for the administrative discussions server user specified. |
| authUserIdFormat | Format of the user ID (in active identity plug-in), that is used by the discussions server authorization API. For example, username, email, nickname, user_name. |
| crawlingMode | Mode for crawling URLs in the source. The default mode is `ACCEPT_ALL`. Valid values are: `ACCEPT_ALL`, `INDEX_ONLY`, `EXAMINE_URL`  
  **ACCEPT_ALL**—Automatically Accept All URLs for Indexing: Crawls and indexes all URLs in the source. It also extracts and indexes any links found in those URLs. Previously crawled URLs are only reindexed if they have changed.  
  **EXAMINE_URL**—Examine URLs Before Indexing: Crawls but does not index any URLs in the source. It also crawls any links found in those URLs.  
  **INDEX_ONLY**—Index Only: Crawls and indexes all URLs in the source. It does not extract any links found in those URLs. Select this option for a source previously crawled using `EXAMINE_URL`. |
The following example creates a discussion forum crawler and an announcements crawler on the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal (webcenter) located at http://myhost.com:8888/webcenter/portal:

createDiscussionsCrawler(appName='webcenter', host='myhost.com', port='8888',
recrawlPolicy Specifies whether to crawl all documents or only documents that have changed. Valid values are PROCESS_ALL and PROCESS_CHANGED:

PROCESS_ALL—All documents are crawled. Use this option to force a full crawl.

PROCESS_CHANGED—Only crawl documents that have changed since the previous crawl. This setting can significantly speed up the crawling process.

freqType Frequency of scheduled crawls. Valid values are: MANUAL, MONTHLY, WEEKLY, DAILY, HOURLY.

To schedule crawls MONTHLY, WEEKLY, DAILY, or HOURLY, specify additional arguments as follows:

MONTHLY: startHour, startDayOfMonth, monthsBetweenLaunches

WEEKLY: startHour, startDayOfWeek, weeksBetweenLaunches

DAILY: startHour, daysBetweenLaunches

HOURLY: hoursBetweenLaunches

If regular crawls are not required, choose MANUAL and then use the startDiscussionsCrawler command to initiate a crawl manually.

startHour Time to start the crawl. Any number between 1 and 24. For example, enter 2 for 2:00am, 14 for 2:00pm, and so on.

hoursBetweenLaunches Number of hours between crawls. Only valid when freqType='HOURLY'.

startDayOfWeek Day on which to start a weekly crawl. For example, MONDAY, TUESDAY, and so on. Only valid when freqType='WEEKLY'.

startDayOfMonth Day of the month on which to start a monthly crawl. For example, enter 1 for 1st day of the month, 2 for 2nd day of the month, and so on. Only valid when freqType='MONTHLY'.

daysBetweenLaunches Number of days between crawls. Only valid when freqType='DAILY'.

weeksBetweenLaunches Number of weeks between crawls. Only valid when freqType='WEEKLY'.

monthsBetweenLaunches Number of months between crawls. Only valid when freqType='MONTHLY'.

server Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.

applicationVersion Optional. Version number of the deployed application. Required if more than one version of the application is deployed.

### 10.21.3.3 Example

The following example creates a discussion forum crawler and an announcements crawler on the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal (webcenter) located at http://myhost.com:8888/webcenter/portal:

createDiscussionsCrawler(appName='webcenter', host='myhost.com', port='8888',

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>recrawlPolicy</td>
<td>Specifies whether to crawl all documents or only documents that have changed. Valid values are PROCESS_ALL and PROCESS_CHANGED:</td>
</tr>
<tr>
<td>freqType</td>
<td>Frequency of scheduled crawls. Valid values are: MANUAL, MONTHLY, WEEKLY, DAILY, HOURLY.</td>
</tr>
<tr>
<td>startHour</td>
<td>Time to start the crawl. Any number between 1 and 24. For example, enter 2 for 2:00am, 14 for 2:00pm, and so on.</td>
</tr>
<tr>
<td>hoursBetweenLaunches</td>
<td>Number of hours between crawls. Only valid when freqType='HOURLY'.</td>
</tr>
<tr>
<td>startDayOfWeek</td>
<td>Day on which to start a weekly crawl. For example, MONDAY, TUESDAY, and so on. Only valid when freqType='WEEKLY'.</td>
</tr>
<tr>
<td>startDayOfMonth</td>
<td>Day of the month on which to start a monthly crawl. For example, enter 1 for 1st day of the month, 2 for 2nd day of the month, and so on. Only valid when freqType='MONTHLY'.</td>
</tr>
<tr>
<td>daysBetweenLaunches</td>
<td>Number of days between crawls. Only valid when freqType='DAILY'.</td>
</tr>
<tr>
<td>weeksBetweenLaunches</td>
<td>Number of weeks between crawls. Only valid when freqType='WEEKLY'.</td>
</tr>
<tr>
<td>monthsBetweenLaunches</td>
<td>Number of months between crawls. Only valid when freqType='MONTHLY'.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
sesUrl='http://mytheshost.com:7777/search/api/admin/AdminService',
sesPassword='password',
dbConnString='jdbc:oracle:thin:@myjivedbhost.com:1521/mysid',
user='app_discussions_crawler', password='password',
authUserIdFormat='nickname', crawlingMode='ACCEPT_ALL',
recrawlPolicy='PROCESS_ALL', freqType='MANUAL', startHour=1,
hoursBetweenLaunches=1, startDayOfWeek='MONDAY',
startDayOfMonth=1, daysBetweenLaunches=1,
weeksBetweenLaunches=1, monthsBetweenLaunches=1)

10.21.4 listSpacesCrawler

Module: Oracle WebCenter Portal

Use with WLST: Online

10.21.4.1 Description

Returns the Spaces crawler configured on an Oracle SES instance for WebCenter Portal.

10.21.4.2 Syntax

listSpacesCrawler(appName, sesUrl, sesPassword, host, port, [sesSchema, verbose, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format: <a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user (eqsys).</td>
</tr>
<tr>
<td>host</td>
<td>Host name of the machine where WebCenter Portal is running.</td>
</tr>
<tr>
<td>port</td>
<td>Port number used to access WebCenter Portal.</td>
</tr>
<tr>
<td>sesSchema</td>
<td>Optional. Username for the Oracle SES administrative user. Default value is eqsys.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Valid options are 1 (true) and 0 (false). When set to 1, listSpacesCrawler returns the Spaces crawler configured for WebCenter Portal in Oracle SES, along with details. When set to 0, only source names are listed. This argument defaults to 0.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where WebCenter Portal is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
### 10.21.4.3 Example

The following example returns the Spaces crawler configured in the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal located at http://myhost.com:8888/webcenter/portal:

```python
listSpacesCrawler(appName='webcenter', 
sesUrl='http://myseshost.com:7777/search/api/admin/AdminService', 
sesSchema='eqsys', sesPassword='password', host='myhost.com', port='8888')
```

Already in Domain Runtime Tree
-----------------------------
Spaces Crawlers
-----------------------------
webcenter_myhost.com_8888_portal

### 10.21.5 listDocumentsCrawler

Module: Oracle WebCenter Portal

Use with WLST: Online

#### 10.21.5.1 Description

Returns the document crawler configured for a named application, on an Oracle SES instance.

#### 10.21.5.2 Syntax

```python
listDocumentsCrawler(appName, sesUrl, sesPassword, host, port, [sesSchema, 
verbose, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format: <a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user (eqsys).</td>
</tr>
<tr>
<td>host</td>
<td>Host name of the machine where the application is running.</td>
</tr>
<tr>
<td>port</td>
<td>Port number used to access the application.</td>
</tr>
<tr>
<td>sesSchema</td>
<td>Optional. Username for the Oracle SES administrative user. Default value is eqsys.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Valid options are 1 (true) and 0 (false). When set to 1, listDocumentsCrawler returns the documents crawler that is configured for the application in Oracle SES, along with details. When set to 0, only source names are listed. This argument defaults to 0.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
</tbody>
</table>
10.21.5.3 Example
The following example returns the documents crawler configured in the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal (webcenter) located at http://myhost.com:8888/webcenter/portal:

listDocumentsCrawler(appName='webcenter', sesUrl='http://myseshost.com:7777/search/api/admin/AdminService', sesSchema='eqsys', sesPassword='password', host='myhost.com', port='8888')

Already in Domain Runtime Tree
-----------------
Documents Crawlers
-----------------
webcenter_myhost.com_8888_documents

10.21.6 listDiscussionsCrawler
Module: Oracle WebCenter Portal
Use with WLST: Online

10.21.6.1 Description
Returns the discussion and announcement crawlers configured for a named application on an Oracle SES instance.

10.21.6.2 Syntax
listDiscussionsCrawler(appName, sesUrl, sesPassword, host, port, [sesSchema, verbose, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format:</td>
</tr>
<tr>
<td></td>
<td><a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user (eqsys).</td>
</tr>
<tr>
<td>host</td>
<td>Host name of the machine where the application is running.</td>
</tr>
<tr>
<td>port</td>
<td>Port number used to access the application.</td>
</tr>
<tr>
<td>sesSchema</td>
<td>Optional. Username for the Oracle SES administrative user. Default value</td>
</tr>
<tr>
<td></td>
<td>is eqsys.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Valid options are 1 (true) and 0 (false). When set to 1,</td>
</tr>
<tr>
<td></td>
<td>listDiscussionsCrawler returns discussion and announcement crawlers that</td>
</tr>
<tr>
<td></td>
<td>are configured in Oracle SES for an application, along with details. When</td>
</tr>
<tr>
<td></td>
<td>set to 0, only source names are listed. This argument defaults to 0.</td>
</tr>
</tbody>
</table>
The following example returns discussion and announcement crawlers configured in the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal located at http://myhost.com:8888/webcenter/portal:

```
listDiscussionsCrawler(appName='webcenter',
sesUrl='http://myseshost.com:7777/search/api/admin/AdminService',
sesSchema='eqsys', sesPassword='password', host='myhost.com', port='8888')
```

Already in Domain Runtime Tree
-----------------------------
Discussions Crawler
-----------------------------
webcenter_myhost.com_8888_forums
webcenter_myhost.com_8888_announcements

### 10.21.7 startSpacesCrawler

**Module:** Oracle WebCenter Portal

**Use with WLST:** Online

#### 10.21.7.1 Description

Starts the Spaces crawler configured for WebCenter Portal, on an Oracle SES instance.

#### 10.21.7.2 Syntax

```
startSpacesCrawler(appName, sesUrl, sesPassword, host, port, [sesSchema, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format: <a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user (eqsys).</td>
</tr>
<tr>
<td>host</td>
<td>Host name of the machine where WebCenter Portal is running.</td>
</tr>
<tr>
<td>port</td>
<td>Port number used to access WebCenter Portal.</td>
</tr>
<tr>
<td>sesSchema</td>
<td>Optional. Username for the Oracle SES administrative user. Default value is eqsys.</td>
</tr>
</tbody>
</table>
10.21.7.3 Example
The following example starts the Spaces crawler configured on the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal (webcenter) located at http://myhost.com:8888/webcenter/portal:

```
startSpacesCrawler(appName='webcenter',
sesUrl='http://myseshost.com:7777/search/api/admin/AdminService',
sesSchema='eqsys', sesPassword='password', host='myhost.com', port='8888')
```

10.21.8 startDocumentsCrawler
Module: Oracle WebCenter Portal
Use with WLST: Online

10.21.8.1 Description
Starts the documents crawler configured for a named application, on an Oracle SES instance.

10.21.8.2 Syntax

```
startDocumentsCrawler(appName, sesUrl, sesPassword, host, port, [sesSchema, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format: <a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user (eqsys).</td>
</tr>
<tr>
<td>host</td>
<td>Host name of the machine where the application is running.</td>
</tr>
<tr>
<td>port</td>
<td>Port number used to access the application.</td>
</tr>
<tr>
<td>sesSchema</td>
<td>Optional. Username for the Oracle SES administrative user. Default value is eqsys.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.21.8.3 Example
The following example starts the document crawler configured on the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal (webcenter) located at http://myhost.com:8888/webcenter/portal:

startDocumentsCrawler(appName='webcenter',
sesUrl='http://myseshost.com:7777/search/api/admin/AdminService',
sesSchema='eqsys', sesPassword='password', host='myhost.com', port='8888')

10.21.9 startDiscussionsCrawler
Module: Oracle WebCenter Portal
Use with WLST: Online

10.21.9.1 Description
Starts the discussion and announcement crawlers configured for a named application, on an Oracle Secure Enterprise Search (SES) instance.

10.21.9.2 Syntax
startDiscussionsCrawler(appName, sesUrl, sesPassword, host, port, [sesSchema, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format:</td>
</tr>
<tr>
<td></td>
<td><a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user (eqsys).</td>
</tr>
<tr>
<td>host</td>
<td>Host name of the machine where the application is running.</td>
</tr>
<tr>
<td>port</td>
<td>Port number used to access the application.</td>
</tr>
<tr>
<td>sesSchema</td>
<td>Optional. Username for the Oracle SES administrative user. Default value is eqsys.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.21.9.3 Example
The following example starts the discussion and announcement crawlers configured on the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal (webcenter) located at http://myhost.com:8888/webcenter/portal:

startDiscussionsCrawler(appName='webcenter',
sesUrl='http://myseshost.com:7777/search/api/admin/AdminService',
sesSchema='eqsys', sesPassword='password', host='myhost.com', port='8888')
10.21.10 stopSpacesCrawler

Module: Oracle WebCenter Portal

Use with WLST: Online

10.21.10.1 Description

Stops the Spaces crawler configured for WebCenter Portal, on an Oracle SES instance.

10.21.10.2 Syntax

stopSpacesCrawler(appName, sesUrl, sesPassword, host, port, [sesSchema, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format: <a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user (eqsys).</td>
</tr>
<tr>
<td>host</td>
<td>Host name of the machine where WebCenter Portal is running.</td>
</tr>
<tr>
<td>port</td>
<td>Port number used to access WebCenter Portal.</td>
</tr>
<tr>
<td>sesSchema</td>
<td>Optional. Username for the Oracle SES administrative user. Default value is eqsys.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where WebCenter Portal is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of WebCenter Portal is deployed.</td>
</tr>
</tbody>
</table>

10.21.10.3 Example

The following example stops the Spaces crawler configured on the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal located at http://myhost.com:8888/webcenter/portal:

stopSpacesCrawler(appName='webcenter', sesUrl='http://myseshost.com:7777/search/api/admin/AdminService', sesSchema='eqsys', sesPassword='password', host='myhost.com', port='8888')

10.21.11 stopDocumentsCrawler

Module: Oracle WebCenter Portal

Use with WLST: Online

10.21.11.1 Description

Stops the documents crawler configured for a named application, on an Oracle SES instance.
10.21.11.2 Syntax

stopDocumentsCrawler(appName, sesUrl, sesPassword, host, port, [sesSchema, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format:</td>
</tr>
<tr>
<td></td>
<td><code>http://host:port/search/api/admin/AdminService</code></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user (<code>eqsys</code>).</td>
</tr>
<tr>
<td>host</td>
<td>Host name of the machine where the application is running.</td>
</tr>
<tr>
<td>port</td>
<td>Port number used to access the application.</td>
</tr>
<tr>
<td>sesSchema</td>
<td>Optional. Username for the Oracle SES administrative user. Default value is <code>eqsys</code>.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, <code>WC_Spaces</code>.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.21.11.3 Example

The following example stops the document crawler configured on the Oracle SES instance `http://myseshost.com:7777` for WebCenter Portal located at `http://myhost.com:8888/webcenter/portal`:

```
stopDocumentsCrawler(appName='webcenter',
sesUrl='http://myseshost.com:7777/search/api/admin/AdminService',
sesSchema='eqsys', sesPassword='password', host='myhost.com', port='8888')
```

10.21.12 stopDiscussionsCrawler

Module: Oracle WebCenter Portal

Use with WLST: Online

10.21.12.1 Description

Stops the discussion and announcement crawlers configured for a named application, on an Oracle SES instance.

10.21.12.2 Syntax

stopDiscussionsCrawler(appName, sesUrl, sesPassword, host, port, [sesSchema, server, applicationVersion])
Example

The following example stops the discussion and announcement crawlers configured on the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal located at http://myhost.com:8888/webcenter/portal:

```
stopDiscussionsCrawler(appName='webcenter',
sesUrl='http://myseshost.com:7777/search/api/admin/AdminService',
sesSchema='eqsys', sesPassword='password', host='myhost.com', port='8888')
```

---

**deleteSpacesCrawler**

Module: Oracle WebCenter Portal

Use with WLST: Online

**10.21.13.1 Description**

Deletes the Spaces crawler configured for WebCenter Portal, on an Oracle SES instance.

**10.21.13.2 Syntax**

```
deleteSpacesCrawler(appName, sesUrl, sesPassword, host, port,[sesSchema, server, applicationVersion])
```

---

**Argument** | **Definition**
--- | ---
`appName` | Name of the application in which to perform this operation. For WebCenter Portal, the application name is always `webcenter`. For Portal Framework applications, specify the appropriate name.

```
<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td><code>sesUrl</code></td>
<td>Web service URL for the Oracle SES Administration API. Use the format: <code>http://host:port/search/api/admin/AdminService</code></td>
</tr>
<tr>
<td><code>sesPassword</code></td>
<td>Password for the Oracle SES administrative user (<code>eqsys</code>).</td>
</tr>
<tr>
<td><code>host</code></td>
<td>Host name of the machine where the application is running.</td>
</tr>
<tr>
<td><code>port</code></td>
<td>Port number used to access the application.</td>
</tr>
<tr>
<td><code>sesSchema</code></td>
<td>Optional. Username for the Oracle SES administrative user. Default value is <code>eqsys</code>.</td>
</tr>
<tr>
<td><code>server</code></td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><code>applicationVersion</code></td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
Example

The following example deletes the Spaces crawler configured on the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal named webcenter located at http://myhost.com:8888/webcenter/portal:

```
deleteSpacesCrawler(appName='webcenter',
sesUrl='http://myseshost.com:7777/search/api/admin/AdminService',
sesSchema='eqsys', sesPassword='password', host='myhost.com', port='8888')
```

10.21.14 deleteDocumentsCrawler

Module: Oracle WebCenter Portal

Use with WLST: Online

**10.21.14.1 Description**

Deletes the documents crawler configured for a named application, on an Oracle SES instance.

**10.21.14.2 Syntax**

```
deleteDocumentsCrawler(appName, sesUrl, sesPassword, host, port, [sesSchema, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format:</td>
</tr>
<tr>
<td></td>
<td><a href="http://host:port/search/api/admin/AdminService">http://host:port/search/api/admin/AdminService</a></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user (eqsys).</td>
</tr>
<tr>
<td>host</td>
<td>Host name of the machine where the application is running.</td>
</tr>
<tr>
<td>port</td>
<td>Port number used to access the application.</td>
</tr>
<tr>
<td>sesSchema</td>
<td>Optional. Username for the Oracle SES administrative user. Default value is eqsys.</td>
</tr>
</tbody>
</table>
Example
The following example deletes the document crawler configured on the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal located at http://myhost.com:8888/webcenter/portal:

```
deleteDocumentsCrawler(appName='webcenter',
    sesUrl='http://myseshost.com:7777/search/api/admin/AdminService',
    sesSchema='eqsys', sesPassword='password', host='myhost.com', port='8888')
```

10.21.15 deleteDiscussionsCrawler
Module: Oracle WebCenter Portal
Use with WLST: Online

10.21.15.1 Description
Deletes the discussion and announcement crawlers configured for a named application, on an Oracle SES instance.

10.21.15.2 Syntax
```
deleteDiscussionsCrawler(appName, sesUrl, sesPassword, host, port, [server], [applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>sesUrl</td>
<td>Web service URL for the Oracle SES Administration API. Use the format:</td>
</tr>
<tr>
<td></td>
<td><code>http://host:port/search/api/admin/AdminService</code></td>
</tr>
<tr>
<td>sesPassword</td>
<td>Password for the Oracle SES administrative user (eqsys).</td>
</tr>
<tr>
<td>host</td>
<td>Host name of the machine where the application is running.</td>
</tr>
<tr>
<td>port</td>
<td>Port number used to access the application.</td>
</tr>
<tr>
<td>sesSchema</td>
<td>Optional. Username for the Oracle SES administrative user. Default value is eqsys.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.21.15.3 Example
The following example deletes the discussion and announcement crawlers configured on the Oracle SES instance http://myseshost.com:7777 for WebCenter Portal located at http://myhost.com:8888/webcenter/portal:

```
deleteDiscussionsCrawler(appName='webcenter',
    sesUrl='http://myseshost.com:7777/search/api/admin/AdminService',
    sesSchema='eqsys', sesPassword='password', host='myhost.com', port='8888')
```

10.22 Search - WebCenter Portal Search
Use the commands listed in Table 10–31 to manage search settings and crawl options for WebCenter Portal and Portal Framework applications.

Configuration changes made using these WLST commands are effective immediately; no restart is required.

### Table 10–31  WebCenter Portal Search WLST Commands

<table>
<thead>
<tr>
<th>Use This Command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>setSearchConfig</td>
<td>Modify search settings for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>listSearchConfig</td>
<td>List search properties for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>setSpacesCrawlProperties</td>
<td>Specify crawl properties for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>getSpacesCrawlProperties</td>
<td>Return the current crawl settings for a named application.</td>
<td>Online</td>
</tr>
</tbody>
</table>

10.22.1 setSearchConfig
Module: Oracle WebCenter Portal
Use with WLST: Online

10.22.1.1 Description
Modifies search settings for a named application. If a parameter is not specified, it is not modified.

10.22.1.2 Syntax
```
setSearchConfig(appName, [numSavedSearches, numResultsRegion, numResultsMain, executionTimeout, prepareTimeout, showAllExecutionTimeout, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>numSavedSearches</td>
<td>Optional. The number of saved searches to display in the Saved Searches drop down (on main search page).</td>
</tr>
<tr>
<td>numResultsRegion</td>
<td>Optional. The number of saved searches displayed in a Saved Search task flow.</td>
</tr>
</tbody>
</table>
10.22.1.3 Examples

The following example specifies that saved searches display five search results per tool or service. Additionally, that a seven second search execution timeout is required:

```
wl$:/weblogic/serverConfig> setSearchConfig(appName='webcenter',
numResultsRegion=5, executionTimeout=7000);
```

The following example increases the number of saved searches in the Saved Searches drop down list to eight:

```
wl$:/weblogic/serverConfig> setSearchConfig(appName='webcenter',
numSavedSearches=8);
```

The following example sets the search execution timeout to five seconds and allows each tool or service fifteen seconds to display search results before timing out:

```
wl$:/weblogic/serverConfig> setSearchConfig(appName='webcenter',
executionTimeout=5000, showAllExecutionTimeout=15000);
```

10.22.2 listSearchConfig

Module: Oracle WebCenter Portal

Use with WLST: Online

10.22.2.1 Description

Lists search settings for a named application.

10.22.2.2 Syntax

```
listSearchConfig(appName, [server, applicationVersion])
```
The following example displays search configuration information for WebCenter Portal:

```
wlst:/weblogic/serverConfig>listSearchConfig(appName='webcenter')
```

### 10.22.3 setSpacesCrawlProperties

**Module:** Oracle WebCenter Portal

**Use with WLST:** Online

### 10.22.3.1 Description

Specifies Oracle SES crawl properties for WebCenter Portal or Portal Framework applications.

WebCenter Portal and Portal Framework applications can be crawled by Oracle SES to provide a faster, more unified search experience across WebCenter Portal objects, specifically: portals, lists, pages, people (profiles), wikis, blogs, documents, discussions, and announcements. Three distinct crawlers make this possible:

- Spaces Crawler (for portals, lists, pages, and people)
- Documents Crawler (for documents, wikis, blogs)
- Discussions Crawler (for discussions and announcements).

Use this command to enable or disable Oracle SES crawlers in the following applications:

- **WebCenter Portal**—To use Oracle SES crawlers in WebCenter Portal, you must enable all three crawlers.

- **Portal Framework applications**—To use Oracle SES crawlers in Portal Framework applications, you must enable both the documents and discussions crawlers. The Spaces crawler is not applicable.

(WebCenter Portal only) You can also use this command to specify an interval between full crawls for the Spaces crawler. During a full crawl, all of the Spaces crawler content is re-read. Out-of-the-box, full crawls for the Spaces crawler occur every seven days but you can specify a different frequency to suit your installation.

Note that incremental crawls, for all three crawlers, are initiated by a scheduler running from Oracle SES. During these incremental crawls, only content added or updated since the previous crawl is processed.
10.22.3.2 Syntax

```
setSpacesCrawlProperties(appName, [fullCrawlIntervalInHours, spacesCrawlEnabled, documentCrawlEnabled, discussionsCrawlEnabled, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always <code>webcenter</code>. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>fullCrawlIntervalInHours</td>
<td>Optional. (WebCenter Portal only) Number of hours between full crawls. The default is 168 hours or 7 days.</td>
</tr>
<tr>
<td>spacesCrawlEnabled</td>
<td>Optional. Specifies whether the Spaces crawler is enabled in Oracle SES. Valid values are 1 (true) and 0 (false). This argument defaults to 0. When set to 0, Oracle WebCenter Portal’s internal search adapters return search results.</td>
</tr>
<tr>
<td>documentCrawlEnabled</td>
<td>Optional. Specifies whether the Documents crawler is enabled in Oracle SES. Valid values are 1 (true) and 0 (false). This argument defaults to 0. When set to 0, Oracle WebCenter Portal’s internal search adapters return search results.</td>
</tr>
<tr>
<td>discussionsCrawlEnabled</td>
<td>Optional. Specifies whether the Discussions and Announcement crawlers are enabled in Oracle SES. Valid values are 1 (true) and 0 (false). This argument defaults to 0. When set to 0, Oracle WebCenter Portal’s internal search adapters return search results.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, <code>WC_Spaces</code>. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.22.3.3 Example

The following example enables Oracle SES crawlers in WebCenter Portal and specifies that WebCenter Portal runs a full crawl through the Spaces crawler every 8 days:

```
wlsc:/weblogic/serverConfig> setSpacesCrawlProperties(appName='webcenter', fullCrawlIntervalInHours=192, spacesCrawlEnabled=1, documentCrawlEnabled=1, discussionsCrawlEnabled=1)
```

10.22.4 getSpacesCrawlProperties

Module: Oracle WebCenter Portal

Use with WLST: Online

10.22.4.1 Description

Returns the current crawl settings for a named application, such as the number of hours between full crawls (Spaces crawler), and whether Oracle SES crawlers are enabled.
10.22.4.2 Syntax

getSpacesCrawlProperties(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different</td>
</tr>
<tr>
<td></td>
<td>servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more</td>
</tr>
<tr>
<td></td>
<td>than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.22.4.3 Example

The following example returns the current crawl settings for WebCenter Portal (webcenter):

wls:/weblogic/serverConfig> getSpacesCrawlProperties(appName='webcenter')

Spaces Crawl Properties:
-----------------------
fullCrawlIntervalInHours: 124
spacesCrawlEnabled:       1
documentCrawlEnabled:     1
discussionsCrawlEnabled:  1

10.23 Worklists

Use the commands listed in Table 10–32 to manage BPEL server connections for applications.

Configuration changes made using these WLST commands are only effective after your restart the Managed Server on which the application is deployed. For details, see Oracle Fusion Middleware Administering Oracle WebCenter Portal.

**Table 10–32 Worklist Commands**

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>createBPELConnection</td>
<td>Create a connection to a BPEL server for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>setBPELConnection</td>
<td>Edit an existing BPEL server connection.</td>
<td>Online</td>
</tr>
<tr>
<td>listBPELConnections</td>
<td>List all of the BPEL server connections that are configured for a named application.</td>
<td>Online</td>
</tr>
<tr>
<td>addWorklistConnection</td>
<td>Enable an existing BPEL server connection for worklists.</td>
<td>Online</td>
</tr>
<tr>
<td>removeWorklistConnection</td>
<td>Disable a BPEL server connection currently used by worklists.</td>
<td>Online</td>
</tr>
<tr>
<td>listWorklistConnections</td>
<td>List individual or all BPEL server connections configured for worklists.</td>
<td>Online</td>
</tr>
</tbody>
</table>
10.23.1 createBPELConnection

Module: Oracle WebCenter Portal
Use with WLST: Online

10.23.1.1 Description

Creates a connection to a BPEL server for a named application. BPEL server connections can be used by worklists and also by WebCenter Portal’s internal portal workflows.

To configure worklists to actively use a new BPEL server connection, use the addWorklistConnection command. See Section 10.23.4, "addWorklistConnection".

To specify the BPEL server connection that WebCenter Portal uses for its internal workflows, use the setSpacesWorkflowConnectionName command. See Section 10.24.2, "setSpacesWorkflowConnectionName".

10.23.1.2 Syntax

createBPELConnection(appName, name, url, [policy, recipientKeyAlias, linkUrl, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td></td>
<td>For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Connection name. The name must be unique (across all connection types) within the application.</td>
</tr>
<tr>
<td>url</td>
<td>URL required to access the BPEL server.</td>
</tr>
<tr>
<td></td>
<td>Use the format: protocol://host:port</td>
</tr>
<tr>
<td></td>
<td>The BPEL server URL must be unique within the application.</td>
</tr>
<tr>
<td>policy</td>
<td>Optional. SAML token policy this connection uses for authentication.</td>
</tr>
<tr>
<td></td>
<td>Enter any valid policy. Valid values include:</td>
</tr>
<tr>
<td></td>
<td>■ oracle/wss10_saml_token_client_policy—use to access the BPEL server with the default, non message protected policy.</td>
</tr>
<tr>
<td></td>
<td>■ oracle/wss10_saml_token_with_message_protection_client_policy—use to access the BPEL server with a message protected policy. If selected, you must configure keys stores both in your application and in the BPEL application.</td>
</tr>
<tr>
<td></td>
<td>■ GPA—use if your environment supports Global Policy Attachments (GPA).</td>
</tr>
<tr>
<td></td>
<td>If you omit this argument, the connection defaults to oracle/wss10_saml_token_client_policy.</td>
</tr>
<tr>
<td>recipientKeyAlias</td>
<td>Optional. Recipient key alias to be used for message protected SAML policy authentication. Only required when the BPEL server connection is using a SAML token policy for authentication and the application’s worklist is using multiple BPEL server connections. The default is null.</td>
</tr>
</tbody>
</table>
|                   | See also “Configuring WS-Security” in Oracle Fusion Middleware Administering Oracle WebCenter Portal.
10.23.1.3 Examples
The following example creates a connection named WebCenter Worklist with the default security policy:

```
ws:/weblogic/serverConfig> createBPELConnection(appName='webcenter',
name='WebCenter Worklist', url='http://myhost.com:8001',
policy='oracle/wss10_saml_token_client_policy')
```

The following example creates a connection that uses a message protected security policy, and defines a specific link URL:

```
ws:/weblogic/serverConfig> createBPELConnection(appName='webcenter',
name='WebCenter Worklist', url='http://myhost.com:8001', policy='oracle/wss10_saml_token_with_message_protection_client_policy',
recipientKeyAlias='myalias', linkUrl='http://mySSO.com:7777')
```

The following example creates a connection to be used in an environment that supports Global Policy Attachments (GPA):

```
ws:/weblogic/serverConfig> createBPELConnection(appName='webcenter',
name='WebCenter Worklist', url='http://myhost.com:8001', policy='GPA')
```

10.23.2 setBPELConnection
Module: Oracle WebCenter Portal
Use with WLST: Online

10.23.2.1 Description
Edits an existing BPEL server connection.

To configure the worklists to actively use an existing BPEL server connection, use the addWorklistConnection command. See Section 10.23.4, "addWorklistConnection".

To specify the BPEL server connection used for WebCenter Portal's internal workflows, use the setSpacesWorkflowConnectionName command. See Section 10.24.2, "setSpacesWorkflowConnectionName".

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>linkUrl</td>
<td>Optional. URL used to link to the BPEL server. Only required if it is different to the url argument. For example, when SSO or HTTPS is configured. Use the format: protocol://host:port. The default is null. For performance reasons, in an HTTPS or SSO environment, linkUrl specifies user access to BPEL worklist items, through HTTPS or SSO web servers, whereas url specifies direct access to BPEL web services, without redirection through HTTPS or SSO Web servers.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
### 10.23.2.2 Syntax

```
setBPELConnection(appName, name, [url, policy, recipientKeyAlias, linkUrl, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>name</td>
<td>Existing BPEL server connection name.</td>
</tr>
<tr>
<td>url</td>
<td>Optional. URL required to access the BPEL server. Use the format: <code>&lt;protocol&gt;://&lt;host&gt;:&lt;port&gt;</code> The BPEL server URL must be unique within the application.</td>
</tr>
</tbody>
</table>
| policy            | Optional. SAML token policy this connection uses for authentication. Enter any valid policy. Valid values include:  
|                   | - oracle/wss10_saml_token_client_policy—use to access the BPEL server with the default, non message protected policy.  
|                   | - oracle/wss10_saml_token_with_message_protection_client_policy—use to access the BPEL server with a message protected policy. If selected, you must configure keys stores both in your application and in the BPEL application.  
|                   | - GPA—use if your environment supports Global Policy Attachments (GPA).  
|                   | If you omit this argument, the connection defaults to oracle/wss10_saml_token_client_policy.                                                  |
| recipientKeyAlias | Optional. Recipient key alias to be used for message protected SAML policy authentication. Only required when the BPEL server connection is using a SAML token policy for authentication and the application's worklist is using multiple BPEL server connections. The default is null.  
| linkUrl           | Optional. URL used to link to the BPEL server. Only required if it is different to the url argument. For example, when SSO or https is configured. Use the format: `protocol://host:port` For example, http://mySSO.host.com:7777  
| server            | Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.  
| applicationVersion| Optional. Version number of the deployed application. Required if more than one version of the application is deployed.                        |
10.23.2.3 Examples

The following example updates the BPEL server URL, security policy, recipient key alias, and link url for a connection named WebCenter Worklist.

```bash
wls:/weblogic/serverConfig> setBPELConnection(appName='webcenter', name='WebCenter Worklist', url='http://myhost.com:6666', policy='oracle/wss10_saml_token_with_message_protection_client_policy', recipientKeyAlias='myalias', linkUrl='http://mySSO.com:7777')
```

The following example changes the security policy to use Global Policy Attachments (GPA):

```bash
wls:/weblogic/serverConfig> setBPELConnection(appName='webcenter', name='WebCenter Worklist', policy='GPA')
```

10.23.3 listBPELConnections

Module: Oracle WebCenter Portal

Use with WLST: Online

10.23.3.1 Description

Without any arguments, this command lists all the BPEL connections that are configured for a named application. All BPEL connections are listed, even connections not currently used.

10.23.3.2 Syntax

```
listBPELConnections(appName, [verbose, name, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Displays BPEL server connection details in verbose mode. Valid options are 1 (true) and 0 (false). When set to 1, listBPELConnections lists all of the BPEL server connections that are configured, along with their details. When set to 0, listBPELConnections lists connection names only. This argument defaults to 0. If you set this argument to 0, do not specify the name argument.</td>
</tr>
<tr>
<td>name</td>
<td>Optional. Name of an existing BPEL server connection. You can use this argument to view details about a specific connection. To list all the connections, omit the name argument.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>
10.23.3 Examples

The following example lists the names of all the BPEL server connections that are configured for WebCenter Portal:

```
`wls:/weblogic/serverConfig> listBPELConnections(appName='webcenter')
------------------
WebCenter Worklist
------------------
Human Resources Worklist
------------------
```

The following example lists the names and details of all of the BPEL server connections that are configured for WebCenter Portal:

```
`wls:/weblogic/serverConfig> listBPELConnections(appName='webcenter', verbose=1)
------------------
Connection Name: WebCenter Worklist
PolicyURI:oracle/wss10_saml_token_client_policy
URL:http://myhost.com:8001
------------------
Connection Name: Human Resources Worklist
PolicyURI:oracle/wss10_saml_token_client_policy
URL:http://myhost.com:8888
------------------
```

10.23.4 addWorklistConnection

Module: Oracle WebCenter Portal

Use with WLST: Online

10.23.4.1 Description

Enables an existing BPEL server connection for worklists. Worklists supports multiple connections so that users can monitor and manage assignments and notifications from a range of BPEL servers.

The name must specify an existing BPEL server connection.

10.23.4.2 Syntax

```
addWorklistConnection(appName, name,
```
10.23.3 Examples
The following example enables the Human Resources Worklist connection for the worklists in WebCenter Portal:

```
    wls:/weblogic/serverConfig> addWorklistConnection(appName='webcenter', name='Human Resources Worklist', verbose=1)
    Human Resources Worklist successfully added to WorkList
    ----------------------------------------
    Human Resources Worklist
    ------------------------
    Connection Name: Human Resources Worklist
    PolicyURI:oracle/wss10_saml_token_client_policy
    URL:http://myhost.com:8888
```

The following example also enables the Human Resources Worklist connection for worklists in WebCenter Portal:

```
    wls:/weblogic/serverConfig> addWorklistConnection(appName='webcenter', name='Human Resources Worklist', verbose=1)
    Human Resources Worklist successfully added to WorkList
    ----------------------------------------
    Human Resources Worklist
    ------------------------
    Connection Name: Human Resources Worklist
    PolicyURI:oracle/oracle/wss10_saml_token_client_policy
    URL:http://myhost.com:8888
```

10.23.5 removeWorklistConnection
Module: Oracle WebCenter Portal
Use with WLST: Online

10.23.5.1 Description
Disables a BPEL server connection that is currently used by worklists. Connection details are retained but worklists no longer use the connection specified.

10.23.5.2 Syntax
```
    removeWorklistConnection(appName, name, [server, applicationVersion])
```
Worklists

10.23.5.3 Example
The following example disables the BPEL server connection named WebCenter Worklist for worklists in WebCenter Portal:

```
$ wls:/weblogic/serverConfig> removeWorklistConnection(appName='webcenter',
name='WebCenter Worklist')
WebCenter Worklist successfully removed from WorkList
```

10.23.6 listWorklistConnections
Module: Oracle WebCenter Portal
Use with WLST: Online

10.23.6.1 Description
Without any arguments, this command lists all of the BPEL server connections that are configured for the worklists, in a named application.

10.23.6.2 Syntax
```
listWorklistConnections(appName, [verbose, name, server, applicationVersion])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal, the application name is always webcenter. For Portal Framework applications, specify the appropriate name.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. Displays BPEL server connection details in verbose mode. Valid options are 1 (true) and 0 (false). When set to 1, listWorklistConnections lists all of the BPEL server connections that are configured for worklists, along with their details. When set to 0, listWorklistConnections lists connection names only. This argument defaults to 0. If you set this argument to 0, do not specify the name argument.</td>
</tr>
<tr>
<td>name</td>
<td>Optional. Name of an existing BPEL server connection. You can use this argument to view details about a specific connection. To list all connections, omit the name argument.</td>
</tr>
</tbody>
</table>

```
10.23.6.3 Examples

The following example lists the names of all of the BPEL server connections that are configured for worklists in WebCenter Portal:

```
> listWorklistConnections(appName='webcenter')
```

```
------------------
| WebCenter Worklist |
------------------
```

The following example lists both the names and connection details of all of the BPEL server connections that are configured for worklists in WebCenter Portal:

```
> listWorklistConnections(appName='webcenter', verbose=1)
```

```
------------------
| WebCenter Worklist |
------------------
Connection Name: WebCenter Worklist
PolicyURI:oracle/wss10_saml_token_client_policy
URL:http://myhost.com:8001
```

The following example lists connection details of a named BPEL server connection—MyWorklist. As worklists is not currently configured to use MyWorklist, an appropriate message displays.

```
> listWorklistConnections(appName='webcenter', verbose=1, name='MyWorklist')
```

```
------------------
The following connection is not in the ADF Worklist:MyWorklist
```

10.24 WebCenter Portal Application

Use the commands listed in Table 10–33 to manage workflow settings and metadata for the "out-of-the-box" WebCenter Portal application.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 10–33 WebCenter Portal Application WLST Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use This Command...</td>
</tr>
<tr>
<td>getSpacesWorkflowConnectionName</td>
</tr>
<tr>
<td>setSpacesWorkflowConnectionName</td>
</tr>
<tr>
<td>refreshGroupSpaceCache</td>
</tr>
<tr>
<td>refreshSpaceTemplateCache</td>
</tr>
</tbody>
</table>
10.24.1 `getSpacesWorkflowConnectionName`

Module: Oracle WebCenter Portal

Use with WLST: Online

10.24.1.1 Description

Returns the name of the BPEL server connection that WebCenter Portal is currently using for internal workflows (portal membership notifications, portal subscription requests, and so on).

10.24.1.2 Syntax

getSpacesWorkflowConnectionName(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always <code>webcenter</code>.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where WebCenter Portal is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, <code>WC_Spaces</code>.</td>
</tr>
<tr>
<td></td>
<td>Required when applications with the same name are deployed to different</td>
</tr>
<tr>
<td></td>
<td>servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than</td>
</tr>
<tr>
<td></td>
<td>one version of WebCenter Portal is deployed.</td>
</tr>
</tbody>
</table>

10.24.1.3 Example

The following example names the BPEL server connection that WebCenter Portal is currently using for internal workflows:

```
wls:/weblogic/serverConfig> getSpacesWorkflowConnectionName(appName='webcenter')
WorkflowConfigConnectionName: WebCenter-Worklist
```

10.24.2 `setSpacesWorkflowConnectionName`

Module: Oracle WebCenter Portal

Use with WLST: Online

10.24.2.1 Description

Specifies the BPEL server connection that WebCenter Portal uses for internal workflows. WebCenter Portal uses a BPEL server included with the Oracle SOA Suite to host internal workflows, such as portal membership notifications, portal subscription requests, and so on. The connection name specified here must be a valid BPEL server connection.

| Note: | Configuration changes made using this WLST command are only effective after your restart the Managed Server on which WebCenter Portal is deployed. For details, see Oracle Fusion Middleware Administering Oracle WebCenter Portal. |

10.24.2.2 Syntax

setSpacesWorkflowConnectionName(appName, name, [server, applicationVersion])
10.24.2.3 Example
The following example specifies that WebCenter Portal uses the BPEL server connection named WebCenter-Worklist for its internal workflows.

```wls:/weblogic/serverConfig>setSpacesWorkflowConnectionName(appName='webcenter', name='WebCenter-Worklist')```

10.24.3 refreshGroupSpaceCache
Module: Oracle WebCenter Portal
Use with WLST: Online

10.24.3.1 Description
(Used when patching WebCenter Spaces 11.1.1.1.0 to 11.1.1.2.0).
Migrates metadata for individual spaces (in MDS) and space security data (in a policy store) to the Spaces Cache.
WebCenter Spaces 11.1.1.2.0 and later uses tables (referred to as the Spaces Cache) to store space metadata and security-related data. When you patch from WebCenter Spaces 11.1.1.1.0 to 11.1.1.2.0, you must manually run the refreshGroupSpaceCache command to migrate your existing space data to the new ‘Spaces Cache’.

10.24.3.2 Syntax
refreshGroupSpaceCache(appName, [spaceNames, syncMode, updateType, cleanCache, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal (previously known as Spaces), the application name is always webcenter.</td>
</tr>
<tr>
<td>spaceNames</td>
<td>Optional. Names of one or more spaces (group spaces) that you want to refresh. To refresh all the spaces in MDS, enter spaceNames=''. To refresh selective spaces, enter one or more space names separated with a comma, for example: spaceNames='MyGroupSpace1, MyGroupSpace'</td>
</tr>
</tbody>
</table>
The following examples update the cache to include all space-related metadata (in MDS) and security data (in the policy store) in synchronous mode:

```
wls:/weblogic/serverConfig>refreshGroupSpaceCache(appName='webcenter', spaceNames='', syncMode=1, updateType='all', cleanCache=0)
```

### Argument Definitions

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>updateType</code></td>
<td>Optional. Indicates the type of data to refresh. Valid values are: security, metadata, all. The default value is security.</td>
</tr>
<tr>
<td></td>
<td>- <code>security</code> - Refreshes the cache with security data stored in the policy store. The security data that is stored includes member data for the space, whether or not a space is public, and whether or not a space is accessible to users assigned the Authenticated-User role (in earlier releases this role was named Spaces-User).</td>
</tr>
<tr>
<td></td>
<td>- <code>metadata</code> - Refreshes the cache with space-related metadata stored in MDS. The data that is stored includes metadata information such as the display name, keywords, icon, logo, and so on.</td>
</tr>
<tr>
<td></td>
<td>- <code>all</code> - Refreshes the cache with data stored in MDS and the policy store.</td>
</tr>
<tr>
<td><code>syncMode</code></td>
<td>Optional. Indicates whether to refresh the cache in synchronous or asynchronous mode. Valid values are 1 and 0. The default value is 1.</td>
</tr>
<tr>
<td></td>
<td>When set to 1, the refresh process runs in synchronous mode. When set to 0, the refresh is asynchronous, that is, a new thread is spawned for the refresh process. Synchronous mode is recommended.</td>
</tr>
<tr>
<td><code>cleanCache</code></td>
<td>Optional. Indicates whether to remove data that no longer exists in MDS from the WebCenter Portal Cache during the refresh operation. Valid values are 1 and 0. The default value is 0.</td>
</tr>
<tr>
<td></td>
<td>0 - No content in the WebCenter Portal Cache is cleared during the refresh operation.</td>
</tr>
<tr>
<td></td>
<td>1 - Clears space data from the WebCenter Portal Cache if the space does not exist in MDS:</td>
</tr>
<tr>
<td></td>
<td>- If <code>spaceNames=''</code>, data in the Cache is refreshed for all the spaces in MDS and data for any other space (not in MDS) is removed from the Cache.</td>
</tr>
<tr>
<td></td>
<td>- If <code>spaceNames</code> specifies the names of one or more spaces, only data for the named spaces is refreshed in the Cache. If you name a space that does not exist in MDS, data for that space is removed from the Cache.</td>
</tr>
<tr>
<td></td>
<td>Only set <code>cleanCache</code> to 1 if you are refreshing all the spaces. Do not set <code>cleanCache</code> to 1 if you are refreshing one or two individual spaces as this setting will unnecessarily remove all the spaces that are not refreshed.</td>
</tr>
<tr>
<td><code>server</code></td>
<td>Optional. Name of the managed server where WebCenter Portal is deployed. For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td><code>applicationVersion</code></td>
<td>Optional. Version number of the deployed application. Required if more than one version of WebCenter Portal is deployed.</td>
</tr>
</tbody>
</table>
wls:/weblogic/serverConfig>refreshGroupSpaceCache(appName='webcenter')

The following example updates the Spaces Cache to include space-related metadata (in MDS) and security data (in the policy store) for two group spaces named MyGroupSpace1 and MyGroupSpace2. The cache refreshes in synchronous mode.

wls:/weblogic/serverConfig>refreshGroupSpaceCache(appName='webcenter', spaceNames='MyGroupSpace1,MyGroupSpace2')

10.24.4 refreshSpaceTemplateCache

Module: Oracle WebCenter Portal
Use with WLST: Online

10.24.4.1 Description
Migrates metadata for individual space templates (in MDS) and template security data (in a policy store) to the Space Template Cache.

WebCenter Spaces 11.1.1.2.0 (and later) uses tables (referred to as the Space Template Cache) to store space template metadata and security-related data. When you patch from WebCenter Spaces 11.1.1.1.0 to 11.1.1.2.0, you must run the refreshSpaceTemplateCache command so that all your existing template data is migrated to the new Space Templates Cache.

10.24.4.2 Syntax
refreshSpaceTemplateCache(appName, [spaceTemplateNames, syncMode, updateType, cleanCache, server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation. For WebCenter Portal (previously known as Spaces), the application name is always webcenter.</td>
</tr>
<tr>
<td>spaceNames</td>
<td>Optional. Names of one or more space templates that you want to refresh. To refresh all the space templates in MDS, enter spaceTemplateName=''. To refresh selective space templates, enter one or more template names separated with a comma, for example: spaceNames='MySpaceTemplate1,MySpaceTemplate2'</td>
</tr>
</tbody>
</table>
| updateType    | Optional. Indicates the type of data to refresh. Valid values are: security, metadata, all. The default value is security.  
- security - Refreshes the cache with security data stored in the policy store. The security data that is stored includes member data for the space template, whether or not a space template is public, and whether or not a space template is accessible to users assigned the Authenticated-User role (in earlier releases this role was named Spaces-User).  
- metadata - Refreshes the cache with space template-related metadata stored in MDS. The data that is stored includes metadata information such as the display name, keywords, icon, logo, and so on.  
- all - Refreshes the space template cache with data stored in MDS and the policy store. |
### Example

The following examples update the cache to include all space template-related metadata (in MDS) and security data (in the policy store) in synchronous mode:

```bash
wls:/weblogic/serverConfig>refreshSpaceTemplateCache(appName='webcenter', spaceTemplateName='', syncMode=1, updateType='all', cleanCache=0)
```

```bash
wls:/weblogic/serverConfig>refreshSpaceTemplateCache(appName='webcenter')
```

The following example updates the Space Templates Cache to include space template related metadata (in MDS) and security data (in the policy store) for two space templates named `MySpaceTemplate1` and `MySpaceTemplate2`. The cache refreshes in synchronous mode.

```bash
wls:/weblogic/serverConfig>refreshSpaceTemplateCache(appName='webcenter', spaceNames='MySpaceTemplate1,MySpaceTemplate2')
```
10.25 Upgrade

Use the commands listed in Table 10–34 when upgrading from a previous Oracle WebCenter Portal release.

See also, Oracle Fusion Middleware Upgrade Guide for Oracle SOA Suite, WebCenter Portal, and ADF.

Table 10–34 Oracle WebCenter Portal Upgrade WLST Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>upgradeWebCenterDomain</td>
<td>Upgrade a Oracle WebCenter Portal domain.</td>
<td>Offline</td>
</tr>
<tr>
<td>upgradeWebCenterPermissions</td>
<td>Upgrade WebCenter Portal permissions.</td>
<td>Online</td>
</tr>
<tr>
<td>upgradeWebCenterApplication</td>
<td>Upgrade WebCenter Portal or a Portal Framework application.</td>
<td>Online</td>
</tr>
</tbody>
</table>

10.25.1 upgradeWebCenterDomain

Module: Oracle WebCenter Portal

Use with WLST: Offline

10.25.1.1 Description

Upgrades a Oracle WebCenter Portal domain from 11.1.1.2.0 or 11.1.1.3.0 to 11.1.1.4.0

10.25.1.2 Syntax

upgradeWebCenterDomain(domainDirName, [oracleHome, upgradeCustomSpaces])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>domainDirName</td>
<td>Full path to the domain’s home directory.</td>
</tr>
<tr>
<td></td>
<td>For example, /home/Oracle/Domains/wc_domain.</td>
</tr>
<tr>
<td>oracleHome</td>
<td>Optional. Path to Oracle WebCenter Portal’s Oracle home directory.</td>
</tr>
<tr>
<td></td>
<td>For example, /home/Oracle/Middleware/Oracle_WC.</td>
</tr>
<tr>
<td>upgradeCustomSpaces</td>
<td>Optional. Determines whether to upgrade the custom.webcenter.spaces shared library. Valid values are 1 (true) and 0 (false).</td>
</tr>
<tr>
<td></td>
<td>Set to 1 if you customized the Spaces application (previous name for WebCenter Portal) and you want your customizations to be included when you upgrade.</td>
</tr>
<tr>
<td></td>
<td>The default value is 0.</td>
</tr>
</tbody>
</table>

10.25.1.3 Example

The following example upgrades an Oracle WebCenter Portal domain named base_domain:

wls:/weblogic/serverConfig> upgradeWebCenterDomain(domainDirName="/mr_home/user_project/domains/base_domain");
10.25.2 upgradeWebCenterPermissions

Module: Oracle WebCenter Portal
Use with WLST: Online

10.25.2.1 Description
Upgrades permissions for WebCenter Portal (previously known as WebCenter Spaces).
This command creates additional application roles and grants some additional
permissions that are requirement for WebCenter Portal 11.1.1.4.0 and later.

10.25.2.2 Syntax
upgradeWebCenterPermissions()

10.25.2.3 Example
The following example upgrades permissions for WebCenter Portal:

wls:/weblogic/serverConfig> upgradeWebCenterPermissions();

10.25.3 upgradeWebCenterApplication

Module: Oracle WebCenter Portal
Use with WLST: Online

10.25.3.1 Description
Upgrades WebCenter Portal (previously known as WebCenter Spaces) from 11.1.1.2.0 or
11.1.1.3.0 to 11.1.1.4.0.

10.25.3.2 Syntax
upgradeWebCenterApplication(appName, [server, applicationVersion])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the application in which to perform this operation.</td>
</tr>
<tr>
<td></td>
<td>For WebCenter Portal, the application name is always webcenter.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. Name of the managed server where the application is deployed.</td>
</tr>
<tr>
<td></td>
<td>For example, WC_Spaces. Required when applications with the same name are deployed to different servers and also when you have a cluster.</td>
</tr>
<tr>
<td>applicationVersion</td>
<td>Optional. Version number of the deployed application. Required if more than one version of the application is deployed.</td>
</tr>
</tbody>
</table>

10.25.3.3 Example
The following example upgrades WebCenter Portal (webcenter):

wls:/weblogic/serverConfig> upgradeWebCenterApplication(appName='webcenter');
This chapter provides detailed descriptions of custom WLST commands for ADF, including command syntax, arguments and command examples.

The following sections describe the WLST custom commands and variables in detail. Topics include:

- Section 11.1, "Overview of WLST Command Categories"
- Section 11.2, "ADF-Specific WLST Commands"

### Note:
To use these ADF custom WLST commands, you must invoke the WLST script from the Oracle Common home. See "Using Custom WLST Commands" in the Oracle Fusion Middleware Administrator’s Guide.

#### 11.1 Overview of WLST Command Categories

Use the ADF-based URL Connections WLST commands to navigate the hierarchy of configuration or runtime beans and control the prompt display. Use the getADFMArchiveConfig command to manage the ADFMArchiveConfig object.

#### 11.2 ADF-Specific WLST Commands

Use the commands in Table 11–1 to managing URL-based connections.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>adf_createFileUrlConnection</td>
<td>Create a new ADF File connection.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>adf_createHttpUrlConnection</td>
<td>Create a new ADF URL connection.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>adf_setURLConnectionAttributes</td>
<td>Set or edit the attributes of a newly created or existing ADF connection.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>adf_listUrlConnection</td>
<td>List a new URL connection.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>getADFMArchiveConfig</td>
<td>Return a handle to the ADFMArchiveConfig object for the specified archive.</td>
<td>Online or Offline</td>
</tr>
</tbody>
</table>
11.2.1 adf_createFileUrlConnection

Use with WLST: Online or Offline

11.2.1.1 Description

Use this command to create a new connection based on the oracle.adf.model.connection.url.FileURLConnection connection class.

11.2.1.2 Syntax

adf_createFileURLConnection(appName, name, URL)

11.2.1.3 Example

adf_createFileURLConnection('myapp','tempDir','/scratch/tmp')

11.2.2 adf_createHttpUrlConnection

Use with WLST: Online or Offline
11.2.2.1 Description
Use this command to create a new connection based on the oracle.adf.model.connection.url.HttpURLConnection connection type class.

11.2.2.2 Syntax
adf.createHttpURLConnection (appName, name, [URL], [authenticationType], [realm], [user], [password]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Application name for which the connection is to be created.</td>
</tr>
<tr>
<td>name</td>
<td>The name of the new connection.</td>
</tr>
<tr>
<td>url</td>
<td>(Optional) The URL associated with this connection.</td>
</tr>
<tr>
<td>authenticationType</td>
<td>(Optional) The default is basic.</td>
</tr>
<tr>
<td>realm</td>
<td>(Optional) If this connection deals with authentication, then this should be set. The default is basic.</td>
</tr>
<tr>
<td>user</td>
<td>(Optional)</td>
</tr>
<tr>
<td>password</td>
<td>(Optional)</td>
</tr>
</tbody>
</table>

11.2.2.3 Example
adf_createHttpURLConnection('myapp','cnn','http://www.cnn.com')

11.2.3 adf_setURLConnectionAttributes
Use with WLST: Online or Offline

11.2.3.1 Description
Use this command to set or edit the attributes of a newly created or existing ADF connection.

11.2.3.2 Syntax
adf_setURLConnectionAttributes(appname, connectionname, attributes)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appname</td>
<td>Application name for which the connection that will be created.</td>
</tr>
<tr>
<td>connectionname</td>
<td>The name of the new connection.</td>
</tr>
<tr>
<td>attributes</td>
<td>The array containing attributes to set in key/value pairs.</td>
</tr>
</tbody>
</table>

11.2.3.3 Example
adf_setURLConnectionAttributes('myapp','cnn','ChallengeAuthenticationType:digest', 'AuthenticationRealm:XMLRealm'

11.2.4 adf_listUrlConnection
Use with WLST: Online or Offline
11.2.4.1 Description
Use this command to list the connections of the application.

11.2.4.2 Syntax
adf_listURLConnection(appname)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appname</td>
<td>Application name</td>
</tr>
</tbody>
</table>

11.2.4.3 Example
adf_listURLConnection ('myapp')

11.2.5 getADFMArchiveConfig
Use with WLST: Online or Offline.

11.2.5.1 Description
Returns a handle to the ADFMArchiveConfig object for the specified archive. The returned ADFMArchiveConfig object's methods can be used to change application configuration in an archive.

The ADFMArchiveConfig object provides the following methods:

- `setDatabaseJboSQLBuilder([[value]])`—Sets the Database jbo.SQLBuilder attribute.
- `getDatabaseJboSQLBuilder()`—Returns the current value of the jbo.SQLBuilder attribute.
- `setDatabaseJboSQLBuilderClass([[value]])`—Sets the Database jbo.SQLBuilderClass attribute. Value is the full name of the custom builder class.
- `getDatabaseJboSQLBuilderClass()`—Returns the current value of the jbo.SQLBuilderClass attribute.
- `setDefaultRowLimit([[value]])`—Sets the defaults rowLimit attribute. Value is a long specifying the row limit (Default -1).
- `getDefaultRowLimit()`—Returns the current value of the rowLimit attribute.
- `save([toLocation])`—If you specify the toLocation, then the changes will be stored in the target archive file and the original file will remain unchanged. Otherwise, the changes will be saved in the original file itself.

11.2.5.2 Syntax
archiveConfigObject = ADFMAdmin.getADFMArchiveConfig(fromLocation)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromLocation</td>
<td>The name of the ear file, including its complete path.</td>
</tr>
</tbody>
</table>

The syntax for `setDatabaseJboSQLBuilder([[value]])` is:
archiveConfigObject.setDatabaseJboSQLBuilder([[value]])
The syntax for `getDatabaseJboSQLBuilder()` is:

```java
archiveConfigObject.getDatabaseJboSQLBuilder()
```

The syntax for `setDatabaseJboSQLBuilderClass([value])` is:

```java
archiveConfigObject.setDatabaseJboSQLBuilderClass([value])
```

The syntax for `getDatabaseJboSQLBuilderClass` is:

```java
archiveConfigObject.getDatabaseJboSQLBuilderClass()
```

The syntax for `setDefaultRowLimit([value])` is:

```java
archiveConfigObject.setDefaultRowLimit([value])
```

The syntax for `getDefaultRowLimit()` is:

```java
archiveConfigObject.getDefaultRowLimit()
```

The syntax for `save([toLocation])` is:

```java
archiveConfigObject.save([toLocation])
```

### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value</code></td>
<td>The value of the <code>jbo.SQLBuilder</code> attribute. Valid values are: 'Oracle' (Default), 'OLite', 'DB2', 'SQL92', 'SQLServer', or 'Custom'. If 'Custom' is specified, then the <code>jbo.SQLBuilderClass</code> attribute should also be set.</td>
</tr>
</tbody>
</table>

### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value</code></td>
<td>The value of the <code>jbo.SQLBuilderClass</code> attribute.</td>
</tr>
</tbody>
</table>

### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>value</code></td>
<td>The value of the <code>rowLimit</code> attribute.</td>
</tr>
</tbody>
</table>

### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>toLocation</code></td>
<td>The file name along with the absolute path to store the changes.</td>
</tr>
</tbody>
</table>

#### 11.2.5.3 Example

In the following example, the `jbo.SQLBuilder` attribute is set to 'DB2'.

```java
wls:/offline> archive = ADFMAAdmin.getADFMArchiveConfig(fromLocation='/tmp/testArchive.ear')
wls:/offline> archive.setDatabaseJboSQLBuilder(value='DB2')
wls:/offline> archive.save()
```

In the following example, the `jbo.SQLBuilder` attribute is removed so that application default is used.

```java
wls:/offline> archive = ADFMAAdmin.getADFMArchiveConfig(fromLocation='/tmp/testArchive.ear')
wls:/offline> archive.setDatabaseJboSQLBuilder()
wls:/offline> archive.save(toLocation='/tmp/targetArchive.ear')
```
In the following example, the `jbo.SQLBuilder` attribute is set to 'Custom', and the `jbo.SQLBuilderClass` attribute is set to the class 'com.example.CustomBuilder'.

```java
wls:/offline> archive = ADFMAAdmin.getADFMArchiveConfig(fromLocation='/tmp/testArchive.ear')
wls:/offline> archive.setDatabaseJboSQLBuilder('Custom')
wls:/offline> archive.setDatabaseJboSQLBuilderClass('com.example.CustomBuilder')
wls:/offline> archive.save(toLocation='/tmp/targetArchive.ear')
```

In the following example, the `rowLimit` attribute is set to 100.

```java
wls:/offline> archive = getADFMArchiveConfig(fromLocation='/tmp/testArchive.ear')
wls:/offline> archive.setDefaultRowLimit(100)
wls:/offline> archive.save(toLocation='/tmp/targetArchive.ear')
```

### 11.2.6 exportJarVersions

Use with WLST: Offline.

#### 11.2.6.1 Description

Use to export CSV format of jars versions from current ORACLE_HOME at a specified location. Exported jars versions information can be opened in Oracle OpenOffice or MS Excel.

#### 11.2.6.2 Syntax

`exportJarVersions(path)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>path</td>
<td>Location to extract jars versions.</td>
</tr>
</tbody>
</table>

#### 11.2.6.3 Example

This example shows how jars versions are exported to `/tmp/export-MyApp-Versions.csv`. R/W privileges for the CSV file need to be verified.

```java
wls:/offline>exportJarVersions('/tmp/export-MyApp-Versions.csv')
```

### 11.2.7 exportApplicationJarVersions

Use with WLST: Online.

#### 11.2.7.1 Description

Used to export CSV format of runtime jars versions of a specified application at a specified location.

#### 11.2.7.2 Syntax

`exportApplicationJarVersions(applicationName, path)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationName</td>
<td>Application name to export jars versions</td>
</tr>
<tr>
<td>path</td>
<td>Location to export jars versions.</td>
</tr>
</tbody>
</table>
11.2.7.3 Example
This example shows how MyApp runtime jars versions are exported to
/tmp/export-MyApp-Versions.csv. R/W privileges for the CSV file need to be verified.

wls:/DefaultDomain/serverConfig>exportApplicationJarVersions('MyApp',
'/tmp/export-MyApp-Versions.csv')

11.2.8 exportApplicationSelectedJarVersions
Use with WLST: Online.

11.2.8.1 Description
Used to export CSV format of jars versions of selected jars at a specified location.

11.2.8.2 Syntax
exportApplicationSelectedJarVersions(applicationName, path, jarsLocation)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationName</td>
<td>Application name to export jars versions.</td>
</tr>
<tr>
<td>path</td>
<td>Location to extract jars versions.</td>
</tr>
<tr>
<td>jarsLocation</td>
<td>Optional list of selected jars. If not specified, default jars runtime</td>
</tr>
<tr>
<td></td>
<td>version list from %WLSDOMAIN%/config/fmwconfig/Versions.xml will be</td>
</tr>
<tr>
<td></td>
<td>exported. If the selectedJars property in Versions.xml is empty, version</td>
</tr>
<tr>
<td></td>
<td>information of adfm.jar, adf-richclient-impl-11.jar,</td>
</tr>
<tr>
<td></td>
<td>adf-controller.jar, adf-pageflow-impl.jar, adf-share-support.jar and mdsrt.jar will be exported.</td>
</tr>
</tbody>
</table>

11.2.8.3 Example
This example shows how jars versions are exported to
/tmp/export-MyApp-Versions.csv using the selectedJars property of the
Versions.xml file. In this case, since the jarsLocation parameter is not specified, the libraries listed in the selectedJars property of the Versions.xml file will be exported. R/W privileges for the CSV file need to be verified.

wls:/offline>exportApplicationSelectedJarVersions('MyApp',
'/tmp/export-MyApp-Versions.csv')

Versions.xml
<Diagnosics xmlns="xmlns.oracle.com/adf/diagnostics">
  <Versions xmlns="xmlns.oracle.com/adf/diagnostics/versions"
    exportVersionsOnApplicationStartup="true"
    selectedJars="$ORACLE_HOME$/modules/oracle.adf.model_11.1.1/adfm.jar;
    $ORACLE_HOME$/modules/oracle.adf.view_11.1.1/adf-richclient-impl-11.jar;
    $ORACLE_HOME$/modules/oracle.adf.controller_11.1.1/adf-controller.jar;
    $ORACLE_HOME$/modules/oracle.adf.pageflow_11.1.1/adf-pageflow-impl.jar;
    $ORACLE_HOME$/modules/oracle.adf.share_11.1.1/adf-share-support.jar;
    $ORACLE_HOME$/modules/oracle.mds_11.1.1/mdsrt.jar" />
</Diagnosics>
This example shows how jars versions are exported to /tmp/export-MyApp-Versions.csv using the jarsLocation parameter. In this case, the libraries passed explicitly in the jarsLocation parameter will be exported. R/W privileges for the CSV file need to be verified.

```
```

### 11.2.9 createWebServiceConnection

Use with WLST: Online.

#### 11.2.9.1 Description

Used to create a Web Service connection for an ADF application.

Returns a set of service name and port names in the format {serviceName: List of portName}.

For example: 

```
{'PolicyReferenceEchoBeanService': array(java.lang.String,["PolicyReferenceEchoBeanPort"])}
```

#### 11.2.9.2 Syntax

`createWebServiceConnection(appName, wsConnName, wsdlUrlStr, readerProps)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the ADF application for which you want to create a Web service connection.</td>
</tr>
<tr>
<td>wsConnName</td>
<td>Name of the new Web service connection.</td>
</tr>
<tr>
<td>wsdlUrlStr</td>
<td>Name of the service WSDL URL string.</td>
</tr>
<tr>
<td>readerProps</td>
<td>The optional WSDL reader properties. For example:</td>
</tr>
<tr>
<td></td>
<td>'{&quot;wsdl.reader.proxy.host=proxy.my.com&quot;, &quot;wsdl.reader.proxy.port=80&quot;}'}</td>
</tr>
</tbody>
</table>

#### 11.2.9.3 Example

This example shows how to create a Web service connection for an ADF application myapp, with a Web service connection mywsconn, and a WSDL URL string of http://myserver/myservice?WSDL.

```
createWebServiceConnection('myapp','mywsconn','http://myserver/myservice?WSDL')
```

### 11.2.10 listWebServiceConnection

Use with WLST: Online.

#### 11.2.10.1 Description

Used to list the Web service connections associated with an ADF application.
11.2.10.2 Syntax

```
listWebServiceConnection(appName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the ADF application for which you want to list its Web service connections.</td>
</tr>
</tbody>
</table>

11.2.10.3 Example

This example shows how list the Web service connections for the application `myapp`.

```
listWebServiceConnection('myapp')
```

11.2.11 `deleteWebServiceConnection`

Use with WLST: Online.

11.2.11.1 Description

Used to delete a Web service connection associated with an ADF application.

11.2.11.2 Syntax

```
listWebServiceConnection(appName, wsConnName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the ADF application for which you want to delete a Web service connection.</td>
</tr>
<tr>
<td>wsConnName</td>
<td>Name of the Web service connection you want to delete.</td>
</tr>
</tbody>
</table>

11.2.11.3 Example

This example shows how delete the Web service connection `mywsconn` from the application `myapp`.

```
deleteWebServiceConnection('myapp','mywsconn')
```

11.2.12 `listUpgradeHandlers`

Use with WLST: Online.

11.2.12.1 Description

Used to list all upgrade handlers of an application.

11.2.12.2 Syntax

```
listUpgradeHandlers(applicationName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationName</td>
<td>Application name to list upgrade handlers.</td>
</tr>
</tbody>
</table>
11.2.13 upgradeADFMetadataApp

Use with WLST: Online.

11.2.13.1 Description

Used to upgrade all registered ADF Metadata of an application.

11.2.13.2 Syntax

upgradeADFMetadataApp(applicationName,siteCC)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationName</td>
<td>Application name to upgrade handlers.</td>
</tr>
<tr>
<td>siteCC</td>
<td>Site Customization Class name.</td>
</tr>
</tbody>
</table>

11.2.13.3 Example

In the following example, all registered ADF Metadata of the application are upgraded.

wls:/DefaultDomain/serverConfig>upgradeADFMetadataApp('MyApp','oracle.apps.fnd.app lcore.customization.SiteCC')

11.2.14 upgradeADFMetadataAppHandlers

Use with WLST: Online.

11.2.14.1 Description

Used to upgrade selected registered ADF Metadata of an application.

11.2.14.2 Syntax

upgradeADFMetadataAppHandlers(applicationName,sitecc,handlers)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationName</td>
<td>Application name to upgrade handlers.</td>
</tr>
<tr>
<td>siteCC</td>
<td>Site Customization Class name.</td>
</tr>
<tr>
<td>handlers</td>
<td>Registered handlers to be updated.</td>
</tr>
</tbody>
</table>

11.2.14.3 Example

In the following example, selected registered ADF Metadata of the application are upgraded.

11.2.15 upgradeADFMetadata

Use with WLST: Online.

11.2.15.1 Description

Used to upgrade all registered ADF Metadata of all the applications.

11.2.15.2 Syntax

`upgradeADFMetadata(siteCC)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>siteCC</td>
<td>Site Customization Class name.</td>
</tr>
</tbody>
</table>

11.2.15.3 Example

In the following example, all registered ADF Metadata of all the applications are upgraded.

`wls:/DefaultDomain/serverConfig>upgradeADFMetadata('oracle.apps.fnd.application.customization.SiteCC')`

11.2.16 upgradeADFMetadataHandlers

Use with WLST: Online.

11.2.16.1 Description

Used to upgrade selected registered ADF Metadata of all the applications.

11.2.16.2 Syntax

`upgradeADFMetadataHandlers(sitecc, handlers)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>siteCC</td>
<td>Site Customization Class name.</td>
</tr>
<tr>
<td>handlers</td>
<td>Registered handlers to be updated.</td>
</tr>
</tbody>
</table>

11.2.16.3 Example

In the following example, selected registered ADF Metadata of all the applications are upgraded.

`wls:/DefaultDomain/serverConfig>upgradeADFMetadataHandlers('oracle.apps.fnd.application.customization.SiteCC','http://xmlns.oracle.com/adf/metadataUpgrade/bc4j/propertiesUpgrade')`
This chapter provides detailed descriptions of custom WLST commands for Oracle Portal, including command syntax, arguments and command examples.

Portal custom WLST commands are extensions to the WLST commands and are specific to Oracle Portal. Table 12–1 lists the Portal custom WLST command categories.

For additional information about administration and configuration of Portal, see the Oracle Portal Configuration Guide.

Note: To use these commands, you must invoke WLST from the Oracle home in which the component has been installed. See "Using Custom WLST Commands" in the Oracle Fusion Middleware Administrator’s Guide.

<table>
<thead>
<tr>
<th>Command category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database Access Descriptor Commands</td>
<td>Create, edit, or delete a general DAD or Portal DAD.</td>
</tr>
<tr>
<td>Configuration Commands</td>
<td>The Configuration commands:</td>
</tr>
<tr>
<td></td>
<td>- List and update the WebCache configuration and Oracle Internet Directory data</td>
</tr>
<tr>
<td></td>
<td>- Configure the Portal cache, Portal Page Engine, and Portal mid-tier</td>
</tr>
<tr>
<td></td>
<td>- List Portal site configuration.</td>
</tr>
</tbody>
</table>

12.1 Database Access Descriptor Commands

A Database Access Descriptor (DAD) is a set of values that specify how an application connects to an Oracle database to fulfill an HTTP request. The information in the DAD includes the user name (which also specifies the schema and the privileges), password, connect string, and globalization support language of the database.

There are two types of DADs: general DAD and portal DAD. An Oracle Portal middle tier uses a Portal DAD to access the Oracle Metadata Repository. For information about general DADs, refer to the Oracle Fusion Middleware Administrator’s Guide for Oracle HTTP Server.

Use the Database Access Descriptor commands listed in Table 12–2 to create, edit, or delete a Portal DAD from the WLST command-line scripting interface. Based on your actions, the portal_dads.conf file is updated.
### 12.1.1 listDads

**Command Category:** Database Access Descriptor Commands  
**Use with WLST:** Online

#### 12.1.1.1 Description

Lists the parameters specified in all the Database Access Descriptors (both general DADs and Portal DADs).

#### 12.1.1.2 Syntax

```plaintext
listDads()
```

#### 12.1.1.3 Example

The following example lists the various DADs in the domain.

```
listDads()
------------
/pls/portal1
Schema: h1user
Connect String: foo.oracle.com:1521:orcl
NLS Language: "AMERICAN_AMERICA.AL32UTF8"
```

### 12.1.2 createPortalDad

**Command Category:** Database Access Descriptor Commands  
**Use with WLST:** Online

#### 12.1.2.1 Description

Creates a Portal Database Access Descriptor.

#### 12.1.2.2 Syntax

```plaintext
createPortalDad (name, schema, password, [connect_string], nls_language)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the Database Access Descriptor.</td>
</tr>
<tr>
<td>schema</td>
<td>The Portal database account user name.</td>
</tr>
<tr>
<td>password</td>
<td>The Portal database account password.</td>
</tr>
</tbody>
</table>
12.1.2 Example
The following example creates the portal1 Portal DAD based on the specified arguments.

```
createPortalDad(name='portal1', schema='schema', password='welcome1', connect_string='foo.oracle.com:1521:orcl', nls_language='AMERICAN_AMERICA.AL32UTF8')
```

12.1.3 updatePortalDad
Command Category: Database Access Descriptor Commands
Use with WLST: Online

12.1.3.1 Description
Updates the attributes of the Portal Database Access Descriptor.

12.1.3.2 Syntax
```
updatePortalDad (name, [schema], [password], [connect_string], [nls_language])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>name</code></td>
<td>Name of the Database Access Descriptor. This name cannot be changed during update.</td>
</tr>
<tr>
<td><code>schema</code></td>
<td>Optional. The Portal database account user name.</td>
</tr>
<tr>
<td><code>password</code></td>
<td>Optional. The Portal database account password.</td>
</tr>
<tr>
<td><code>connect_string</code></td>
<td>Optional. The connection string used to connect to a remote database.</td>
</tr>
<tr>
<td></td>
<td>Connect string may be host name: port number: connect string. The connect string format may be ServiceNameFormat (host:port:database_service_name), SIDFormat (host:port:database_sid), or TNSFormat (TNS alias or the whole TNS entry).</td>
</tr>
<tr>
<td><code>nls_language</code></td>
<td>Optional. The globalization support language of the Portal database that is represented by this DAD. This setting overrides the NLS_LANG environment variable for a database session and defines some important globalization support properties of the response, including the response character set. Make sure that this language setting matches the NLS_LANG of the back-end database.</td>
</tr>
</tbody>
</table>
12.1.3.3 Example
The following example updates the portal1 Portal DAD based on the specified arguments.

updatePortalDad(name='portal1',schema='user1',password='welcome2',connect_string='foo.oracle.com:1521:orcl',nls_language='AMERICAN_AMERICA.AL32UTF8')

12.1.4 deletePortalDad
Command Category: Database Access Descriptor Commands
Use with WLST: Online

12.1.4.1 Description
Deletes a Portal Database Access Descriptor.

12.1.4.2 Syntax
deletePortalDad(name)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the Portal Database Access Descriptor.</td>
</tr>
</tbody>
</table>

12.1.4.3 Example
The following example deletes the portal1 Portal DAD entry from the portal_dads.conf file.

deletePortalDad(name='portal1')

12.2 Configuration Commands
Use the Configuration commands in Table 12–3 to view and configure Portal cache, WebCache, Oracle Internet Directory data and so on.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>configurePortalCache</td>
<td>Update the attributes of the Portal cache.</td>
<td>Online</td>
</tr>
<tr>
<td>configurePortalPageEngine</td>
<td>Update the attributes of the Portal mid-tier.</td>
<td>Online</td>
</tr>
<tr>
<td>listPortalWebcacheConfigAttributes</td>
<td>List the attributes of WebCache configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>listPortalSiteConfigAttributes</td>
<td>List the attributes of Portal site configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>listPortalOIDConfigAttributes</td>
<td>List the attributes of Oracle Internet Directory configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>setPortalWebcacheConfig</td>
<td>Update the attributes of the WebCache configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>setPortalOIDConfig</td>
<td>Update the attributes of the Oracle Internet Directory configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>setPortalMidtierConfig</td>
<td>Update the attributes of the Portal mid-tier configuration.</td>
<td>Online</td>
</tr>
</tbody>
</table>
12.2.1 configurePortalCache

Command Category: Configuration Commands

Use with WLST: Online

12.2.1.1 Description

Portal cache is a file system-based cache for Oracle Portal pages and portlets. Portal cache supports validation-based caching and expiry-based caching. Portal cache consists of both Portal content cache and session cache.

This command updates the attributes of the Portal cache. These configuration details are maintained in the `<Middleware Home>/user_projects/domains/<DOMAIN_HOME>/servers/WLS_PORTAL.stage/portal/configuration/portal_cache.conf` file.

12.2.1.2 Syntax

```text
configurePortalCache(enable, directory, total_size, max_size, cleanup_time, max_age)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>enable</td>
<td>Optional. Enables (On) or disables (Off) portal content and session caching.</td>
</tr>
<tr>
<td>directory</td>
<td>Optional. The directory where cached content is stored. Make sure that this directory exists and has read-write access.</td>
</tr>
<tr>
<td>total_size</td>
<td>Optional. The total amount of disk space (in megabytes) that the Portal cache may use. The maximum value allowed is 4 GB.</td>
</tr>
<tr>
<td>max_size</td>
<td>Optional. The maximum size (in bytes) for all cached files. The maximum value allowed is 4 GB. Any dynamically generated content that exceeds this limit is not cached.</td>
</tr>
<tr>
<td>cleanup_time</td>
<td>Optional. The time at which to start the cleanup of the cache storage. Use the [Sunday-Saturday, Everyday, Everymonth][hh:mm] format to define the exact day and time in which cleanup should occur.</td>
</tr>
<tr>
<td>max_age</td>
<td>Optional. Maximum age of a single cached document. This setting ensures the cache system does not contain any old content. Old cache files are removed to make space for new cache files. The default is 30 days.</td>
</tr>
</tbody>
</table>

12.2.1.3 Example

The following example configures the Portal cache.

```text
configurePortalCache(enable=true, directory='/scratch/user/installs/Inst_1/cache/PortalComponent/portal', total_size=10101010, max_size=12300033, cleanup_time='Everyday 11:00', max_age=20)
```

12.2.2 configurePortalPageEngine

Command Category: Configuration Commands

Use with WLST: Online
12.2.2.1 Description

The Oracle Fusion Middleware Portal architecture is designed around a three-tier architecture that allows any browser to connect to it. This flexible architecture allows each component (browser, Oracle HTTP Server listener, Oracle Database 11g, and Oracle Portal) to be upgraded individually as required.

A part of the Oracle Portal middle tier, the Parallel Page Engine (PPE) is a servlet that runs under Oracle Containers for Java EE and services page requests. The PPE reads page metadata, calls providers for portlet content, accepts provider responses, and assembles the requested page in the specified page layout.

This command updates the properties in the appConfig.xml file, the configuration file that is used by the Portal mid-tier repository servlet. This configuration file is located in the $MWHOME/user_projects/domains/AllClassicDomain/servers/WLS_PORTAL/stage/portal/portal/configuration/ directory.

12.2.2.2 Syntax

configurePortalPageEngine([encrypt_key], [resource_url_key], [use_port], [use_scheme], [x509certfile])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>encrypt_key</td>
<td>Optional. Specifies the HMCA key to obscure the headers used for caching using WebCache. This allows for a more secure cache key, and makes retrieving a cached object by unwanted requests more difficult.</td>
</tr>
<tr>
<td>resource_url_key</td>
<td>Optional. This key, used by the PPE servlet, calculates checksums for URLs that are requested by WSRP and JPDK resource proxying. For WSRP resource proxying to work, the key must be set to an alpha-numeric value of 10 characters or more. In addition, for JPDK proxying, a JNDI environment variable, also called resourceUrlKey, must be set for the provider.</td>
</tr>
<tr>
<td>use_port</td>
<td>Optional. Overrides the port used when the PPE makes requests to the portal. The default, if not specified, is to always use the page request port. Note that if you set useScheme, you must also set the usePort argument. This may be used for other reasons, but mostly it is used when SSL is running between the browser and the PPE but not between the PPE and Portal. In this case, the non-SSL port for loop back requests will be different from the SSL port used by the browser.</td>
</tr>
<tr>
<td>use_scheme</td>
<td>Optional. Overrides the scheme (HTTP or HTTPS) used when the PPE makes requests to the Portal. The default, if not specified, is to always use the page request scheme. Note that if you set useScheme, you must also set the usePort argument.</td>
</tr>
<tr>
<td>x509certfile</td>
<td>Optional. Specifies a file containing a list of certificates to be implicitly trusted by HTTPClient. These certificates are added as trust points to all connections made by HTTPClient using SSL.</td>
</tr>
</tbody>
</table>

12.2.2.3 Example

The following example updates the Portal page engine based on the specified arguments.

```java
configurePortalPageEngine(encrypt_key='encryption key', resource_url_key='foo.oracle.com', use_port=9999, use_scheme='page_engine_1', x509certfile='file')
```
12.2.3 listPortalWebcacheConfigAttributes

Command Category: Configuration Commands
Use with WLST: Online

12.2.3.1 Description
Lists the attributes of WebCache configuration used by the Portal repository.

12.2.3.2 Syntax
listPortalWebcacheConfigAttributes { [dad_name] } 

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>dad_name</td>
<td>Optional. Name of the Database Access Descriptor. Default DAD name is 'portal'.</td>
</tr>
</tbody>
</table>

12.2.3.3 Example
The following example lists the WebCache configuration used by the Portal repository. The WebCache host name to which the invalidation messages are sent, the invalidation user name, password and the invalidation port to which the invalidation messages are sent are listed.

```
listPortalWebcacheConfigAttributes(dad_name='portal1')
listPortalWebcacheConfigAttributes('portal1')
--------------
WebCacheConfig
--------------
WebCache Host: foo.oracle.com
WebCache Invalidation Password: invalidator
WebCache Invalidation Port: 6523
WebCache Invalidation User: invalidator
```

12.2.4 listPortalSiteConfigAttributes

Command Category: Configuration Commands
Use with WLST: Online

12.2.4.1 Description
Lists the attributes of the Portal site configuration.

12.2.4.2 Syntax
listPortalSiteConfigAttributes { [dad_name] } 

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>dad_name</td>
<td>Optional. Name of the Database Access Descriptor. Default DAD name is 'portal'.</td>
</tr>
</tbody>
</table>

12.2.4.3 Example
The following example lists the Portal site configuration. Site protocol can be true or false. HTTP is the protocol when site protocol is false and HTTPS is the protocol when the site protocol is true. The site host name and port number are also listed.

```
listPortalSiteConfigAttributes(dad_name='portal1')
```
listPortalSiteConfigAttributes('portal1')

---------------
SiteConfig
---------------
Site Protocol: false
Site Host: foo.oracle.com
Site Port: 8090

12.2.5 listPortalOIDConfigAttributes

Command Category: Configuration Commands
Use with WLST: Online

12.2.5.1 Description
Lists the attributes of the Oracle Internet Directory configuration.

12.2.5.2 Syntax
listPortalOIDConfigAttributes ([dad_name])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>dad_name</td>
<td>Optional. Name of the Database Access Descriptor. Default DAD name is 'portal'.</td>
</tr>
</tbody>
</table>

12.2.5.3 Example
The following example lists the Oracle Internet Directory data, which includes the Oracle Internet Directory host name and port number.

listPortalOIDConfigAttributes(dad_name='portal1')
listPortalOIDConfigAttributes('portal1')

---------------
OidConfig
---------------
OID Port: 13060
OID Host: foo.oracle.com

12.2.6 setPortalWebcacheConfig

Command Category: Configuration Commands
Use with WLST: Online

12.2.6.1 Description
WebCache offers caching, page assembly, and compression features. Oracle WebCache accelerates the delivery of both static and dynamic Web content, and provides load balancing and failover features for Oracle Fusion Middleware.

This command updates the WebCache configuration.

12.2.6.2 Syntax
setPortalWebcacheConfig([dad_name], [host], [inv_port], [inv_user], [inv_passwd])
12.2.6.3 Example
The following example updates the WebCache configuration based on the specified values.

```
setPortalWebcacheConfig(dad_name='portal1',host='foo.oracle.com',
inv_port= '6523',inv_user= 'invalidator',inv_passwd=' invalidator')
```

12.2.7 setPortalOIDConfig
Command Category: Configuration Commands
Use with WLST: Online

12.2.7.1 Description
Updates the attributes of the Oracle Internet Directory configuration.

12.2.7.2 Syntax
```
setPortalOIDConfig ([dad_name], [host], [port], [protocol], [admin_user],
[admin_passwd])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>dad_name</td>
<td>Optional. Name of the Database Access Descriptor. Default DAD name is 'portal'.</td>
</tr>
<tr>
<td>host</td>
<td>Optional. The Oracle Internet Directory host name.</td>
</tr>
<tr>
<td>port</td>
<td>Optional. The Oracle Internet Directory port number.</td>
</tr>
<tr>
<td>protocol</td>
<td>Optional. The Oracle Internet Directory protocol.</td>
</tr>
<tr>
<td>admin_user</td>
<td>Optional. The Oracle Internet Directory administrator's name.</td>
</tr>
<tr>
<td>admin_passwd</td>
<td>Optional. The Oracle Internet Directory administrator's password.</td>
</tr>
</tbody>
</table>

12.2.7.3 Example
The following example updates the OID configuration based on the specified values.

```
setPortalOIDConfig(dad_name='portal1',
host='foo.oracle.com',port='13060',protocol=false,
admin_user='cn=orcladmin',admin_passwd='oracle1')
```

12.2.8 setPortalMidtierConfig
Command Category: Configuration Commands
Use with WLST: Online

12.2.8.1 Description
Updates the Portal repository with the latest Portal mid-tier configuration.

12.2.8.2 Syntax
```
setPortalMidtierConfig([dad_name], [ohs_host], [ohs_port], [ohs_protocol],
[webcache_host], [webcache_inv_user], [webcache_inv_port],
[webcache_inv_passwd])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>dad_name</code></td>
<td>Optional. Name of the Database Access Descriptor. Default DAD name is 'portal'.</td>
</tr>
<tr>
<td><code>ohs_host</code></td>
<td>Optional. Oracle HTTP Server host name.</td>
</tr>
<tr>
<td><code>ohs_port</code></td>
<td>Optional. Oracle HTTP Server port number.</td>
</tr>
<tr>
<td><code>ohs_protocol</code></td>
<td>Optional. Oracle HTTP Server protocol.</td>
</tr>
<tr>
<td><code>webcache_host</code></td>
<td>Optional. The name of the WebCache host to which invalidation messages are sent.</td>
</tr>
<tr>
<td><code>webcache_inv_user</code></td>
<td>Optional. The WebCache user name used for sending the invalidation messages.</td>
</tr>
<tr>
<td><code>webcache_inv_port</code></td>
<td>Optional. The WebCache port number to which invalidation messages are sent.</td>
</tr>
<tr>
<td><code>webcache_inv_passwd</code></td>
<td>Optional. WebCache invalidation password.</td>
</tr>
</tbody>
</table>

12.2.8.3 Example
The following example updates the Portal mid-tier configuration based on the specified values.
```
setPortalMidtierConfig(dad_name='portal1', ohs_host='foo.oracle.com',
ohs_port='8090', ohs_protocol='false', webcache_host='foo.oracle.com',
webcache_inv_user= 'invalidator', webcache_inv_port='6523',
webcache_inv_passwd='invalidator')
```
Oracle JRF consists of those components not included in the WebLogic Server installation that provide common functionality for Oracle business applications and application frameworks. This chapter provides detailed descriptions of custom WLST commands for Java Required Files (JRF), including command syntax, arguments and command examples.

Oracle JRF consists of a number of independently developed libraries and applications that are deployed into a common location. The following components are considered part of Java Required Files: Oracle Application Development Framework, Oracle Fusion Middleware Audit Framework, Dynamic Monitoring Service, Fabric Common, HTTP Client, Infrastructure Security, Java Object Cache, JMX Framework, JPS, logging, MDS, OJSP, Next, Oracle Web Services, Oracle Web Services Manager, Oracle TopLink, UCP, XDK.

### 13.1 Java Required Files Commands

Use the commands in Table 13–1 to configure a Managed Server or cluster with Java Required Files (JRF) applications and services or to copy the applications and services from one Managed Server or cluster and apply them to another Managed Server or cluster.

In the Use with WLST column, online means the command can only be used when connected to a running server. Offline means the command can only be used when not connected to a running server. Online or offline means the command can be used in both situations.

**Note:** To use these JRF custom WLST commands, you must invoke the WLST script from the Oracle Common home. See "Using Custom WLST Commands" in the Oracle Fusion Middleware Administrator’s Guide.

**Table 13–1  JRF Commands**

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>applyJRF</td>
<td>Configures a Managed Server or cluster with Java Required Files applications and services.</td>
<td>Online or Offline</td>
</tr>
<tr>
<td>cloneDeployments</td>
<td>Copies the applications and services from Managed Server or cluster and applies them to another Managed Server or cluster.</td>
<td>Online or Offline</td>
</tr>
</tbody>
</table>
13.1.1 applyJRF

Use with WLST: Online or Offline

13.1.1.1 Description

Configures a Managed Server or cluster with Java Required Files (JRF). Managed Servers that are added by product templates during the template extension process do not need to be explicitly configured with JRF using this command.

Use the applyJRF command when additional Managed Servers or clusters are added to a domain after it is initially extended with a product template. The applyJRF command is required any time you add a Managed Server to a JRF-only domain, or if you add a Managed Server that has been configured for JRF to a domain that contains other Oracle products.

13.1.1.2 Syntax

applyJRF(target, [domainDir], [shouldUpdateDomain])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>The name of the Managed Server or cluster to be configured with JRF applications and services. A value of an asterisk (*) for the target indicates that all clusters and standalone Managed Servers should be configured with JRF.</td>
</tr>
<tr>
<td>domainDir</td>
<td>The absolute path of the WebLogic Server domain.</td>
</tr>
<tr>
<td>shouldUpdateDomain</td>
<td>An optional boolean flag that controls how domain updates are carried out. When you set it to true (the default), the function implicitly invokes the following offline commands: readDomain() and updateDomain(), or the online commands: edit(), startEdit(), save(), and activate(). When you set it to false, you must call WLST commands to update the domain.</td>
</tr>
</tbody>
</table>

13.1.1.3 Example

The following example configures the Managed Server server1 with JRF:

```
wlst:/offline> applyJRF('server1', '/my_path/user_templates/domains/my_domain')
```

13.1.2 cloneDeployments

Use with WLST: Online or Offline

13.1.2.1 Description

Replicates all deployments targeted to a particular Managed Server or cluster on a second Managed Server or cluster. This command is provided as a convenience to configure a new Managed Server or cluster so that it has the same deployments as a pre-existing Managed Server or cluster.

The cloneDeployments command does not create new Managed Servers, and it does not copy properties other than deployment information to the target Managed Server.

13.1.2.2 Syntax

cloneDeployments(domain, source, target, [shouldUpdateDomain])
The following example replicates the deployments from sourceServer to destinationServer:

```plaintext
wls:/offline> cloneDeployments('my_path/user_templates/domains/my_domain',
    'sourceServer','destinationServer', 'false')
```
This chapter provides detailed descriptions of custom WLST commands for Web services, including command syntax, arguments and command examples.

The following sections describe the WebLogic Scripting Tool (WLST) commands for Oracle Infrastructure Web services (which includes SOA composites, ADF Business Components, and WebCenter services) and Java EE Web services. You can use these commands to manage Web services from the command line.

Topics in this chapter include:

- Section 14.1, "Overview of Web Services WLST Commands"
- Section 14.2, "Web Service and Client Management Commands"
- Section 14.3, "Policy Management Commands"
- Section 14.4, "Policy Set Management Commands"
- Section 14.5, "Oracle WSM Repository Management Commands"
- Section 14.6, "Deployment Descriptor Migration Commands"
- Section 14.7, "Token Issuer Trust Configuration Commands"
- Section 14.8, "Diagnostic Commands"
- Section 14.9, "JKS Keystore Configuration Commands"
- Section 14.10, "OWSM Configuration Commands"
- Section 14.11, "Upgrade OWSM Configuration Command"

For additional details about using these WLST commands for Web services, see the Oracle Fusion Middleware Security and Administrator's Guide for Web Services.

---

**Notes:** To use the Web Services custom WLST commands, you must invoke WLST from the Oracle Common home directory. See "Using Custom WLST Commands" in the Oracle Fusion Middleware Administrator’s Guide.

To display the help for the Web service and client management, policy management, and deployment descriptor migration commands, connect to a running instance of the server and enter `help('WebServices')`.

To display the help for the policy set management, Oracle WSM repository management, token issuer trust configuration, and diagnostic commands, connect to a running instance of the server and enter `help('wsmManage')`. 
14.1 Overview of Web Services WLST Commands

You can use the Web services WLST commands, in online mode, to:

- Perform Web service configuration and Oracle WSM policy management tasks.
- Manage the Oracle WSM Repository
- Migrate post-deployment policy changes persisted in proprietary deployment descriptor (PDD) files for ADF Business Components and WebCenter services and propagate policy changes to all server instances in a domain.
- Check the status of OWSM components.
- View and define trusted issuers and DN lists for SAML signing certificates.

The Web services WLST commands manage deployed, active, and running Web services applications. They can be executed everywhere in WLST online mode, for example:

```
wls:/domain/serverConfig
wls:/domain/domainRuntime
```

The Web services WLST commands perform many of the same functions that you can complete using Fusion Middleware Control. When using the WLST commands to manage a Web service of an ADF or WebCenter application, you can apply the change only to a Web service deployed in an application on a specific server. If the application is deployed in a cluster or multi-server environment, you need to make the same change to each of the servers to which the application is deployed. Additionally, when you set or change an attached policy in ADF and WebCenter Web service and client applications, you must restart the application for the changes to take effect.

In contrast, if you are using the WLST commands to manage a SOA composite, you only need to issue the command once, and the change is propagated to all the server instances in the composite. When you set or change an attached policy in a SOA composite, you do not need to restart it. The SOA fabric runtime engine internally implements all of the policy management changes.

14.1.1 Specifying Application, Composite, and Service Names

The Web service WLST commands configure a Web service for a specific application. Therefore, the application path name has to uniquely identify the application and the server instance to which it is deployed.

**Specifying a Web Service Application Name**

To specify a Web service application in a WLST command, use the following format:

```
[/domain/server/]application[#version_number]
```

Parameters shown in brackets [] are optional. The following examples show the sample format for a Web service application name:

```
/soainfra/AdminServer/HelloWorld#1_0
/soainfra/server1/HelloWorld#1_0
```

If there is only one deployed instance of an application in a domain, you may omit the `domain/server` parameter, as shown in the following example:

```
HelloWorld#1_0
```
In all other instances, the domain/server parameter is required. If it is not specified and WLST finds more than one deployment of the same application on different servers in the domain, you are prompted to specify the domain and the server names.

Web Service client applications are deployed directly to WebLogic Server server instances. Each client application is managed separately. For example, if the application myapp is deployed to both the AdminServer and server1 instances in the domain mydomain, then you need to issue configuration commands to each of the servers using the appropriate application path name:

/mydomain/AdminServer/myapp#1_0
/mydomain/server1/myapp#1_0

Specifying a SOA Composite Name
When there are multiple SOA partition folders in a domain, you must specify the partition name and the composite name using the following format:

partition/composite[version]

The following example shows the sample format for a SOA composite application name:

default/myComposite[1.0]

If there is a single SOA server (non-clustered) and only one SOA partition folder in a domain, you may omit the partition parameter, as shown in the following example:

myComposite[1.0]

Specifying a Service Name
When there are multiple versions (namespaces) of a Web service name for Web Service and Web Service clients, you must specify the namespace and the service name using the following format:

{http://namespace/}serviceName

Note the following:

■ For Web service and client management commands, and policy management commands, you do not need to enter the namespace if there is only one service name qualified. If there are multiple versions of the service and you do not specify the namespace with the service name, an exception is thrown.

■ The namespace ({http://namespace/}) should not be included for a SOA composite.

■ For policy set management commands, both the namespace and service name are required for Web Service and Web Service Client (ws-service and ws-client) resource types.

For more information, see "Determining the Namespace for a Web Service" in Oracle Fusion Middleware Security and Administrator’s Guide for Web Services.

14.1.2 Web Services WLST Command Categories

Web services WLST commands are divided into the categories described in Table 14–1.
Web Service and Client Management Commands

Use the WLST commands listed in Table 14–2 to view and manage Web services for deployed, active, and running Web service applications.

### Table 14–2 Web Service and Client Management WLST Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>listWebServices</td>
<td>List the Web service information for an application, composite, or domain.</td>
<td>Online</td>
</tr>
<tr>
<td>listWebServicePorts</td>
<td>List the Web service ports for a Web service application or SOA composite.</td>
<td>Online</td>
</tr>
<tr>
<td>listWebServiceConfiguration</td>
<td>List Web services and port configuration for an application or SOA composite.</td>
<td>Online</td>
</tr>
<tr>
<td>setWebServiceConfiguration</td>
<td>Set or change the Web service port configuration for a Web service application or SOA composite.</td>
<td>Online</td>
</tr>
<tr>
<td>listWebServiceClients</td>
<td>List Web service client information for an application, SOA composite, or domain.</td>
<td>Online</td>
</tr>
</tbody>
</table>
14.2.1 listWebServices

Command Category: Web Service and Client Management

Use with WLST: Online

14.2.1.1 Description

Lists the Web service information for an application, SOA composite, or domain. If you don’t specify a Web service application or a SOA composite, the command lists all services in all applications and composites for every server instance in the domain.

You can specify the amount of information to be displayed in the output using the detail argument. When specified, the output provides endpoint (port) and policy details for all applications and composites in the domain, the secure status of the endpoints, any configuration overrides and constraints, and if the endpoints have a valid configuration. A subject is considered secure if the policies attached to it (either directly or globally) enforce authentication, authorization, or message protection behaviors. Because you can specify the priority of a global or directly attached policy (using the reference.priority configuration override), the effective field indicates if the directly attached policies are in effect for the endpoint.

Note that to simplify endpoint management, all directly attached policies are shown in the output regardless of whether they are in effect. In contrast, only globally attached policies that are in effect for the endpoint are displayed. For more information, see "How the Effective Set of Policies is Calculated" in Oracle Fusion Middleware Security and Administrator’s Guide for Web Services.

The output is listed by each application deployed as shown in the following example:

```
/domin/server/application#version_number:
    moduleName=helloModule, moduleType=web,
    serviceName={http://namespace/}service
/soainfra/AdminServer/soa-infra:
    compositeName=default/HelloWorld[1.0], moduleType=soa, serviceName=service
```

---

Table 14–2 (Cont.) Web Service and Client Management WLST Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>listWebServiceClientPorts</td>
<td>List Web service client ports information for an application or SOA composite.</td>
<td>Online</td>
</tr>
<tr>
<td>listWebServiceClientStubProperties</td>
<td>List Web service client port stub properties for an application or SOA composite.</td>
<td>Online</td>
</tr>
<tr>
<td>setWebServiceClientStubProperty</td>
<td>Set, change, or delete a single stub property of a Web service client port for an application or SOA composite.</td>
<td>Online</td>
</tr>
<tr>
<td>setWebServiceClientStubProperties</td>
<td>Configure the set of stub properties of a Web service client port for an application or SOA composite.</td>
<td>Online</td>
</tr>
</tbody>
</table>
### 14.2.1.2 Syntax

```
listWebServices {application, composite, [detail]}
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application for which you want to list the Web services. For example, <code>/domain/server/application#version_number</code> If specified, all Web services in the application are listed.</td>
</tr>
<tr>
<td>composite</td>
<td>Name of the SOA composite for which you want to list the Web services. For example, <code>default/HelloWorld[1.0]</code> If specified, all Web services in the composite are listed.</td>
</tr>
</tbody>
</table>
| detail       | Optional. Specifies whether to list port and policy details for the Web service. Valid values are:  
  - `true`—Output includes details about the service, the port, and the policies.  
  - `false`—Output lists only the services. The default is `false`. |

### 14.2.1.3 Examples

The following example for an Oracle Infrastructure Web service lists all the Web services in all applications and composites in the domain. Sample output is shown in this example.

```wls:/soainfra/serverConfig> listWebServices()
```

```
/soainfra/AdminServer/soa-infra:
  compositeName=default/HelloWorld[1.0], moduleType=soa, serviceName=service
  compositeName=default/Project1[1.0], moduleType=soa,
  serviceName=bpelprocess1_client_ep

/soainfra/AdminServer/HelloWorld#1_0:
  moduleName=j2wbasicPolicy, moduleType=web, serviceName=WssUsernameService
```

The following example for an Oracle Infrastructure Web service sets the detail argument to `true`. Sample output is shown in this example. Note that the directly attached policy is not in effect for the endpoint TestPort in the application `jaxws-sut`.

```wls:/jrfServer_domain/serverConfig> listWebServices(detail='true')
```

```
/jrfServer_domain/jrfServer_admin/jaxws-sut-no-policy:
  moduleName=jaxws-service, moduleType=web, serviceName=(http://namespace/)TestService
  enableTestPage: true
  enableWSDL: true

  TestPort
  http://host.example.com:9315/jaxws-service/TestService
  enable: true
  enableREST: false
  enableSOAP: true
```

### Notes

The `listWebServices` command output does not include details on SOA components, including policy attachments.

For applications assembled prior to PS5, the namespace is not displayed with the `serviceName` in the output.
maxRequestSize: -1
loggingLevel: NULL
wsat.flowOption: NEVER
wsat.version: DEFAULT
Constraint: No Constraint

global security : oracle/wss_saml_or_username_token_service_policy, enabled=true

/policysets/global/all-domains-default-web-service-policies : Domain(***)
  reference.priority=1
  Constraint: HTTPHeader('VIRTUAL_HOST_TYPE','external')
  {global} security : oracle/wss10_message_protection_service_policy, enabled=true
  /policysets/global/domainExternal : Domain(***)
Attached policy or policies are valid; endpoint is secure.

/jrfServer_domain/jrfServer_admin/jaxws-sut :
  moduleName=jaxws-sut-service, moduleType=web,
  serviceName={http://namespace/}TestService
  enableTestPage: true
  enableWSDL: true

TestPort
http://host.example.com:9315/jaxws-sut-service/TestService
  enable: true
  enableREST: false
  enableSOAP: true
  maxRequestSize: -1
  loggingLevel: NULL
  wsat.flowOption: NEVER
  wsat.version: DEFAULT
  management : oracle/log_policy, enabled=true
  security : oracle/wss_username_token_service_policy , enabled=true, effective=false
  Constraint: No Constraint
  {global} security : oracle/wss_saml_or_username_token_service_policy, enabled=true

/policysets/global/all-domains-default-web-service-policies : Domain(***)
  reference.priority=1
  Constraint: HTTPHeader('VIRTUAL_HOST_TYPE','external')
  {global} security : oracle/wss10_message_protection_service_policy, enabled=true
  /policysets/global/domainExternal : Domain(***)
Attached policy or policies are valid; endpoint is secure.

The following example for a Java EE Web service sets the detail argument to true. Sample output is shown in this example. The output lists all the Web services in all applications and composites in the domain.

/wls-domain/AdminServer/hellows :
  moduleName=hellows#1!HelloWSService, moduleType=wls,
  serviceName=HelloWSService
  HelloWSPort
No policies attached; endpoint is not secure.

### 14.2.2 listWebServicePorts

Command Category: Web Service and Client Management
Use with WLST: Online

14.2.2.1 Description
List the Web service port names and the endpoint URLs for a Web service application or SOA composite.
The output will display the port name and endpoint URL of the Web service port. For example:

$$\text{JRFWssUsernamePort} \quad \text{http://localhost:7001/j2wbasicPolicy/WssUsername}$$

14.2.2.2 Syntax
listWebServicePorts(application,moduleOrCompName,moduleType,serviceName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application for which you want to list the Web services port information. For example, <code>/domain/server/application#version_number</code></td>
</tr>
<tr>
<td>moduleOrCompName</td>
<td>Name of the Web module or SOA composite (for example HelloWorld[1.0]) for which you want to list the Web services port information.</td>
</tr>
<tr>
<td>moduleType</td>
<td>Module type. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>- soa—SOA composite.</td>
</tr>
<tr>
<td></td>
<td>- web—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).</td>
</tr>
<tr>
<td></td>
<td>- wls—Java EE Web services.</td>
</tr>
<tr>
<td>serviceName</td>
<td>Name of the Web service in the application or SOA composite for which you want to list the port information. For example, [<a href="http://namespace/%5DserviceName">http://namespace/]serviceName</a>. Note that the namespace ([<a href="http://namespace/">http://namespace/</a>]) should not be included for a SOA composite.</td>
</tr>
</tbody>
</table>

14.2.2.3 Example
The following example lists the Web service ports and endpoint URLs for the j2wbasicPolicy service in the soainfra/AdminServer/HelloWorld#1_0 application. Note that the WssUsernameService module name is specified, and the moduleType is set to web.

```
<soainfra:/serverConfig> listWebServicePorts
('soainfra/AdminServer/HelloWorld#1_0',
 'WssUsernameService','web',[http://namespace/]j2wbasicPolicy')

JRFWssUsernamePort http://localhost:7001/j2wbasicPolicy/WssUsername
```

The following example lists the Web service ports and endpoint URLs for the Java EE Web service helloWorldJaxws in the wls-domain/AdminServer/helloWorldJaxws application. Note that the moduleType is set to wls.

```
wls:/wls-domain/serverConfig> listWebServicePorts
('wls-domain/AdminServer/helloWorldJaxws','helloWorldJaxws#1!helloWorldJaxws',
```
14.2.3 listWebServiceConfiguration

**Notes:** This command applies to Oracle Infrastructure Web services only.

This command is deprecated and will be removed in a future release.

Command Category: Web Service and Client Management

Use with WLST: Online

### 14.2.3.1 Description

List the Web service port configuration for a Web service application or SOA composite.

The output will display the configuration information for the Web service port. For example:

```
enableREST: false
maxRequestSize: -1
```

### 14.2.3.2 Syntax

```
listWebServiceConfiguration{application, moduleOrCompName, moduleType, serviceName, [subjectName]}
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application for which you want to list the Web services port configuration. For example, <code>/domain/server/application#version_number</code> To list the port configuration for a Web service application, this argument is required.</td>
</tr>
<tr>
<td>moduleOrCompName</td>
<td>Name of the Web module or SOA composite (for example HelloWorld[1.0]) for which you want to list the Web services port configuration. To list the port configuration for a SOA composite, the composite name is required (for example default/HelloWorld[1.0]), and the moduleType argument must be set to soa.</td>
</tr>
<tr>
<td>moduleType</td>
<td>Module type. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>- soa—SOA composite.</td>
</tr>
<tr>
<td></td>
<td>- web—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).</td>
</tr>
<tr>
<td>serviceName</td>
<td>Name of the Web service in the application or SOA composite for which you want to list the port configuration. For example, [<a href="http://namespace/%5DserviceName">http://namespace/]serviceName</a>. Note that the namespace ([<a href="http://namespace/">http://namespace/</a>]) should not be included for a SOA composite.</td>
</tr>
<tr>
<td>subjectName</td>
<td>Optional. Policy subject, port, or operation for which you want to list configuration information.</td>
</tr>
</tbody>
</table>
14.2.3 Example

The following example lists the Web service and port configuration information for the application HelloWorld#1_0 for the server soa1 in the domain soainfra. In this example, the Web module name is j2wbasicPolicy, the service name is WssUsernameService, and the subject is a port named JRFWssUsernamePort.

```
  wls:/wls-domain/serverConfig>listWebServiceConfiguration
  ('/soainfra/soa1/HelloWorld#1_0','j2wbasicPolicy','web',
   '{http://namespace/}WssUsernameService','JRFWssUsernamePort')
```

14.2.4 setWebServiceConfiguration

**Note:** This command applies to Oracle Infrastructure Web services only.
This command is deprecated and will be removed in a future release.

Command Category: Web Service and Client Management
Use with WLST: Online

**14.2.4.1 Description**

Set or change the Web service port configuration for a Web service application or SOA composite.

Additional information about using this command is provided in "Configuring the Web Service Endpoint" in *Oracle Fusion Middleware Security and Administrator’s Guide for Web Services*.

**14.2.4.2 Syntax**

```
setWebServiceConfiguration(application,moduleOrCompName,moduleType,serviceName,
subjectName,itemProperties)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application for which you want to set or change the Web services port configuration. For example, /domain/server/application#version_number To set or change the port configuration for a Web service application, this argument is required.</td>
</tr>
<tr>
<td>moduleOrCompName</td>
<td>Name of the Web module or SOA composite (for example HelloWorld[1.0]) for which you want to set or change the Web services port configuration. To set or change the port configuration for a SOA composite, the composite name is required (for example default/HelloWorld[1.0]), and the moduleType argument must be set to soa.</td>
</tr>
<tr>
<td>moduleType</td>
<td>Module type. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>- soa—SOA composite.</td>
</tr>
<tr>
<td></td>
<td>- web—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).</td>
</tr>
</tbody>
</table>
### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceName</td>
<td>Name of the Web service in the application or SOA composite for which you want to set or change the port configuration. For example, [<a href="http://namespace/%5DserviceName">http://namespace/]serviceName</a>. Note that the namespace ([<a href="http://namespace/">http://namespace/</a>]) should not be included for a SOA composite.</td>
</tr>
<tr>
<td>subjectName</td>
<td>Policy subject, port or operation name for which you want to set or change the configuration information.</td>
</tr>
<tr>
<td>itemProperties</td>
<td>Configurable properties that you can set or change. Specify the properties using the following format: ((\text{&quot;name&quot;,&quot;value&quot;})) Valid port configuration name and value pairs are as follows:</td>
</tr>
<tr>
<td></td>
<td>■ enable—true or false. Default is true.</td>
</tr>
<tr>
<td></td>
<td>■ enableTestPage—true or false. Default is true.</td>
</tr>
<tr>
<td></td>
<td>■ enableWSDL—true or false. Default is true.</td>
</tr>
<tr>
<td></td>
<td>■ enableREST—true or false. Default is false.</td>
</tr>
<tr>
<td></td>
<td>■ maxRequestSize—long integer, -1 for values not set. The default is -1.</td>
</tr>
<tr>
<td></td>
<td>■ loggingLevel—NULL, FINEST...SEVERE (java.util.logging.Level). The default is NULL.</td>
</tr>
<tr>
<td></td>
<td>■ wsat.flowOption—Atomic transaction flow option. Valid values are: NEVER—Do not export transaction coordination context. (This is the default.), SUPPORTS—Export transaction coordination context if transaction is available, MANDATORY—Export transaction coordination context. An exception is thrown if there is no active transaction. This property is valid for SOA services only.</td>
</tr>
<tr>
<td></td>
<td>■ wsat.version—Atomic transaction version. Valid values are: WSAT10, WSAT11, WSAT12, and DEFAULT. This property is valid for SOA services only.</td>
</tr>
</tbody>
</table>

For additional information about the atomic transaction properties, see 'Configuring Web Services Atomic Transactions' in Oracle Fusion Middleware Security and Administrator’s Guide for Web Services.

**Note:** If any configuration item contains an unrecognized property name or invalid value, this set command is rejected and an error message is displayed.

#### 14.2.4.3 Example

The following example enables the port JRFWssUsernamePort for the service WssUsernameService in the Web module j2wbasicPolicy. The service is in the application HelloWorld#1_0 for the server soa1 in the domain soainfra.

```bash
wls:/wls-domain/serverConfig> setWebServiceConfiguration
'/(soainfra/soa1/HelloWorld#1_0','j2wbasicPolicy','web',
'(http://namespace/)WssUsernameService','JRFWssUsernamePort',{("enable", "true")})
```

#### 14.2.5 listWebServiceClients

**Command Category:** Web Service and Client Management  
**Use with WLST:** Online
14.2.5.1 Description

List Web service clients information for an application, SOA composite, or domain. If neither an application nor a composite is specified, the command lists information about all Web service clients in all applications and composites for every server instance in the domain.

You can specify the amount of information to be displayed in the output using the detail argument. When specified, the output provides endpoint (port) and policy details for clients in the domain, the secure status of the endpoints, any configuration overrides and constraints, and if the endpoints have a valid configuration. A subject is considered secure if the policies attached to it (either directly or globally) enforce authentication, authorization, or message protection behaviors. Because you can specify the priority of a global or directly attached policy (using the reference.priority configuration override), the effective field indicates if the directly attached policies are in effect for the endpoint.

Note that to simplify endpoint management, all directly attached policies are shown in the output regardless of whether they are in effect. In contrast, only globally attached policies that are in effect for the endpoint are displayed. For more information, see "How the Effective Set of Policies is Calculated" in Oracle Fusion Middleware Security and Administrator's Guide for Web Services.

The output is listed by each application deployed as shown in the following examples:

This example shows the output of an unsecured endpoint:

```
/soa_domain/soa_server1/soa-infra:
    compositeName=default/Basic_SOA_Client[1.0], moduleType=soa,
    serviceRefName=Service1
    Basic_soa_service_pt
    serviceWSDLURI=http://host.example.com:38001/soa-infra/services/default/Basic_SOA_service/Basic_soa_service.wsdl
    oracle.webservices.contentTransferEncoding=base64
    oracle.webservices.charsetEncoding=UTF-8
    oracle.webservices.operationStyleProperty=document
    wsat.flowOption=WSDLDriven
    oracle.webservices.soapVersion=soap1.1
    oracle.webservices.chunkSize=4096
    oracle.webservices.session.maintain=false
    oracle.webservices.preemptiveBasicAuth=false

    oracle.webservices.encodingStyleProperty=http://schemas.xmlsoap.org/soap/encoding/
    oracle.webservices.donotChunk=true
    No attached policies found; endpoint is not secure.
```

This example shows the output for secured endpoints:

```
/soa_domain/soa_server1/AsynchronizedBC_asyncbc:
    moduleName=Asychronized-AsynchronizedBC-context-root, moduleType=web,
    serviceRefName=callback
    owsm.qa.server.serviceinterface.AppModule_asyncServiceImpl/_
    oracleAsyncResponseClient
    Constraint: No Constraint
    (global) security : oracle/wss_username_token_client_policy,
    enabled=true
    /policysets/global/web_callback_add_1 : Module("*")
    Attached policy or policies are valid; endpoint is secure.

/soa_domain/soa_server1/ADF_DC_4:
    moduleName=wsdl, moduleType=wsconn, serviceRefName=TestService
```
TestPort

serviceWSDLURI=http://host.example.com:12345/jaxws-sut-service/TestService?wsdl
security: oracle/wss_username_token_client_policy, enabled=true, effective=false

Constraint: No Constraint
(global) security: oracle/wss11_username_token_with_message_protection_client_policy, enabled=true
/policysets/global/PolicySet-Testport: port('TestPort')
reference.priority=1

Attached policy or policies are valid; endpoint is secure.

/soa_domain/AdminServer/adf_dc_to_bc:
moduleName=ADF_BC, moduleType=wsconn, serviceRefName=AppModuleService
AppModuleServiceSoapHttpPort

Constraint: No Constraint
(global) security: oracle/wss11_username_token_with_message_protection_client_policy, enabled=true
/policysets/global/web_reference_add_1: Domain("soa_domain")

Attached policy or policies are valid; endpoint is secure.

14.2.5.2 Syntax
listWebServiceClients(application, composite, [detail])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application for which you want to list the Web service clients. For example, /domain/server/application#version_number</td>
</tr>
<tr>
<td>composite</td>
<td>Name of the SOA composite for which you want to list the Web service clients. For example, default/HelloWorld[1.0]</td>
</tr>
<tr>
<td>detail</td>
<td>Optional. Specifies whether to list port and policy details for the Web service clients. Valid values are:</td>
</tr>
<tr>
<td>true</td>
<td>Output includes details about the clients, ports, policies, and whether the endpoint is secure or not.</td>
</tr>
<tr>
<td>false</td>
<td>Output lists only the clients. The default is false.</td>
</tr>
</tbody>
</table>

14.2.5.3 Examples
The following example lists information for all Web service clients in the domain.

wls:/wls-domain/serverConfig>listWebServiceClients()

The following example lists the Web service clients for the application jwsclient_1#1.10 for the server soa1 in the domain soainfra.

wls:/wls-domain/serverConfig>listWebServiceClients('soainfra/soa1/jwsclient_1#1.10')
The following example lists the Web service clients for the SOA composite default/HelloWorld[1.0].

```
wlsl:wlsl-domain/serverConfig>listWebServiceClients(None,'default/HelloWorld[1.0']
```

The following example lists details for all of the Web service clients in the domain.

```
wlsl:wlsl-domain/serverConfig>listWebServiceClients(None,None,true)
```

### 14.2.6 `listWebServiceClientPorts`

**Command Category: Web Service and Client Management**

Use with WLST: Online

#### 14.2.6.1 Description

List the Web service port names and the endpoint URLs for Web service clients in an application or SOA composite.

The output will display the name of the Web service client/reference port. For example:

```
AppModuleServiceSoapHttpPort
```

#### 14.2.6.2 Syntax

```
listWebServiceClientPorts(application,moduleOrCompName,moduleType,serviceRefName)
```

**Argument** | **Definition**
--- | ---
`application` | Name and path of the application for which you want to list the Web services port information. For example, `/domain/server/application#version_number`

To list the client port information for an application, this argument is required.

`moduleOrCompName` | Name of the Web module or SOA composite (for example HelloWorld[1.0]) for which you want to list the Web service client port information.

To list the client port information for a SOA composite, the composite name is required (for example `default/HelloWorld[1.0]`), and the `moduleType` argument must be set to `soa`.

`moduleType` | Module type. Valid options are:
- `soa`—SOA composite.
- `web`—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).
- `wls`—Java EE Web services.
- `wsconn`—Use with a connection-based Web service client such as an ADF DC Web service client, ADF JAX-WS Indirection Proxy, or WebCenter client.

`serviceRefName` | Service reference name of the application or SOA composite for which you want to list the Web service client port information.

When the client is an asynchronous Web service callback client, the `serviceRefName` argument must be set to `callback`. 
14.2.6.3 Examples
The following example lists the client ports for the WssUsernameClient Web module in the /soainfra/soa1/jwsclient_1#1.1.0 application. Note that the moduleType is set to wsconn, and the serviceRefName is set to WssUsernameClient.

```
wlsoainfra/serverConfig> listWebServiceClientPorts('/soainfra/soa1/jwsclient_1#1.1.0','WssUsernameClient','wsconn','WssUsernameClient')
```

The following example lists the client ports in the default/HelloWorld[1.0] SOA composite. Note that the moduleType is set to soa, and the serviceRefName is set to client.

```
wlsoainfra/serverConfig> listWebServiceClientPorts(None, 'default/HelloWorld[1.0]','soa','client')
```

14.2.7 listWebServiceClientStubProperties

Note: This command applies to Oracle Infrastructure Web service clients only.

Command Category: Web Service and Client Management
Use with WLST: Online

14.2.7.1 Description
List Web service client port stub properties for an application or SOA composite.

14.2.7.2 Syntax
```
listWebServiceClientStubProperties(application, moduleOrCompName, moduleType, serviceRefName, portInfoName)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application for which you want to list the Web services client port stub properties. For example, /domain/server/application#version_number</td>
</tr>
<tr>
<td></td>
<td>To list the client port stub properties information for an application, this argument is required.</td>
</tr>
<tr>
<td>moduleOrCompName</td>
<td>Name of the Web module or SOA composite (for example HelloWorld[1.0]) for which you want to list the Web services client port stub properties.</td>
</tr>
<tr>
<td></td>
<td>To list the client port stub properties information for a SOA composite, the composite name is required (for example default/HelloWorld[1.0]), and the moduleType argument must be set to soa.</td>
</tr>
<tr>
<td>moduleType</td>
<td>Module type. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>- soa—SOA composite.</td>
</tr>
<tr>
<td></td>
<td>- web—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).</td>
</tr>
<tr>
<td></td>
<td>- wsconn—Use with a connection-based Web service client such as an ADF DC Web service client, ADF JAX-WS Indirection Proxy, or WebCenter client.</td>
</tr>
</tbody>
</table>
### 14.2.7.3 Example

The following example lists the client port stub properties for the `JRFWssUsernamePort` port of the `WssUsernameClient` Web module in the `/soainfra/soa1/jwsclient_1#1.1.0` application. Note that the `moduleType` is set to `wsconn`, and the `serviceRefName` is set to `WssUsernameClient`.

```
<wls:/soainfra/serverConfig>listWebServiceClientStubProperties
  ('/soainfra/soa1/jwsclient_1#1.1.0','WssUsernameClient','wsconn',
   'WssUsernameClient','JRFWssUsernamePort')
```

### 14.2.8 setWebServiceClientStubProperty

**Note:** This command applies to Oracle Infrastructure Web service clients only.

**Command Category:** Web Service and Client Management

**Use with WLST:** Online

#### 14.2.8.1 Description

Set, change, or delete a single stub property of a Web service client port for an application or SOA composite.

#### 14.2.8.2 Syntax

```
setWebServiceClientStubProperty(application, moduleOrCompName, moduleType,
  serviceRefName, portInfoName, propName, [propValue])
```

**Argument** | **Definition**
--- | ---
`application` | Name and path of the application for which you want to set the Web services client port stub property. For example, `/domain/server/application#version_number`

To set a client port stub property for an application, this argument is required.

`moduleOrCompName` | Name of the Web module or SOA composite (for example `HelloWorld[1.0]`) for which you want to set the Web services client port stub property.

To set a client port stub property for a SOA composite, the composite name is required (for example `default/HelloWorld[1.0]`), and the `moduleType` argument must be set to `soa`.

`moduleType` | The type of the module or composite.

`serviceRefName` | Service reference name of the application or SOA composite for which you want to list the Web service client port stub properties.

`portInfoName` | The name of the client port for which you want to list the stub properties.

`propName` | Name of the property you want to set, change, or delete.

`propValue` | Value of the property you want to set or change.
The following example sets the client port stub property `keystore.recipient.alias` to the value `oracle` for the client port `JRFWssUsernamePort`. The port is a client port of the `WssUsernameClient` Web module in the `/soainfra/soa1/jwsclient_1#1.1.0` application. Note that the `moduleType` is set to `wsconn`, and the `serviceRefName` is set to `WssUsernameClient`.

```
wlst:/soainfra/serverConfig> setWebServiceClientStubProperty
('/soainfra/soa1/jwsclient_1#1.1.0','WssUsernameClient','wsconn','WssUsernameClient','JRFWssUsernamePort','keystore.recipient.alias','oracle')
```

### 14.2.9.2 Syntax

```
setWebServiceClientStubProperties(application, moduleOrCompName, moduleType, serviceRefName, portInfoName, properties)
```

**Argument** | **Definition**
--- | ---
`moduleType` | Module type. Valid options are:
  - `soa`—SOA composite.
  - `web`—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).
  - `wsconn`—Use with a connection-based Web service client such as an ADF DC Web service client, ADF JAX-WS Indirection Proxy, or WebCenter client.

`serviceRefName` | Service reference name of the application or SOA composite for which you want to set the Web service client port stub property.

`portInfoName` | The name of the client port for which you want to set the stub property.

`propName` | Stub property name that you want to set, change, or delete. For example, `keystore.recipient.alias`.

`propValue` | Optional. The stub property value, for example `orakey`. To remove the property, specify a blank "" value.

**Note:** This command applies to Oracle Infrastructure Web service clients only.
<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application for which you want to reset the Web services client port stub properties. For example, /domain/server/application#version_number. To configure or reset the client port stub properties for an application, this argument is required.</td>
</tr>
<tr>
<td>moduleOrCompName</td>
<td>Name of the Web module or SOA composite (for example HelloWorld[1.0]) for which you want to reset the Web services client port stub properties. To configure or reset client port stub properties for a SOA composite, the composite name is required (for example default/HelloWorld[1.0]), and the moduleType argument must be set to soa.</td>
</tr>
</tbody>
</table>
| moduleType    | Module type. Valid options are:  
  - soa—SOA composite.  
  - web—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).  
  - wsconn—Use with a connection-based Web service client such as an ADF DC Web service client, ADF JAX-WS Indirection Proxy, or WebCenter client. |
| serviceRefName | Service reference name of the application or SOA composite for which you want to reset the Web service client port stub properties.                                                                                                                                 |
| portInfoName  | The name of the client port for which you want to reset the stub properties.                                                                                                                                     |
| properties    | The list of properties to be set or changed. Properties must be specified using the following format: (“property”, “value”)  
  For example:  
  ```javascript
  [{"keystore.recipient.alias": "oracle"},
   {"csf-key": "oracle"}]
  ```  
  To remove a property or clear the value assigned to it, specify a blank "" value. For example:  
  ```javascript
  [{"csf-key": ""}]
  ```  
  To remove all the properties of the client port, set this argument to None.  
  Sample client port stub properties are as follows:  
  - oracle.webservices.auth.username  
  - oracle.webservices.auth.password  
  - keystore.recipient.alias  
  - csf-key  
  - saml.issuer.name  
  - javax.xml.ws.session.maintain  
  - wsat.Version—SOA references only  
  - wsat.flowOption—SOA references only  
  For a complete list of the configurable properties, see "Configuring the Web Service Client" in Oracle Fusion Middleware Security and Administrator’s Guide for Web Services.
14.2.9.3 Example

The following example resets the client port stub properties `wsat.flowOption` and `wsat.Version` to `SUPPORTS` and `DEFAULT`, respectively. Any other properties that were previously set for this client port are either reset to the default or removed. The client port is `JRFWssUsernamePort` of the `WssUsernameClient` Web module in the `/soainfra/soa1/jwsclient_1#1.1.0` application. Note that the `moduleType` is set to `wsconn`, and the `serviceRefName` is set to `WssUsernameClient`.

```wls:/soainfra/serverConfig> setWebServiceClientStubProperties('/soainfra/soa1/jwsclient_1#1.1.0', 'WssUsernameClient','wsconn','WssUsernameClient','JRFWssUsernamePort', ["wsat.flowOption","SUPPORTS"],("wsat.Version","DEFAULT") )```

14.3 Policy Management Commands

Use the WLST commands listed in Table 14–3 to directly attach Oracle WSM Web service and client policies to application ports or SOA composites.

When you set or change an attached policy in ADF and WebCenter Web service and client applications, you must restart the application for the changes to take effect. After the policy change is completed, a reminder message is displayed prompting you to restart the application. You can stop and restart the application using the standard `stopApplication` and `startApplication` WLST commands. For more information about these commands, see “Deployment Commands” on page 3-19.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>listAvailableWebServicePolicies</td>
<td>Display a list of all the available Oracle Web Services Manager (WSM) policies by category or subject type.</td>
<td>Online</td>
</tr>
<tr>
<td>listWebServicePolicies</td>
<td>List Web service port policy information for a Web service in an application or SOA composite.</td>
<td>Online</td>
</tr>
<tr>
<td>attachWebServicePolicy</td>
<td>Attach a policy to a Web service port of an application or SOA composite.</td>
<td>Online</td>
</tr>
<tr>
<td>attachWebServicePolicies</td>
<td>Attach multiple policies to a Web service port of an application or SOA composite.</td>
<td>Online</td>
</tr>
<tr>
<td>enableWebServicePolicy</td>
<td>Enable or disable a policy attached to a port of a Web service application or SOA composite.</td>
<td>Online</td>
</tr>
<tr>
<td>enableWebServicePolicies</td>
<td>Enable or disable multiple policies attached to a port of a Web service application or SOA composite.</td>
<td>Online</td>
</tr>
<tr>
<td>detachWebServicePolicy</td>
<td>Detach an Oracle WSM policy from a Web service port of an application or SOA composite.</td>
<td>Online</td>
</tr>
<tr>
<td>detachWebServicePolicies</td>
<td>Detach multiple Oracle WSM policies from a Web service port of an application or SOA composite.</td>
<td>Online</td>
</tr>
<tr>
<td>listWebServiceClientPolicies</td>
<td>List Web service client port policies information for an application or SOA composite.</td>
<td>Online</td>
</tr>
</tbody>
</table>
14.3.1 listAvailableWebServicePolicies

Command Category: Policy Management
Use with WLST: Online

14.3.1.1 Description
Display a list of all the available Oracle Web Services Manager (WSM) policies by category or subject type.

14.3.1.2 Syntax
listAvailableWebServicePolicies([category],[subject])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>category</td>
<td>Optional. The policy category, for example: ‘security’, ‘management’.</td>
</tr>
<tr>
<td>subject</td>
<td>Optional. The policy subject type, for example: ‘server’ or ‘client’.</td>
</tr>
</tbody>
</table>

14.3.1.3 Example
The following example lists all the available Oracle WSM server security policies in the domain.

wls:/wls-domain/serverConfig>listAvailableWebServicePolicies('security','server')

14.3.2 listWebServicePolicies

Command Category: Policy Management
Use with WLST: Online
14.3.2.1 Description
List Web service port policy information for a Web service in an application or SOA composite.

The output will display the Web service port name, the OWSM policies it has attached to it, and if applicable, any policy override properties. For example:

HelloWorldPort:
security : oracle/wss_username_token_service_policy , enabled=true

14.3.2.2 Syntax
listWebServicePolicies(application,moduleOrCompName,moduleType,serviceName,subjectName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application for which you want to list the Web services port policy information. For example, /domain/server/application#version_number. To list the port policy information for a Web service application, this argument is required.</td>
</tr>
<tr>
<td>moduleOrCompName</td>
<td>Name of the Web module or SOA composite (for example HelloWorld[1.0]) for which you want to list the Web services port policy information. To list the port policy information for a SOA composite, the composite name is required (for example default/HelloWorld[1.0]), and the moduleType argument must be set to soa.</td>
</tr>
</tbody>
</table>
| moduleType       | Module type. Valid options are:  
|                  | • soa—SOA composite.  
|                  | • web—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).  
|                  | • wls—Java EE Web services. |
| serviceName      | Name of the Web service in the application or SOA composite for which you want to list the port policy information. For example, {http://namespace/}serviceName. Note that the namespace ({http://namespace/}) should not be included for a SOA composite. |
| subjectName      | Policy subject, port, or operation name. |

14.3.2.3 Examples
The following example lists the Web service port policy information for the application HelloWorld#1_0 for the server soa1 in the domain soainfra. In this example, the Web module name is j2wbasicPolicy, the service name is WssUsernameService, and the subject is a port named JRFWssUsernamePort.

wls:/wls-domain/serverConfig>listWebServicePolicies ('/soainfra/soa1/HelloWorld#1_0','j2wbasicPolicy','web', '{http://namespace/}WssUsernameService', 'JRFWssUsernamePort')

The following example lists the port policy information for the SOA composite default/HelloWorld[1.0]. Note that the moduleType is set to SOA, the service name is HelloWorld, and the subject is a port named HelloWorld_pt.

wls:/wls-domain/serverConfig>listWebServicePolicies (None,
14.3.3 attachWebServicePolicy

Command Category: Policy Management

Use with WLST: Online

14.3.3.1 Description

Attach a policy to a Web service port of an application or SOA composite.

The policyURI is validated through the Oracle WSM Policy Manager APIs if the wsm-pm application is installed on WebLogic Server and is available. If the PolicyURI that you specify in this command already is attached or exists, then this command enables the policy if it is disabled.

If the wsm-pm application is not installed or is not available, this command is not executed.

**Note:** Policy changes made using this WLST command are only effective after you restart your application. For ADF and WebCenter applications, a message is displayed to remind you to restart your application.

14.3.3.2 Syntax

```python
attachWebServicePolicy(application, moduleOrCompName, moduleType, serviceName, subjectName, policyURI, [subjectType=None])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
</table>
| `application`     | Name and path of the application to which you want to attach a Web service policy. For example, `/domain/server/application#version_number`  
To attach a policy to a port of a Web service application, this argument is required. |
| `moduleOrCompName`| Name of the Web module or SOA composite (for example HelloWorld[1.0]) to which you want to attach a Web service policy.  
To attach a policy to a port of a SOA composite, the composite name is required (for example default/HelloWorld[1.0]), and the moduleType argument must be set to soa. |
| `moduleType`      | Module type. Valid options are:  
- soa—SOA composite.  
- web—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).  
- wls—Java EE Web services. |
| `serviceName`     | Name of the Web service in the application or SOA composite. For example, `{http://namespace/}serviceName. Note that the namespace ({http://namespace/}) should not be included for a SOA composite. |
| `subjectName`     | Name of the policy subject, port, or operation. |
| `policyURI`       | Oracle WSM policy name URI, for example `oracle/log_policy`. |
The following example attaches the policy oracle/wss_username_token_service_policy to the port JRFWssUsernamePort of the Web module WssUsernameService. The Web service is part of the application HelloWorld#1_0 for the server soa1 in the domain soainfra.

```
wlsl:wlsl-domain/serverConfig>attachWebServicePolicy
']/soainfra/soa1/HelloWorld#1_0','j2wbasicPolicy','web',
'(http://namespace/)WssUsernameService',JRFWssUsernamePort',oracle/wss_username_token_service_policy')
```

The following example attaches the policy oracle/log_policy to the port HelloWorld_pt of the service HelloService in the SOA composite default/HelloWorld[1.0]. Note that the namespace ({http://namespace/}) should not be included for a SOA composite.

```
wls:wlsl-domain/serverConfig>attachWebServicePolicy(None,
'default/HelloWorld[1.0]', 'soa', 'HelloService', 'HelloWorld_pt', oracle/log_policy')
```

The following example attaches the policy oracle/wss_username_token_service_policy to the port helloWorldJaxwsSoapHttpPort of the Java EE Web service helloWorldJaxws.

```
wls:wlsl-domain/serverConfig>attachWebServicePolicy
('/wlsl-domain/AdminServer/helloWorldJaxws','helloWorldJaxws#1!helloWorldJaxws',
'wlsl','helloWorldJaxws', 'helloWorldJaxwsSoapHttpPort', oracle/wss_username_token_service_policy')
```

A Web service cannot contain both a WebLogic Web service policy and an Oracle Web service policy. If you have a Web service with a WebLogic Web service policy, you must first detach it before attaching the Oracle Web service policy. The following example detaches the WebLogic Web service policy Wssp1.2-2007-Saml2.0-SenderVouches-Wss1.1.xml from the port SimplePort in the Java EE Web service SimpleEjbService and attaches the Oracle Web service policy oracle/wss_username_token_service_policy.

```
wls:wlsl-domain/serverConfig>detachWebServicePolicy('/wlsl-domain/AdminServer/SimpleJAXWS','SimpleJAXWS#1!SimpleEjbService', 'wlsl','SimpleEjbService', 
'SimplePort', 'policy:Wssp1.2-2007-Saml2.0-SenderVouches-Wss1.1.xml')
wls:wlsl-domain/serverConfig>attachWebServicePolicy('/wlsl-domain/AdminServer/SimpleJAXWS','SimpleJAXWS#1!SimpleEjbService', 'wlsl','SimpleEjbService', 
'SimplePort', oracle/wss_username_token_service_policy')
```

**Note:** The detachWebServicePolicy WLST command allows you to detach WebLogic Web service policies from a Web service. However, you cannot use the attachWebServicePolicy WLST command to attach WebLogic Web service policies. To attach WebLogic Web service policies to a Web service, you must use the WebLogic Administration Console.
14.3.4 attachWebServicePolicies

Command Category: Policy Management
Use with WLST: Online

14.3.4.1 Description
Attach multiple policies to a Web service port of an application or SOA composite. The policyURIs are validated through the Oracle WSM Policy Manager APIs if the wsm-pm application is installed on WebLogic Server and is available. If any of the policies that you specify in this command are already attached or exist, then this command enables the policies that are already attached (if they are disabled), and attaches the others.

If the wsm-pm application is not installed or is not available, this command is not executed.

**Note:** Policy changes made using this WLST command are only effective after you restart your application. For ADF and WebCenter applications, a message is displayed to remind you to restart your application.

14.3.4.2 Syntax
attachWebServicePolicies(application, moduleOrCompName, moduleType, serviceName, subjectName, policyURIs, [subjectType=None])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application to which you want to attach the Web service policies. For example, /domain/server/application#version_number</td>
</tr>
<tr>
<td>moduleOrCompName</td>
<td>Name of the Web module or SOA composite (for example HelloWorld[1.0]) to which you want to attach Web service policies.</td>
</tr>
<tr>
<td>moduleType</td>
<td>Module type. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>- soa—SOA composite.</td>
</tr>
<tr>
<td></td>
<td>- web—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).</td>
</tr>
<tr>
<td></td>
<td>- wls—Java EE Web services.</td>
</tr>
<tr>
<td>serviceName</td>
<td>Name of the Web service in the application or SOA composite. For example, [<a href="http://namespace/%5DserviceName">http://namespace/]serviceName</a>. Note that the namespace ([<a href="http://namespace/">http://namespace/</a>]) should not be included for a SOA composite.</td>
</tr>
<tr>
<td>subjectName</td>
<td>Name of the policy subject, port, or operation.</td>
</tr>
<tr>
<td>policyURIs</td>
<td>List of Oracle WSM policy name URIs, for example [&quot;oracle/log_policy&quot;,&quot;oracle/wss_username_token_service_policy&quot;]</td>
</tr>
</tbody>
</table>

If any of the policies that you specify are already attached or exist, then this command enables the policies that are already attached (if they are disabled), and attaches the others.
14.3.4.3 Example
The following example attaches the policies "oracle/log_policy", "oracle/wss_username_token_service_policy" to the port JRFWssUsernamePort of the Web module WssUsernameService. The Web service is part of the application HelloWorld#1_0 for the server soainfra in the domain soainfra.

```
    wls:/wls-domain/serverConfig>attachWebServicePolicies
    ('/soainfra/soal/HelloWorld#1_0', 'j2wbasicPolicy', 'web',
    '(http://namespace/)WssUsernameService', 'JRFWssUsernamePort',
    ['"oracle/log_policy", "oracle/wss_username_token_service_policy"'])
```

The following example attaches the policies "oracle/binding_authorization_denyall_policy", "oracle/wss_username_token_service_policy" to the port helloWorldJaxwsSoapHttpPort of the Web module helloWorldJaxws. The Java EE Web service is part of the application helloWorldJaxws for the server AdminServer in the domain wls-domain.

```
    wls:/wls-domain/ServerConfig>attachWebServicePolicies
    ('/wls-domain/AdminServer/helloWorldJaxws', 'helloWorldJaxws#1!helloWorldJaxws',
    'wls', 'helloWorldJaxws', 'helloWorldJaxwsSoapHttpPort',
    ['"oracle/binding_authorization_denyall_policy",
    "oracle/wss_username_token_service_policy"'])
```

14.3.5 enableWebServicePolicy
Command Category: Policy Management
Use with WLST: Online

14.3.5.1 Description
Enable or disable a policy attached to a port of a Web service application or SOA composite.

If the policy that you specify in this command is not attached to the port, an error message is displayed and/or an exception is thrown.

**Note:** Policy changes made using this WLST command are only effective after you restart your application. For ADF and WebCenter applications, a message is displayed to remind you to restart your application.

14.3.5.2 Syntax
```
    enableWebServicePolicy(application, moduleOrCompName, moduleType, serviceName,
    subjectName, policyURI, [enable], [subjectType=None] )
```
## 14.3.5.3 Examples

The following example enables the policy oracle/wss_username_token_service_policy attached to the port JRFWssUsernamePort of the Web module WssUsernameService. The Web service is part of the application HelloWorld#1_0 for the server soainfra in the domain soainfra.

```bash
css:/wls-domain/serverConfig>enableWebServicePolicy
'/soainfra/soa1/HelloWorld#1.0','j2wbasicPolicy','web','
'{http://namespace/}WssUsernameService','JRFWssUsernamePort',"oracle/wss_username

token_service_policy",true)
```

The following example enables the policy oracle/log_policy attached to the port HelloWorld_pt for the service HelloService in the SOA composite default/HelloWorld[1.0]. Note that the namespace ('http://namespace/)) should not be included for a SOA composite.

```bash
css:/wls-domain/serverConfig>enableWebServicePolicy(None,
'default/HelloWorld[1.0]', 'soa','HelloService','HelloWorld_pt','oracle/log_
The following example disables the policy oracle/log_policy attached to the port HelloWorld_pt for the service HelloWorld in the SOA composite default/HelloWorld[1.0]. Note that the namespace ({http://namespace/}) should not be included for a SOA composite.

```
wls:/wls-domain/serverConfig> enableWebServicePolicy(None, 'default/HelloWorld[1.0]', 'soa', 'HelloService', 'HelloWorld_pt', 'oracle/log_policy', false)
```

The following example disables the policy oracle/wss_username_token_service_policy attached to the port helloWorldJaxwsSoapHttpPort for the service helloWorldJaxws in the Java EE Web service wls-domain/AdminServer/helloWorldJaxws

```
```

### 14.3.6 enableWebServicePolicies

**Command Category:** Policy Management

**Use with WLST:** Online

#### 14.3.6.1 Description

Enable or disable multiple policies attached to a port of a Web service application or SOA composite.

If the policyURIs that you specify in this command are not attached to the port, an error message is displayed and/or an exception is thrown.

**Note:** Policy changes made using this WLST command are only effective after you restart your application. For ADF and WebCenter applications, a message is displayed to remind you to restart your application.

#### 14.3.6.2 Syntax

```
enableWebServicePolicies(application, moduleOrCompName, moduleType, serviceName, subjectName, policyURIs, [enable], [subjectType=None])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application for which you want to enable the Web service policies. For example, /domain/server/application#version_number</td>
</tr>
<tr>
<td></td>
<td>To enable policies that are attached to a port of a Web service application, this argument is required.</td>
</tr>
</tbody>
</table>
**14.3.6.3 Example**

The following example enables the policies \["oracle/log_policy", "oracle/wss_username_token_service_policy"\] attached to the port JRFWssUsernamePort of the Web module WssUsernameService. The Web service is part of the application HelloWorld#1_0 for the server soa1 in the domain soainfra.

```
wls:/wls-domain/serverConfig> enableWebServicePolicies
('/soainfra/soa1/HelloWorld#1_0','j2wbasicPolicy','web',
'{http://namespace/}WssUsernameService','JRFWssUsernamePort',
["oracle/log_policy", "oracle/wss_username_token_service_policy"],true)
```

The following example disables the policies \["oracle/binding_authorization_denyall_policy", "oracle/wss_username_token_service_policy"\] attached to the port helloWorldJaxwsSoapHttpPort of the Web module helloWorldJaxws#1!helloWorldJaxws. The Web service is part of the application helloWorldJaxws for the server AdminServer in the domain wls-domain.

```
wls:/wls-domain/serverConfig> enableWebServicePolicies
('/wls-domain/AdminServer/helloWorldJaxws','helloWorldJaxws#1!helloWorldJaxws',
'wls','helloWorldJaxws', 'helloWorldJaxwsSoapHttpPort', ['oracle/binding_authorization_denyall_policy'],false)
```
**14.3.7 detachWebServicePolicy**

Command Category: Policy Management

Use with WLST: Online

**14.3.7.1 Description**

Detach an Oracle WSM policy from a Web service port of an application or SOA composite.

---

**Note:** Policy changes made using this WLST command are only effective after you restart your application. For ADF and WebCenter applications, a message is displayed to remind you to restart your application.

---

**14.3.7.2 Syntax**

`detachWebServicePolicy(application, moduleOrCompName, moduleType, serviceName, subjectName, policyURI, [subjectType=None])`

**Argument** | **Definition**
--- | ---
`application` | Name and path of the application from which you want to detach a Web service policy. For example, `/domain/server/application#version_number` To detach a policy from a port of a Web service application, this argument is required.

`moduleOrCompName` | Name of the Web module or SOA composite (for example `HelloWorld[1.0]`) from which you want to detach a Web service policy. To detach a policy from a port of a SOA composite, the composite name is required (for example `default/HelloWorld[1.0]`), and the `moduleType` argument must be set to `soa`.

`moduleType` | Module type. Valid options are:
- `soa`—SOA composite.
- `web`—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).
- `wls`—Java EE Web services.

`serviceName` | Name of the Web service in the application or SOA composite. For example, `[http://namespace/]serviceName`. Note that the namespace (`[http://namespace/]`) should not be included for a SOA composite.

`subjectName` | Name of the policy subject, port, or operation.

`policyURI` | Oracle WSM policy name URI, for example `'oracle/log_policy'` If the policy specified is not attached, an error message is displayed and/or an exception is thrown.

`subjectType` | Optional. Policy subject type. Valid options are:
- `P`—Port. The default is `P`.
- `O`—Not supported in this release.
14.3.7.3 Examples
The following example detaches the policy oracle/wss_username_token_service_policy from the port JRFWssUsernamePort of the Web module WssUsernameService. The Web service is part of the application HelloWorld#1_0 for the server soa1 in the domain soainfra.

```
wls:/wls-domain/serverConfig> detachWebServicePolicy('/soainfra/soa>HelloWorld#1_0','j2wbasicPolicy','web', '{http://namespace/}WssUsernameService','JRFWssUsernamePort','oracle/wss_username_token_service_policy')
```

The following example detaches the policy oracle/log_policy from the port HelloWorld_pt of the service HelloService in the SOA composite default/HelloWorld[1.0]. Note that the namespace ({http://namespace/}) should not be included for a SOA composite.

```
wls:/wls-domain/serverConfig> detachWebServicePolicy(None, 'default/HelloWorld[1.0]','soa','HelloService','HelloWorld_pt','oracle/log_policy')
```

The following example detaches the policy oracle/wss_username_token_service_policy from the port helloWorldJaxwsSoapHttpPort of the service helloWorldJaxws in the Java EE Web service wls-domain/AdminServer/helloWorldJaxws.

```
```

14.3.8 detachWebServicePolicies
Command Category: Policy Management

Use with WLST: Online

14.3.8.1 Description
Detach multiple Oracle WSM policies from a Web service port of an application or SOA composite.

If the wsm-pm application is not installed or is not available, this command is not executed.

**Note:** Policy changes made using this WLST command are only effective after you restart your application. For ADF and WebCenter applications, a message is displayed to remind you to restart your application.

14.3.8.2 Syntax
```
detachWebServicePolicies(application, moduleOrCompName, moduleType, serviceName, subjectName, policyURIs, [subjectType=None])
```
14.3.8.3 Example

The following example detaches the policies "oracle/log_policy", "oracle/wss_username_token_service_policy" from the port JRFWssUsernamePort of the Web module WssUsernameService. The Web service is part of the application HelloWorld#1_0 for the server soa1 in the domain soainfra.

```bash
wls:/wls-domain/serverConfig> detachWebServicePolicies
('/soainfra/soa1/HelloWorld#1_0','j2wbasicPolicy','web','{http://namespace/}WssUsernameService','JRFWssUsernamePort',
[
"oracle/log_policy",
"oracle/wss_username_token_service_policy"
])
```

The following example detaches the policies "oracle/binding_authorization_denyall_policy", "oracle/wss_username_token_service_policy" from the port helloWorldJaxwsSoapHttpPort of the Java EE Web module helloWorldJaxws. The Web service is part of the application helloWorldJaxws for the server AdminServer in the domain wls-domain.

```bash
wls:/wls-domain/serverConfig> detachWebServicePolicies
('/wls-domain/AdminServer/helloWorldJaxws','helloWorldJaxws#1_helloWorldJaxws',
'wls','helloWorldJaxws', 'helloWorldJaxwsSoapHttpPort',
['oracle/binding_authorization_denyall_policy',
"oracle/wss_username_token_service_policy"])
```
14.3.9 **listWebServiceClientPolicies**

Command Category: Policy Management

Use with WLST: Online

14.3.9.1 **Description**

List Web service client port policies information for an application or SOA composite.

The output will display the Web service client/reference port name and the Oracle WSM policies it has attached to it. For example:

```
test-port:
security: oracle/wss_username_token_client_policy, enabled=true
```

14.3.9.2 **Syntax**

```
listWebServiceClientPolicies(application, moduleOrCompName, moduleType, serviceRefName, portInfoName)
```

14.3.9.3 **Example**

The following example lists the Web service client port policy information for the application `jwsclient_1#1.1.0` for the server `soa1` in the domain `soainfra`. In this example, the Web module name is `WssUsernameClient`, the module type is `wsconn`, the service reference name is `WssUsernameClient`, and the client port name is `JRFWssUsernamePort`.

```
wls:/wls-domain/serverConfig> listWebServiceClientPolicies ('/soainfra/soa1/jwsclient_1#1.1.0', 'WssUsernameClient', 'wsconn', 'WssUsernameClient', 'JRFWssUsernamePort')
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>application</strong></td>
<td>Name and path of the application for which you want to list the Web service client port policy information. For example, <code>/domain/server/application#version_number</code>&lt;br&gt;To list the client port policy information for a Web services application, this argument is required.</td>
</tr>
<tr>
<td><strong>moduleOrCompName</strong></td>
<td>Name of the Web module or SOA composite (for example <code>HelloWorld[1.0]</code>) for which you want to list the Web services port policy information.&lt;br&gt;To list the client port policy information for a SOA composite, the composite name is required (for example <code>default/HelloWorld[1.0]</code>), and the <code>moduleType</code> argument must be set to <code>soa</code>.</td>
</tr>
<tr>
<td><strong>moduleType</strong></td>
<td>Module type. Valid options are:&lt;br&gt;■ soa—SOA composite.&lt;br&gt;■ web—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).&lt;br&gt;■ wls—Java EE Web services.&lt;br&gt;■ wsconn—Use with a connection-based Web service client such as an ADF DC Web service client, ADF JAX-WS Indirection Proxy, or WebCenter client.</td>
</tr>
<tr>
<td><strong>serviceRefName</strong></td>
<td>The service reference name of the application or composite.</td>
</tr>
<tr>
<td><strong>portInfoName</strong></td>
<td>The client port name.</td>
</tr>
</tbody>
</table>
14.3.10 attachWebServiceClientPolicy

Command Category: Policy Management
Use with WLST: Online

14.3.10.1 Description
Attach a Oracle WSM policy to a Web service client port of an application or SOA composite.

The policyURI is validated through the Oracle WSM Policy Manager APIs if the wsm-pm application is installed on WebLogic Server and is available. If the PolicyURI that you specify in this command already is attached or exists, then this command enables the policy if it is disabled.

If the wsm-pm application is not installed or is not available, this command is not executed.

**Note:** Policy changes made using this WLST command are only effective after you restart your application. For ADF and WebCenter applications, a message is displayed to remind you to restart your application.

14.3.10.2 Syntax

attachWebServiceClientPolicy(application, moduleOrCompName, moduleType, serviceRefName, portInfoName, policyURI, [subjectType=None] )

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application for which you want to attach a policy to the Web service client port. For example, /domain/server/application#version_number</td>
</tr>
<tr>
<td></td>
<td>To attach a policy to a client port of a Web service application, this argument is required.</td>
</tr>
<tr>
<td>moduleOrCompName</td>
<td>Name of the Web module or SOA composite (for example HelloWorld[1.0]) for which you want to attach the policy to the client port.</td>
</tr>
<tr>
<td></td>
<td>To attach a policy to a client port of a SOA composite, the composite name is required (for example default/HelloWorld[1.0]), and the moduleType argument must be set to soa.</td>
</tr>
<tr>
<td>moduleType</td>
<td>Module type. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>- soa—SOA composite.</td>
</tr>
<tr>
<td></td>
<td>- web—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).</td>
</tr>
<tr>
<td></td>
<td>- wls—Java EE Web services.</td>
</tr>
<tr>
<td></td>
<td>- wsconn—Use with a connection-based Web service client such as an ADF DC Web service client, ADF JAX-WS Indirection Proxy, or WebCenter client.</td>
</tr>
<tr>
<td>serviceRefName</td>
<td>The service reference name of the application or composite.</td>
</tr>
<tr>
<td>portInfoName</td>
<td>The client port to which you want to attach the Oracle WSM client policy.</td>
</tr>
</tbody>
</table>
14.3.10.3 Examples

The following example attaches the client policy `oracle/wss_username_token_client_policy` to the port `JRFWssUsernamePort` of the Web module `WssUsernameClient`. The Web service is part of the application `jwsclient_1#1.1.0` for the server `soa1` in the domain `soainfra`.

```
<wls>/wls-domain/serverConfig> attachWebServiceClientPolicy
('/soainfra/soa1/jwsclient_1#1.1.0','WssUsernameClient','wsconn',
'WssUsernameClient','JRFWssUsernamePort',"oracle/wss_username_token_client_policy")
```

The following example attaches the client policy `oracle/log_policy` to the client port `HelloWorld_pt` in the SOA composite `default/HelloWorld[1.0]`.

```
<wls>/wls-domain/serverConfig> attachWebServiceClientPolicy
(None, 'default/HelloWorld[1.0]','soa','client','HelloWorld_pt','oracle/log_policy')
```

The following example attaches the `oracle/wss_username_token_client_policy` client policy to the Java EE Web service client port `UpperCaseImplPort` of the Web module `owsm_mbean.resource_pattern.web.ClientJWS/sei2`. The Web service is part of the application `ClientJWS`.

```
wls:/wls-domain/serverConfig> attachWebServiceClientPolicy
('/wls-domain/AdminServer/ClientJWS','owsm_mbean.resource_pattern.web.ClientJWS/sei2',
'wls','owsm_mbean.resource_pattern.web.ClientJWS/sei2', 'UpperCaseImplPort',
"oracle/wss_username_token_client_policy")
```

14.3.11 attachWebServiceClientPolicies

Command Category: Policy Management

Use with WLST: Online

14.3.11.1 Description

Attach multiple policies to a Web service client port of an application or SOA composite.

The policyURIs are validated through the Oracle WSM Policy Manager APIs if the wsm-pm application is installed on WebLogic Server and is available. If the policies that you specify in this command are already attached or exist, then this command enables the policies that are already attached (if they are disabled), and attaches the others.

If the wsm-pm application is not installed or is not available, this command is not executed.
Policy Management Commands

14.3.11.2 Syntax

attachWebServiceClientPolicies(application, moduleOrCompName, moduleType, serviceRefName, portInfoName, policyURIs, [subjectType=None] )

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application for which you want to attach Oracle WSM client policies to the Web service client port. For example, /domain/server/application#version_number.</td>
</tr>
<tr>
<td></td>
<td>To attach policies to a client port of a Web service application, this argument is required.</td>
</tr>
<tr>
<td>moduleOrCompName</td>
<td>Name of the Web module or SOA composite (for example HelloWorld[1.0]) for which you want to attach the policies to the client port.</td>
</tr>
<tr>
<td></td>
<td>To attach policies to a client port of a SOA composite, the composite name is required (for example default/HelloWorld[1.0]), and the moduleType argument must be set to soa.</td>
</tr>
<tr>
<td>moduleType</td>
<td>Module type. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>- soa—SOA composite.</td>
</tr>
<tr>
<td></td>
<td>- web—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).</td>
</tr>
<tr>
<td></td>
<td>- wls—Java EE Web services.</td>
</tr>
<tr>
<td></td>
<td>- wsconn—Use with a connection-based Web service client such as an ADF DC Web service client, ADF JAX-WS Indirection Proxy, or WebCenter client.</td>
</tr>
<tr>
<td>serviceRefName</td>
<td>The service reference name of the application or composite.</td>
</tr>
<tr>
<td>portInfoName</td>
<td>The client port to which you want to attach the Oracle WSM client policy.</td>
</tr>
<tr>
<td>policyURI</td>
<td>The Oracle WSM policy name URIs, for example [&quot;oracle/log_policy&quot;,&quot;oracle/wss_username_token_client_policy&quot;]</td>
</tr>
<tr>
<td></td>
<td>If the policies that you specify in this command are already attached or exist, then this command enables the policies that are already</td>
</tr>
<tr>
<td></td>
<td>attached (if they are disabled), and attaches the others.</td>
</tr>
<tr>
<td>subjectType</td>
<td>Optional. Policy subject type. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>- P—Port. The default is P.</td>
</tr>
<tr>
<td></td>
<td>- 0—Not supported in this release.</td>
</tr>
</tbody>
</table>

14.3.11.3 Examples

The following example attaches the policies oracle/wss_username_token_client_policy and oracle/log_policy to the port JRFWssUsernamePort of the Web module WssUsernameClient. The Web service is part of the application jwsclient_1@1.1.0_ for the server soa in the domain soainfra.

wls://wls-domain/serverConfig> attachWebServiceClientPolicies
('/soainfra/soa/jwsclient_1#1.1.0', 'WssUsernameClient', 'wssconn', 'WssUsernameClient', 'JRFWssUsernamePort', ['oracle/wss_username_token_client_policy', 'oracle/log_policy'], P)
The following example attaches the policies `oracle/wss_username_token_client_policy` and `oracle/log_policy` to the client port `HelloWorld_pt` in the SOA composite `default/HelloWorld[1.0]`.

```
 Côndom wls:/wls-domain/serverConfig>attachWebServiceClientPolicy
 (None, 'default/HelloWorld[1.0]', 'soa', 'client', 'HelloWorld_pt', ['oracle/wss_username_token_client_policy', 'oracle/log_policy'])
```

The following example attaches the policies `oracle/wss10_saml20_token_client_policy` and `oracle/wss11_message_protection_client_policy` to the client port `UpperCaseImplPort` in the Java EE Web module `owsm_mbean.resource_pattern.web.ClientJWS/sei2`.

```
 Côndom wls:/wls-domain/serverConfig>attachWebServiceClientPolicies
 ('/wls-domain/AdminServer/ClientJWS', 'owsm_mbean.resource_pattern.web.ClientJWS/sei2',
  'wls', 'owsm_mbean.resource_pattern.web.ClientJWS/sei2', 'UpperCaseImplPort',
  ['oracle/wss10_saml20_token_client_policy', 'oracle/wss11_message_protection_client_policy'])
```

### 14.3.12 enableWebServiceClientPolicy

**Command Category:** Policy Management

**Use with WLST:** Online

#### 14.3.12.1 Description

Enable or disable a policy of a Web service client port of an application or SOA composite.

---

**Note:** Policy changes made using this WLST command are only effective after you restart your application. For ADF and WebCenter applications, a message is displayed to remind you to restart your application.

---

#### 14.3.12.2 Syntax

```
enableWebServiceClientPolicy(application, moduleName, moduleType,
                          serviceRefName, portInfoName, policyURI, [enable], [subjectType=None])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application for which you want to enable or disable a policy of a Web service client port. For example, /domain/server/application#version_number. To enable or disable a policy of a client port of a Web service application, this argument is required.</td>
</tr>
<tr>
<td>moduleName</td>
<td>Name of the Web module or SOA composite (for example <code>HelloWorld[1.0]</code>) for which you want to enable or disable a policy of a client port.</td>
</tr>
<tr>
<td>moduleType</td>
<td>To enable or disable a policy of a client port for a SOA composite, the composite name is required (for example <code>default/HelloWorld[1.0]</code>), and the moduleType argument must be set to <code>soa</code>.</td>
</tr>
</tbody>
</table>
The following example enables the client policy oracle/wss_username_token_client_policy of the port JRFWssUsernamePort of the Web module WssUsernameClient. The Web service is part of the application jwsclient_1#1.1.0 for the server soa1 in the domain soainfra.

```bash
wls:/wls-domain/serverConfig> enableWebServiceClientPolicy('/soainfra/soa1/jwsclient_1#1.1.0','WssUsernameClient','wsconn','WssUsernameClient','JRFWssUsernamePort', "oracle/wss_username_token_client_policy",true)
```

The following example enables the client policy oracle/log_policy of the client port HelloWorld_pt in the SOA composite default/HelloWorld[1.0].

```bash
wls:/wls-domain/serverConfig> enableWebServiceClientPolicy(None, 'default/HelloWorld[1.0]','soa','client','HelloWorld_pt','oracle/log_policy')
```

The following example disables the client policy oracle/log_policy of the client port HelloWorld_pt in the SOA composite default/HelloWorld[1.0].

```bash
wls:/wls-domain/serverConfig> enableWebServiceClientPolicy(None, 'default/HelloWorld[1.0]','soa','client','HelloWorld_pt','oracle/log_policy',false)
```

The following example disables the client policy oracle/wss_username_token_client_policy on the client port UpperCaseImplPort in the Java EE Web module owsm_mbean.resource_pattern.web.ClientJWS/sei2.

```bash
wls:/wls-domain/serverConfig> enableWebServiceClientPolicy('/wls-domain/AdminServer/ClientJWS',
```
14.3.13 enableWebServiceClientPolicies

Command Category: Policy Management
Use with WLST: Online

14.3.13.1 Description
Enable or disable multiple policies of a Web service client port of an application or SOA composite.

**Note:** Policy changes made using this WLST command are only effective after you restart your application. For ADF and WebCenter applications, a message is displayed to remind you to restart your application.

14.3.13.2 Syntax

```bash
enableWebServiceClientPolicies {application, moduleOrCompName, moduleType, serviceRefName, portInfoName, policyURIs, [enable], [subjectType=None] }
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application for which you want to enable or disable multiple policies of a Web service client port. For example, /domain/server/application#version_number To enable or disable multiple policies of a client port of a Web service application, this argument is required.</td>
</tr>
<tr>
<td>moduleOrCompName</td>
<td>Name of the Web module or SOA composite (for example HelloWorld[1.0]) for which you want to enable or disable multiple policies of a client port. To enable or disable multiple policies of a client port for a SOA composite, the composite name is required (for example default/HelloWorld[1.0]), and the moduleType argument must be set to soa.</td>
</tr>
</tbody>
</table>
| moduleType        | Module type. Valid options are:  
|                   | - soa—SOA composite.  
|                   | - web—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).  
|                   | - wls—Java EE Web services.  
|                   | - wsconn—Use with a connection-based Web service client such as an ADF DC Web service client, ADF JAX-WS Indirection Proxy, or WebCenter client. |
| serviceRefName    | The service reference name of the application or composite.                                                                                                                                                  |
| portInfoName      | The name of the client port to which you want to attach the Oracle WSM client policies.                                                                                                                     |
| policyURIs        | The list of Oracle WSM policy name URIs, for example ["oracle/log_policy","oracle/wss_username_token_client_policy"].                                                                                      |
## 14.3.13.3 Example

The following example enables the client policies `oracle/log_policy` and `oracle/wss_username_token_client_policy` of the port `JRFWssUsernamePort` of the Web module `WssUsernameClient`. The Web service is part of the application `jwsclient_1#1.1.0` for the server `soa1` in the domain `soainfra`.

```java
wls:/wls-domain/serverConfig> enableWebServiceClientPolicies
('soainfra/soa1/jwsclient_1#1.1.0', 'WssUsernameClient', 'wsconn',
'WssUsernameClient', 'JRFWssUsernamePort',
['oracle/log_policy', 'oracle/wss_username_token_client_policy'], true)
```

The following example enables the client policies `oracle/wss10_saml20_token_client_policy` and `oracle/wss11_message_protection_client_policy` of the port `UpperCaseImplPort` of the Java EE Web service module `owsm_mbean.resource_pattern.web.ClientJWS/sei2`.

```java
wls:/wls-domain/serverConfig> enableWebServiceClientPolicies
('/wls-domain/AdminServer/ClientJWS',
'owsm_mbean.resource_pattern.web.ClientJWS/sei2','wls',
'owsm_mbean.resource_pattern.web.ClientJWS/sei2',
'UpperCaseImplPort',
['oracle/wss10_saml20_token_client_policy',
'oracle/wss11_message_protection_client_policy'], true)
```

## 14.3.14 detachWebServiceClientPolicy

**Command Category:** Policy Management

**Use with WLST:** Online

### 14.3.14.1 Description

Detach a policy from a Web service client port of an application or SOA composite.

**Note:** Policy changes made using this WLST command are only effective after you restart your application. For ADF and WebCenter applications, a message is displayed to remind you to restart your application.

### 14.3.14.2 Syntax

```
detachWebServiceClientPolicy(application, moduleOrCompName, moduleType, serviceRefName, portInfoName, policyURI, [subjectType=None])
```
The following example detaches the client policy `oracle/wss_username_token_client_policy` from the port `JRFWssUsernamePort` of the Web module `WssUsernameClient`. The Web service is part of the application `jwsclient_1#1.1.0` for the server `soa1` in the domain `soainfra`.

```
wlst:/wls-domain/serverConfig>detachWebServiceClientPolicy('/soainfra/soa1/jwsclient_1#1.1.0','WssUsernameClient','wsconn','WssUsernameClient','JRFWssUsernamePort','oracle/wss_username_token_client_policy')
```

The following example detaches the client policy `oracle/log_policy` from the client port `HelloWorld_pt` in the SOA composite `default/HelloWorld[1.0]`.

```
wlst:/wls-domain/serverConfig>detachWebServiceClientPolicy(None, 'default/HelloWorld[1.0]','soa','client','HelloWorld_pt','oracle/log_policy')
```

The following command detaches the client policy `oracle/wss_username_token_client_policy` from the client port `UpperCaseImplPort` in the Java EE client module `wsm_mbean.resource_pattern.web.ClientJWS/sei2`.

```
wlst:/wls-domain/serverConfig>detachWebServiceClientPolicy('/domain/server/application#version_number','application','moduleOrCompName','moduleType','serviceRefName','portInfoName','policyURI','subjectType')
```
```python
wls:/wls-domain/serverConfig>
```

### 14.3.15 detachWebServiceClientPolicies

**Command Category:** Policy Management

**Use with WLST:** Online

#### 14.3.15.1 Description

Detach multiple policies from a Web service client port of an application or SOA composite.

**Note:** Policy changes made using this WLST command are only effective after you restart your application. For ADF and WebCenter applications, a message is displayed to remind you to restart your application.

#### 14.3.15.2 Syntax

```python
detachWebServiceClientPolicies(application, moduleOrCompName, moduleType, serviceRefName, portInfoName, policyURIs, [subjectType=None] )
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>application</strong></td>
<td>Name and path of the application for which you want to detach multiple policies from a Web service client port. For example, <code>/domain/server/application#version_number</code> To detach multiple policies from a client port of a Web service application, this argument is required.</td>
</tr>
<tr>
<td><strong>moduleOrCompName</strong></td>
<td>Name of the Web module or SOA composite (for example HelloWorld[1.0]) for which you want to detach multiple policies from a client port. To detach multiple policies from a client port for a SOA composite, the composite name is required (for example default/HelloWorld[1.0]), and the moduleType argument must be set to soa.</td>
</tr>
<tr>
<td><strong>moduleType</strong></td>
<td>Module type. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>- soa—SOA composite.</td>
</tr>
<tr>
<td></td>
<td>- web—Oracle Infrastructure Web Services packaged as a Web module (including an EJB).</td>
</tr>
<tr>
<td></td>
<td>- wls—Java EE Web services.</td>
</tr>
<tr>
<td></td>
<td>- wsconn—Use with a connection-based Web service client such as an ADF DC Web service client, ADF JAX-WS Indirection Proxy, or WebCenter client.</td>
</tr>
<tr>
<td><strong>serviceRefName</strong></td>
<td>The service reference name of the application or composite.</td>
</tr>
<tr>
<td><strong>portInfoName</strong></td>
<td>The client port from which you want to detach the Oracle WSM client policy.</td>
</tr>
</tbody>
</table>
14.3.15.3 Example
The following example detaches the client policies oracle/log_policy and oracle/wss_username_token_client_policy from the port JRFWssUsernamePort of the Web module WssUsernameClient. The Web service is part of the application jwsclient_1#1.1.0 for the server soa1 in the domain soainfra.

```
wlst:/wls-domain/serverConfig>detachWebServiceClientPolicies
('soainfra/soa1/jwsclient_1#1.1.0','WssUsernameClient','wsconn','WssUsernameClient','JRFWssUsernamePort',
["oracle/log_policy","oracle/wss_username_token_client_policy"])
```

The following example detaches the client policies oracle/wss10_saml20_token_client_policy and oracle/wss11_message_protection_client_policy of the port UpperCaseImplPort of the Java EE Web service module owsm_mbean.resource_pattern.web.ClientJWS/sei2.

```
wlst:/wls-domain/serverConfig>detachWebServiceClientPolicies('wlsls-domain/AdminServer/ClientJWS','
'owsm_mbean.resource_pattern.web.ClientJWS/sei2','wlsls-domain/AdminServer/ClientJWS/sei2',
'UpperCaseImplPort','["oracle/wss10_saml20_token_client_policy","oracle/wss11_message_protection_client_policy"])
```

14.3.16 setWebServicePolicyOverride
Command Category: Policy Management
Use with WLST: Online

14.3.16.1 Description
Configure the Web service port policy override properties of an application or SOA composite.

**Note:** This command applies to Oracle Infrastructure Web services only.

14.3.16.2 Syntax
```
setWebServicePolicyOverride(application,moduleOrCompName,moduleType, serviceName, portName,policyURI,properties)
```
### 14.3.16.3 Examples

The following example configures the override properties for the policy `oracle/wss10_message_protection_service_policy` for the port `JRFWssUsernamePort` of the Web module `WssUsernameService`. The Web service is part of the application `HelloWorld#1_0` for the server `soa1` in the domain `soainfra`.

```shell
wls:/wls-domain/serverConfig> setWebServicePolicyOverride
('/soainfra/soa1/HelloWorld#1_0','j2wbasicPolicy',
'web', '{http://namespace/}WssUsernameService', 'JRFWssUsernamePort',
"oracle/wss10_message_protection_service_policy",
[('keystore.sig.csf.key','sigkey')])
```

### 14.4 Policy Set Management Commands

Policy sets enhance the security and manageability of an enterprise by providing a mechanism to globally attach one or more policies to a subject type. Using policy sets, an administrator can specify a default set of policies to be enforced even if none are
directly attached. For detailed information about determining the type and scope of resources to which a policy set can be attached, see "Creating and Managing Policy Sets" in the Oracle Fusion Middleware Security and Administrator’s Guide for Web Services.

All policy set creation, modification, or deletion commands must be performed in the context of a repository session. A repository session can only act on a single document.

---

**Note:** The commands in this section apply to Oracle Infrastructure Web Services only.

To view the help for the WLST commands described in this section, connect to a running instance of the server and enter `help('wsmManage')`.

---

Use the WLST commands listed in Table 14–4 to manage Oracle WSM Web service policy sets for globally attached policies.

**Table 14–4  Web Services WLST Policy Set Management Commands**

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>abortRepositorySession</code></td>
<td>Abort the current Oracle WSM Repository modification session, discarding any changes that were made to the repository during the session.</td>
<td>Online</td>
</tr>
<tr>
<td><code>attachPolicySet</code></td>
<td>Attach a policy set to the specified resource scope.</td>
<td>Online</td>
</tr>
<tr>
<td><code>attachPolicySetPolicy</code></td>
<td>Attach a policy to a policy set using the policy’s URI.</td>
<td>Online</td>
</tr>
<tr>
<td><code>beginRepositorySession</code></td>
<td>Begin a session to modify the Oracle WSM Repository.</td>
<td>Online</td>
</tr>
<tr>
<td><code>clonePolicySet</code></td>
<td>Clone a new policy set from an existing policy set.</td>
<td>Online</td>
</tr>
<tr>
<td><code>commitRepositorySession</code></td>
<td>Write the contents of the current session to the Oracle WSM repository.</td>
<td>Online</td>
</tr>
<tr>
<td><code>createPolicySet</code></td>
<td>Create a new, empty policy set.</td>
<td>Online</td>
</tr>
<tr>
<td><code>deleteAllPolicySets</code></td>
<td>Delete all or selected policy sets from within the Oracle WSM repository.</td>
<td>Online</td>
</tr>
<tr>
<td><code>deletePolicySet</code></td>
<td>Delete a specified policy set.</td>
<td>Online</td>
</tr>
<tr>
<td><code>describeRepositorySession</code></td>
<td>Describe the contents of the current repository session.</td>
<td>Online</td>
</tr>
<tr>
<td><code>detachPolicySetPolicy</code></td>
<td>Detach a policy from a policy set using the policy’s URI.</td>
<td>Online</td>
</tr>
<tr>
<td><code>displayPolicySet</code></td>
<td>Display the configuration of a specified policy set.</td>
<td>Online</td>
</tr>
<tr>
<td><code>enablePolicySet</code></td>
<td>Enable or disable a policy set.</td>
<td>Online</td>
</tr>
<tr>
<td><code>enablePolicySetPolicy</code></td>
<td>Enable or disable a policy attachment for a policy set using the policy’s URI.</td>
<td>Online</td>
</tr>
<tr>
<td><code>listPolicySets</code></td>
<td>List the policy sets in the repository.</td>
<td>Online</td>
</tr>
<tr>
<td><code>migrateAttachments</code></td>
<td>Migrate direct policy attachments to global policy attachments if they are identical.</td>
<td>Online</td>
</tr>
</tbody>
</table>
**Table 14–4 (Cont.) Web Services WLST Policy Set Management Commands**

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>modifyPolicySet</td>
<td>Specify an existing policy set for modification in the current session.</td>
<td>Online</td>
</tr>
<tr>
<td>setPolicySetConstraint</td>
<td>Specify a run-time constraint value for a policy set selected within a session.</td>
<td>Online</td>
</tr>
<tr>
<td>setPolicySetPolicyOverride</td>
<td>Add a configuration override to a policy reference in the current policy set.</td>
<td>Online</td>
</tr>
<tr>
<td>setPolicySetDescription</td>
<td>Specify a description for the policy set selected within a session.</td>
<td>Online</td>
</tr>
<tr>
<td>validatePolicySet</td>
<td>Validate an existing policy set in the repository or in a session.</td>
<td>Online</td>
</tr>
</tbody>
</table>

14.4.1 abortRepositorySession

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management
Use with WLST: Online

14.4.1.1 Description
Abort the current Oracle WSM Repository modification session, discarding any changes that were made to the repository during the session.

14.4.1.2 Syntax
abortRepositorySession()

14.4.1.3 Example
The following example aborts the current Oracle WSM Repository session.

`wls:/wls-domain/serverConfig>abortRepositorySession()`

14.4.2 attachPolicySet

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management
Use with WLST: Online

14.4.2.1 Description
Within a repository session, set an expression that attaches a policy set to the specified resource scope. The expression must define a valid resource scope in a supported format.
Issuing this command outside of a repository session containing a policy set that is being created or modified will result in an error.

### 14.4.2 Syntax

`attachPolicySet(expression)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
</table>
| expression | Expression that attaches the policy set to the specified resource scope.  
For details about specifying the resource scope expression, see "Resource Scope" in Oracle Fusion Middleware Security and Administrator’s Guide for Web Services. |

### 14.4.3 Example

The following example attaches a policy set to the specified `base_domain` resource.

```
> wls:/wls-domain/serverConfig> attachPolicySet('Domain("base_domain")')
```

This example attaches a policy set to the specified `base_domain` and `managed_server` resources.

```
> wls:/wls-domain/serverConfig> attachPolicySet('Domain("base_domain") and Server("managed_server")')
```

### 14.4.3 attachPolicySetPolicy

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management  
Use with WLST: Online

#### 14.4.3.1 Description

Within a repository session, attach a policy, identified by a specified URI, to the current policy set.  
Issuing this command outside of a repository session containing a policy set that is being created or modified will result in an error.

#### 14.4.3.2 Syntax

`attachPolicySetPolicy(uri)`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>uri</td>
<td>URI specifying the policy to attach to the current policy set. For example, 'oracle/log_policy'.</td>
</tr>
</tbody>
</table>

#### 14.4.3.3 Example

The following example attaches the Oracle WSM logging policy to the current policy set.

```
> wls:/wls-domain/serverConfig> attachPolicySetPolicy('oracle/log_policy')
```
14.4.4 beginRepositorySession

Command Category: Policy Set Management
Use with WLST: Online

14.4.4.1 Description
Begin a session to modify the Oracle WSM Repository. A repository session can only act on a single document. An error will be displayed if there is already a current session.

14.4.4.2 Syntax
beginRepositorySession()

14.4.4.3 Example
The following example begins an Oracle WSM Repository modification session.
wls:/wls-domain/serverConfig>\beginRepositorySession()\

14.4.5 clonePolicySet

Command Category: Policy Set Management
Use with WLST: Online

14.4.5.1 Description
Within a repository session, clone a new policy set from an existing policy set. When cloning an existing policy set, all values and attachments in the source policy set are copied into the new policy set, although you can supply a different expression identifying the resource scope. The expression must define a valid resource scope in a supported format.

Issuing this command outside of a repository session will result in an error.

14.4.5.2 Syntax
clonePolicySet(name, source,[attachTo=None],[description=None],[enable='true'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the new policy set clone.</td>
</tr>
<tr>
<td>source</td>
<td>Name of the source policy set that will be cloned.</td>
</tr>
</tbody>
</table>
Example

The first example creates a policy set by cloning the existing `myPolicySet` policy set to create a new `mynewPolicySet` policy set. The second example also creates a policy set, but narrows the resource scope to policy subjects in the specified `soa_server1` server in the domain.

```
wls:/wls-domain/serverConfig> clonePolicySet('myNewPolicySet','myPolicySet')
wls:/wls-domain/serverConfig> clonePolicySet('myNewPolicySet','myPolicySet','Server("soa_server1")')
```

14.4.6 commitRepositorySession

Command Category: Policy Set Management

Use with WLST: Online

14.4.6.1 Description

Write the contents of the current session to the Oracle WSM Repository. Messages are displayed that describe what was committed. An error will be displayed if there is no current session.

14.4.6.2 Syntax

```
commitRepositorySession()
```

14.4.6.3 Example

The following example commits the current repository modification session.

```
wls:/wls-domain/serverConfig> commitRepositorySession()
```

---

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>attachTo=None</code></td>
<td>Optional. Expression that attaches the policy set to the specified resource scope. For details about specifying the resource scope expression, see &quot;Resource Scope&quot; in Oracle Fusion Middleware Security and Administrator's Guide for Web Services. If this argument is set to None, then the expression used in the source policy set to identify the scope of resources is retained.</td>
</tr>
<tr>
<td><code>description=None</code></td>
<td>Optional. Description for the new policy set. If this argument is set to None, then the description used in the source policy set is retained.</td>
</tr>
</tbody>
</table>
| `enable='true'` | Optional. Specifies whether to enable or disable the policy set. Valid options are:  
  - `true`—Enables the policy set. The default is `true`.  
  - `false`—Disables the policy set. If you omit this argument, the policy set is enabled. |

Note: This command applies to Oracle Infrastructure Web services only.
14.4.7 createPolicySet

Note: This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management
Use with WLST: Online

14.4.7.1 Description
Create a new, empty policy set within a repository session. When creating a new policy set, you must specify the type of policy subject that the policy set will apply to, and a supported expression that defines a valid resource scope in a supported format. Issuing this command outside of a repository session will result in an error.

14.4.7.2 Syntax
createPolicySet(name, type, attachTo, [description=None], [enable='true'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the new, empty policy set.</td>
</tr>
<tr>
<td>type</td>
<td>The type of policy subject to which the new policy set applies. The type of policy subject must be one of the following values:</td>
</tr>
<tr>
<td></td>
<td>■ rest-resource—RESTful Resource</td>
</tr>
<tr>
<td></td>
<td>■ sca-component—SOA Component</td>
</tr>
<tr>
<td></td>
<td>■ sca-reference—SOA Reference</td>
</tr>
<tr>
<td></td>
<td>■ sca-service—SOA Service</td>
</tr>
<tr>
<td></td>
<td>■ ws-service—Web Service Endpoint</td>
</tr>
<tr>
<td></td>
<td>■ ws-client—Web Service Client</td>
</tr>
<tr>
<td></td>
<td>■ ws-connection—Web Service Connection</td>
</tr>
<tr>
<td></td>
<td>■ ws-callback—Asynchronous Callback Client</td>
</tr>
<tr>
<td>attachTo</td>
<td>Expression that attaches the policy set to the specified resource scope. For details about specifying the resource scope expression, see &quot;Resource Scope&quot; in Oracle Fusion Middleware Security and Administrator’s Guide for Web Services.</td>
</tr>
<tr>
<td>description</td>
<td>Optional. Description of the new policy set. If no description is specified, then the description for a new policy set will be “Global policy attachments for &lt;type&gt;”, where &lt;type&gt; is the subject type.</td>
</tr>
<tr>
<td>enable</td>
<td>Optional. Specifies whether to enable or disable the new policy set. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>■ true—Enables the new policy set. The default is true.</td>
</tr>
<tr>
<td></td>
<td>■ false—Disables the new policy set.</td>
</tr>
</tbody>
</table>

If you omit this argument, the policy set is enabled.

14.4.7.3 Example
The first example creates a new policy set and specifies the resource scope to only ws-service types (Web Service Endpoint) in the base_domain domain. The second
example creates a new policy set, but also narrows the resource scope to only
sca-service types (SOA Service) in the soa_server1 server in the domain.

wls:/wls-domain/serverConfig>createPolicySet('myPolicySet','ws-service','Domain("base_domain")')

wls:/wls-domain/serverConfig>createPolicySet('myPolicySet','sca-service','Server("soa_server1")','My policySet')

14.4.8 deleteAllPolicySets

Command Category: Policy Set Management
Use with WLST: Online

14.4.8.1 Description
Delete all or selected policy sets from within the Oracle WSM repository. You can
specify whether to force deletion of all the policy sets, or prompt to select individual
policy sets for deletion. If deletion of any policy set fails then this operation throws an
exception and no policy sets are deleted.

14.4.8.2 Syntax
deleteAllPolicySets([mode])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>mode</td>
<td>Optional. The action to be taken for performing policy set deletion. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>■ force—Automatically delete all policy sets without prompting.</td>
</tr>
<tr>
<td></td>
<td>■ prompt—Request user confirmation for each policy set deletion.</td>
</tr>
<tr>
<td></td>
<td>Available options are yes, no, and cancel. If you select cancel for any property set deletion, the operation is canceled and no policy sets are deleted.</td>
</tr>
<tr>
<td></td>
<td>If no mode is specified, this argument defaults to prompt mode.</td>
</tr>
</tbody>
</table>

14.4.8.3 Examples
The following example automatically deletes all policy sets from the repository
without prompting.

wls:/jrfServer_domain/serverConfig> deleteAllPolicySets("force")

Starting Operation deleteAllPolicySets ...

All policy sets were deleted successfully from repository.

deleteAllPolicySets Operation Completed.

The following examples delete selected policy sets from the repository.

wls:/jrfServer_domain/serverConfig> deleteAllPolicySets()

or
deleteAllPolicySets('prompt')

Starting Operation deleteAllPolicySets ... 

Policy Set Name: create_policyset_6
Select 'create_policyset_6' for deletion (yes/no/cancel)? no
Policy Set Name: create_policyset_8
Select 'create_policyset_8' for deletion (yes/no/cancel)? yes
Policy Set Name: create_policyset_21
Select 'create_policyset_21' for deletion (yes/no/cancel)? no
Policy Set Name: create_policyset_10
Select 'create_policyset_10' for deletion (yes/no/cancel)? yes

All the selected policy sets were deleted successfully from repository.

deleteAllPolicySets Operation Completed.

14.4.9 deletePolicySet

Note: This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management
Use with WLST: Online

14.4.9.1 Description
Delete a specified policy set within a repository session. If the session already contains a different policy set, an error will display. If the session already contains the named policy set, then a creation will be undone or a modification will be converted into a deletion.

Issuing this command outside of a repository session will result in an error.

14.4.9.2 Syntax
deletePolicySet(name)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the policy set to be deleted.</td>
</tr>
</tbody>
</table>

14.4.9.3 Example
The following example deletes a specified myPolicySet policy set.

wls:/wls-domain/serverConfig> deletePolicySet('myPolicySet')

14.4.10 describeRepositorySession

Note: This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management
14.4.10.1 Description
Describe the contents of the current session. This will either indicate that the session is empty or list the name of the document that is being updated, along with the type of update (create, modify, or delete). An error will be displayed if there is no current session.

14.4.10.2 Syntax
describeRepositorySession()

14.4.10.3 Example
The following example describes the current repository modification session.

wls:/wls-domain/serverConfig>describeRepositorySession()

14.4.11 detachPolicySetPolicy

Note: This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management
Use with WLST: Online

14.4.11.1 Description
Within a repository session, detach a policy, identified by a specified URI, from the current policy set.
Issuing this command outside of a repository session containing a policy set that is being created or modified will result in an error.

14.4.11.2 Syntax
detachPolicySetPolicy(uri)

14.4.11.3 Example
The following example detaches the Oracle WSM logging policy from the current policy set.

wls:/wls-domain/serverConfig>detachPolicySetPolicy('oracle/log_policy')

14.4.12 displayPolicySet

Note: This command applies to Oracle Infrastructure Web services only.
14.4.12.1 Description
Display the configuration of a specified policy set. If the policy set is being modified in the current session, then that version will be displayed; otherwise, the latest version in the repository will be displayed. An error will display if the policy set does not exist. This command can be issued outside of a repository session.

14.4.12.2 Syntax
displayPolicySet([name])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Optional. Name of the policy set to be displayed. If a name is not specified, the configuration of the policy set, if any, in the current session is displayed or an error message is displayed.</td>
</tr>
</tbody>
</table>

14.4.12.3 Example
The following example displays the configuration of the myPolicySet policy set.

wls:/wls-domain/serverConfig>displayPolicySet('myPolicySet')

14.4.13 enablePolicySet

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management
Use with WLST: Online

14.4.13.1 Description
Enable or disable the current policy set within a repository session. If not specified, this command enables the policy set.

Issuing this command outside of a repository session containing a policy set that is being created or modified will result in an error.

14.4.13.2 Syntax
enablePolicySet([enable=True])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>enable</td>
<td>Optional. Specifies whether to enable or disable the policy set. Valid options are: true—Enables the policy set. The default is true. false—Disables the policy set. If you omit this argument, the policy set is enabled.</td>
</tr>
</tbody>
</table>
14.4.13.3 Example
The following example enables the current policy set.

wls:/wls-domain/serverConfig>enablePolicySet(true)

14.4.14 enablePolicySetPolicy

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management

Use with WLST: Online

14.4.14.1 Description
Within a repository session, enable or disable the policy attachment, which is identified by the provided URI in the current policy set. If not specified, this command enables the policy set. An error displays if the identified policy is not currently attached to the policy set.

Issuing this command outside of a repository session containing a policy set that is being created or modified will result in an error.

14.4.14.2 Syntax

```
enablePolicySetPolicy(uri,[enable=true])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>uri</td>
<td>URI specifying the policy attachment within the policy set.</td>
</tr>
<tr>
<td>enable</td>
<td>Optional. Specifies whether to enable or disable the policy attachment specified by the URI in the policy set. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>■ true—Enables the specified policy attachment in the policy set. The default is true.</td>
</tr>
<tr>
<td></td>
<td>■ false—Disables specified policy attachment in the policy set.</td>
</tr>
<tr>
<td></td>
<td>If you omit this argument, the policy set attachment is enabled.</td>
</tr>
</tbody>
</table>

14.4.14.3 Example
The following example disables the specified logging policy attachment within the current policy set.

wls:/wls-domain/serverConfig>enablePolicySetPolicy('/oracle/log_policy',false)

14.4.15 listPolicySets

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management

Use with WLST: Online
14.4.15.1 Description
Lists the policy sets in the repository. This command will also display a policy set that is being created, modified, or deleted within the current session. You can list all the policy sets or limit the display to include only those that apply to specific policy subject resource types.

14.4.15.2 Syntax
listPolicySets([type=None])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>type=None</td>
<td>Optional. Specifies the type of policy sets to be displayed. The policy subject resource type must be one of the following values:</td>
</tr>
<tr>
<td></td>
<td>■ sca-component—SOA Component</td>
</tr>
<tr>
<td></td>
<td>■ sca-reference—SOA Reference</td>
</tr>
<tr>
<td></td>
<td>■ sca-service—SOA Service</td>
</tr>
<tr>
<td></td>
<td>■ ws-service—Web Service Endpoint</td>
</tr>
<tr>
<td></td>
<td>■ ws-client—Web Service Client</td>
</tr>
<tr>
<td></td>
<td>■ ws-connection—Web Service Connection</td>
</tr>
<tr>
<td></td>
<td>■ ws-callback—Asynchronous Callback Client</td>
</tr>
<tr>
<td></td>
<td>If this argument is set to None, then all the policy sets stored in the repository will be listed.</td>
</tr>
</tbody>
</table>

14.4.15.3 Example
The first two examples list policy sets by either the sca-reference or ws-client resource types. Whereas, the third example lists all the policy sets stored in the repository.

```bash
wls:/wls-domain/serverConfig>listPolicySets('sca-reference')
wls:/wls-domain/serverConfig>listPolicySets('ws-client')
wls:/wls-domain/serverConfig>listPolicySets()
```

14.4.16 migrateAttachments

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management
Use with WLST: Online

14.4.16.1 Description
Migrates direct (local) policy attachments that are identical to the external global policy attachments that would otherwise be attached to each policy subject in the current domain. You can specify whether to force the migration, prompt for confirmation before each migration, or simply list the migrations that would occur. A direct policy attachment is identical if its URI is the same as one provided by a global policy attachment, and if it does not have any scoped configuration overrides.
Whether forced or prompted, the command lists each direct policy attachment that is migrated. This output will identify the policy subject that was modified, the URI of the identical policy reference, and the name of the global policy attachment document that duplicated the direct attachment.

14.4.16.2 Syntax

migrateAttachments([mode])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
</table>
| mode     | The action to be taken for each policy attachment that can be migrated. Valid options are:
|          | ■ force—Automatically migrate all identical policy attachments without prompting. |
|          | ■ preview—List all policy attachments that can be migrated, but does not perform any migration. |
|          | ■ prompt—Request user confirmation before migrating each policy attachment. |
|          | If no mode is specified, this argument defaults to prompt mode. |

14.4.16.3 Example

The following examples describe how to use the repository attachment migration modes.

```
wls:/wls-domain/serverConfig>migrateAttachments()
wls:/wls-domain/serverConfig>migrateAttachments('force')
wls:/wls-domain/serverConfig>migrateAttachments('preview')
wls:/wls-domain/serverConfig>migrateAttachments('prompt')
```

14.4.17 modifyPolicySet

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management
Use with WLST: Online

14.4.17.1 Description

Specify a policy set for modification in the current repository session. The latest version of the named policy set will be loaded into the current session. If the session already contains a different policy set, then an error will be displayed; if the session already contains the named policy set, then no action will be taken. Subsequent attempts to modify the named policy set will show the current version in the session.

Issuing this command outside of a repository session will result in an error.
14.4.17.2 Syntax
modifyPolicySet(name)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name of the policy set to be modified in the current session.</td>
</tr>
</tbody>
</table>

14.4.17.3 Example
The following example opens the myPolicySet policy set for modification in the current session.

wls:/wls-domain/serverConfig>modifyPolicySet('myPolicySet')

14.4.18 setPolicySetConstraint

Note: This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management
Use with WLST: Online

14.4.18.1 Description
Specify a run-time constraint value for a policy set selected within a session. Issuing this command outside of a repository session containing a policy set that is being created or modified will result in an error.


14.4.18.2 Syntax
setPolicySetConstraint(constraint)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>constraint</td>
<td>Expression that specifies the run-time context to which the policy set applies. If not specified, the policy set applies to all run-time contexts.</td>
</tr>
</tbody>
</table>

14.4.18.3 Example
The following example specifies that the policy set apply only to requests from external clients.

setPolicySetConstraint('HTTPHeader("VIRTUAL_HOST_TYPE","external")')

The following example specifies that the policy set apply only to requests from non-external clients.

setPolicySetConstraint('!HTTPHeader("VIRTUAL_HOST_TYPE","external")')
14.4.19 setPolicySetDescription

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management
Use with WLST: Online

14.4.19.1 Description
Specify a description for a policy set selected within a session.
Issuing this command outside of a repository session containing a policy set that is being created or modified will result in an error.

14.4.19.2 Syntax
```
setPolicySetDescription(description)
```

14.4.19.3 Example
The following example creates a description for a policy set.
```
wls:/wls-domain/serverConfig>setPolicySetDescription('PolicySetDescription')
```

14.4.20 setPolicySetPolicyOverride

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: Policy Set Management
Use with WLST: Online

14.4.20.1 Description
Add a configuration override, described by a `name`, `value` pair, to an attached policy reference in the current policy set. The `value` argument is optional. If the `value` argument is omitted, the property specified by the `name` argument is removed from the policy reference in the policy set. If the property specified by the `name` argument already exists and a `value` argument is provided, the current value is overwritten by the new value specified with the `value` argument.
Issuing this command outside of a repository session containing a policy set that is being created or modified results in an error.

14.4.20.2 Syntax
```
setPolicySetPolicyOverride(uri,name,[value=None])
```
14.4.20.3 Example
The following example specifies a configuration override for the reference.priority property for the oracle/wss10_saml_token_service_policy to a value of 1.

setPolicySetPolicyOverride('oracle/wss10_saml_token_service_policy', 'reference.priority','1')

The following example removes the property reference.priority from the oracle/wss10_saml_token_service_policy in the policy set.

setPolicySetPolicyOverride('oracle/wss10_saml_token_service_policy', 'reference.priority')

14.4.21 validatePolicySet

Command Category: Policy Set Management
Use with WLST: Online

14.4.21.1 Description
Validates an existing policy set. If a policy set name is provided, the command will validate the specified policy set. If no policy set name is specified, the command will validate the policy set in the current repository session.

An error message displays if the policy set does not exist, or a name is not provided and the session is not active, or if the Oracle WSM Repository does not contain a suitable policy set.

14.4.21.2 Syntax
validatePolicySet([name=None])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Optional. Name of the policy set to validate. If a name is not provided then the command will validate the policy set being created or modified in the current session.</td>
</tr>
</tbody>
</table>
14.4.21.3 Example
The first example validates the policy set in the current session. The second example validates the specified *myPolicySet* policy set.

```
wls:/wls-domain/serverConfig>validatePolicySet()
wls:/wls-domain/serverConfig>validatePolicySet('myPolicySet')
```

14.5 Oracle WSM Repository Management Commands
Use the commands listed in Table 14–5 to manage the WSM documents stored in the Oracle WSM Repository. For additional information about upgrading or migrating documents in an Oracle WSM Repository, see "Upgrading the Oracle WSM Policies in the Repository" in the *Oracle Fusion Middleware Security and Administrator’s Guide for Web Services*.

Additional MDS WLST commands are described in Chapter 8, "Metadata Services (MDS) Custom WLST Commands."

| Use this command... | To... | Use with WLST...
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>upgradeWSMPolicyRepository</td>
<td>Upgrade the Oracle WSM predefined policies stored in the Oracle WSM Repository with any new predefined policies that are provided in the latest installation of the Oracle Fusion Middleware software.</td>
<td>Online</td>
</tr>
<tr>
<td>resetWSMPolicyRepository</td>
<td>Delete the existing policies stored in the Oracle WSM Repository and refresh it with the latest set of predefined policies that are provided in the new installation of the Oracle Fusion Middleware software.</td>
<td>Online</td>
</tr>
<tr>
<td>exportRepository</td>
<td>Export a set of documents from the repository into a supported ZIP archive. If the specified archive already exists, you can choose whether to overwrite the archive or merge the documents into the existing archive.</td>
<td>Online</td>
</tr>
<tr>
<td>importRepository</td>
<td>Import a set of documents from a supported ZIP archive into the repository. You can provide the location of a file that describes how to map a physical information from the source environment to the target environment.</td>
<td>Online</td>
</tr>
</tbody>
</table>

14.5.1 upgradeWSMPolicyRepository
Command Category: Policy Repository Management
Use with WLST: Online

Note: In 11.1.1.6.0 and later, after you install the latest patch set, the repository is automatically upgraded as part of server startup. You no longer need to execute the *upgradeWSMPolicyRepository* command.
14.5.1.1 Description
Upgrade the Oracle WSM predefined policies stored in the Oracle WSM Repository with any new predefined policies that are provided in the latest installation of the Oracle Fusion Middleware software. If the repository is empty, all of the predefined policies included in the installation are loaded into the repository.

When you execute this command, a message is displayed indicating the policies that have been added to the repository. Any predefined policy that has not been customized for your environment will be replaced. Note, however, that predefined policies that have been customized and user-defined custom policies in the repository are not replaced. To replace these policies also, use the `resetWSMPolicyRepository(true)` command.

The output message also displays a list of any existing predefined policies that Oracle has changed or discontinued in the latest release. If a policy has been discontinued and is no longer supported, Oracle recommends that you remove all references to it and then delete it using Oracle Enterprise Manager.

14.5.1.2 Syntax
```
upgradeWSMPolicyRepository()
```

14.5.1.3 Example
The following example upgrades the existing installation with policies provided in the latest release:

```
wls:/wls-domain/serverConfig>upgradeWSMPolicyRepository()
```

14.5.2 resetWSMPolicyRepository
Command Category: Policy Repository Management
Use with WLST: Online

14.5.2.1 Description
Delete the existing policies stored in the Oracle WSM Repository and refresh it with the latest set of predefined policies that are provided in the new installation of the Oracle Fusion Middleware software. You can use the `clearStore` argument to specify whether to delete all policies, including custom user policies, from the Oracle WSM Repository before loading the new predefined policies.

14.5.2.2 Syntax
```
resetWSMPolicyRepository([clearStore='false'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>clearStore='false'</td>
<td>Policies to be deleted. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>■ true—All policies in the repository, including custom user policies, are deleted.</td>
</tr>
<tr>
<td></td>
<td>■ false—Only the predefined policies supplied by Oracle are deleted. The default is <code>false</code>.</td>
</tr>
</tbody>
</table>

14.5.2.3 Example
The following example deletes all the policies in the repository, including user policies, and adds the predefined policies provided in the current product installation:
Oracle WSM Repository Management Commands

wls:/wls-domain/serverConfig> resetWSMPolicyRepository(true)

14.5.3 exportRepository

Command Category: Policy Repository Management
Use with WLST: Online

14.5.3.1 Description
Export a set of documents from the Oracle WSM Repository into a supported ZIP archive. If the specified archive already exists, the following options are presented:

The specified archive already exists. Update existing archive?
Enter "yes" to merge documents into existing archive, "no" to overwrite, or "cancel" to cancel the operation.

You can also specify a list of the documents to be exported, or use a search expression to find specific documents in the repository.

14.5.3.2 Syntax
exportRepository(archive, [documents=None], [expandReferences='false'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>archive</td>
<td>Name of the archive file. If the specified archive already exists, you can choose whether to overwrite the archive or merge the documents into the existing archive. During override, the original archive is backed up and a message describes the location of the backup archive.</td>
</tr>
<tr>
<td>documents=None</td>
<td>Optional. The documents to be exported to the archive. If no documents are specified, then all assertion templates, intents, policies, and policy sets will be exported. You can specify a list of the documents to be exported, or use a search expression to find specific documents in the repository.</td>
</tr>
<tr>
<td>expandReferences='false'</td>
<td>Optional. Specifies whether the policy references should be expanded during export.</td>
</tr>
</tbody>
</table>

14.5.3.3 Example
The following examples describe repository export sessions. The first example exports all Oracle WSM documents to the policies.zip file.

wls:/wls-domain/serverConfig> exportRepository("/tmp/policies.zip")

This example exports only the sca-component, sca-reference, and sca-service policy sets to the policies.jar file, and also expands the all policy references output during the export process.

wls:/wls-domain/serverConfig> exportRepository("/tmp/policies.jar", ["/policysets/sca_component,"/policysets/sca_reference,"/policysets/sca_service"], true)

This example exports policy sets using wildcards to the some_global_with_noreference_2 file.

wls:/wls-domain/serverConfig> exportRepository('./export/some_global_with_noreference_2', ["policysets:global/web_%", 'policysets:global/web_ref%', 'policysets:global/web_call%'], false)
14.5.4 importRepository

Command Category: Policy Repository Management

Use with WLST: Online

14.5.4.1 Description

Import a set of documents from a supported ZIP archive into the Oracle WSM Repository. You can use the map argument to provide the location of a file that describes how to map physical information from the source environment to the target environment. For example, you can use the map file to ensure that the attachment expression in a policy set document is updated to match the target environment, such as Domain("foo")=Domain("bar")

14.5.4.2 Syntax

importRepository(archive, [map=None], [generateMapFile='false'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>archive</td>
<td>Path to the archive file that contains the list of documents to be imported. If a document being imported is a duplicate of the current version that already exists in the repository, then it will not be imported and a new version of the document is not created.</td>
</tr>
<tr>
<td>map=None</td>
<td>Optional. Location of a sample map file that describes how to map physical information from the source environment to the target environment. You can generate a new map file by setting the generateMapFile argument to true. If you specify a map file without setting the generateMapFile argument to true, and the file does not exist, the operation fails and an error is displayed.</td>
</tr>
<tr>
<td>generateMapFile</td>
<td>Optional. Specify whether to create a sample map file at the location specified by the map argument. No documents are imported when this argument is set to true. The default is false. After the map file is created you can edit it using any text editor. The map file contains the document names given in the archive file and their corresponding attachTo values. The attachTo value can be updated to correspond to the new environment. If a mapping update is not required for a document name, that entry may be either deleted or commented out using the # character. Note: When importing documents into the repository, OWSM validates the attachTo values only. If a value is invalid, then the policy set is disabled. Other text in the map file is not validated.</td>
</tr>
</tbody>
</table>

14.5.4.3 Example

The following examples describe repository import sessions.

The first example imports the contents of the policies.zip file into the repository.

```
importRepository("/tmp/policies.zip")
```

This example uses the generateMapFile argument to generate a map file.

```
importRepository( "./export/some_global_with_noreference_2", map= "./export/some_global_with_noreference_2_map", generateMapFile=true)
```

Here is an example of a generated map file:
Deployment Descriptor Migration Commands

This is an auto generated override file containing the document names given in the archive file and their corresponding attachTo values. The attachTo value can be updated according to the new environment details. If there is no update required for any document name, that entry may be either deleted or commented using the character ("#")

[Resource Scope Mappings]

csa_component_add_1=Composite("*Async*")
csa_reference_add_1=Composite("*Basic_SOA_Client*")
csa_reference_no=Server("*")
csa_service_add_1=Composite("*Basic_SOA_service")
web_callback_add_1=Application("*")
web_client_add_1=Module("*")
web_reference_add_1=Domain("*")
web_service_add_1=Domain("*domain*") and Server("*soa*") and Application("*ADF*")
ws_service_no_1=Server("*Admin*")

This example illustrates how to import documents using a generated map file: /some_global_with_noreference_2_map.

wls:/wls-domain/serverConfig>importRepository('..:/export/export_all', 'export_all_map')

14.6 Deployment Descriptor Migration Commands

Use the commands listed in Table 14–6 to migrate the ADF Business Components and WebCenter services proprietary deployment descriptor (PDD) files between environments, such as from test to production.

For additional information about using these commands, see "Managing Application Migration Between Environments" in the Oracle Fusion Middleware Security and Administrator’s Guide for Web Services.

Note: These commands are deprecated and will be removed in a future release.

| Table 14–6 Deployment Descriptor Migration Commands |
|----------------------------------|----------------------------------|
| **Use this command...** | **To...** | **Use with** |
| exportedJRFWSApplicationPDD | Export an ADF Business Control or WebCenter application deployment descriptor to a Java Archive (JAR) file. | Online |
| importJRFWSApplicationPDD | Import an ADF Business Control or WebCenter Web service application deployment descriptor from the exported JAR file into a new environment, for example, a production environment or a scaled server instance in a cluster. | Online |
| savePddToAllAppInstancesInDomain | Import and save the ADF BC or WebCenter Web service application deployment descriptor from the exported JAR file into all of the server instances in the connected domain. | Online |
14.6.1 exportJRFWSApplicationPDD

Command Category: Deployment descriptor migration
Use with WLST: Online

14.6.1.1 Description
Export an ADF Business Control or WebCenter application deployment descriptor to a Java Archive (JAR) file. If you do not specify a name for the JAR file, the output displays the default name and path to the JAR file.

Note: This command is deprecated and will be removed in a future release.

14.6.1.2 Syntax
exportJRFWSApplicationPDD(application,pddJarFileName=None)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>application</td>
<td>Name and path of the application for which you want to export the configuration information. For example, /domain/server/application#version</td>
</tr>
<tr>
<td>pddJarFileName</td>
<td>Optional. User-specified name for the JAR file. The default is None. For example, /tmp/myPDD.jar</td>
</tr>
</tbody>
</table>

14.6.1.3 Example
The following example exports the Web service PDD for the application ADFBCHelloWorld into a JAR file named exportPDD.jar.

wls:/wls-domain/serverConfig>exportJRFWSApplicationPDD('/wls-domain/ManagedServer/ADFBCHelloWorld','/tmp/exportPDD.jar')
/tmp/exportPDD.jar

14.6.2 importJRFWSApplicationPDD

Command Category: Deployment descriptor migration
Use with WLST: Online

14.6.2.1 Description
Import an ADF Business Control or WebCenter Web service application deployment descriptor from the exported JAR file into a new environment, for example, a production environment or a scaled server instance in a cluster.

Notes: Changes made using this WLST command are only effective after you restart your application. After importing the deployment descriptor, a message is displayed to remind you to restart your application.

This command is deprecated and will be removed in a future release.

14.6.2.2 Syntax
importJRFWSApplicationPDD(application,pddJarFileName)
### 14.6.2.3 Example

The following example imports the Web service application deployment descriptor for the ADFBCHelloWorld application that has been migrated to the server ManagedServer2. The command uses the name of the JAR file that was generated when the `exportJRFWSApplicationPDD` command was executed.

```bash
wls:/wls-domain/serverConfig> importJRFWSApplicationPDD('/wls-domain/ManagedServer2/ADFBCHelloWorld', '/tmp/exportPDD.jar')

application /wls-domain/ManagedServer2/ADFBCHelloWorld PDD has been reset, please restart application now to uptake changes!
```

### 14.6.3 `savePddToAllAppInstancesInDomain`

Command Category: Deployment descriptor migration
Use with WLST: Online

#### 14.6.3.1 Description

Import and save the ADF BC or WebCenter Web service application deployment descriptor from the exported JAR file into all of the server instances in the connected domain. You can also use the optional `restartApp` argument to restart the application automatically.

#### 14.6.3.2 Syntax

```bash
savePddToAllAppInstancesInDomain(applicationName, pddJarFileName, restartApp=true)
```

#### Argument | Definition
--- | ---
`applicationName` | Name of the application to which you want to import the configuration information. For example, `application#version`  
`pddJarFileName` | Name of the JAR file that contains the PDD file to be imported. For example, `/tmp/myPDD.jar`  
`restartApp` | Optional. Restart the application. Valid values are:  
  - `true`—Restart the application automatically. The default is `true`.  
  - `false`—Do not restart the application automatically.

#### Note: This command is deprecated and will be removed in a future release.

#### 14.6.3.3 Example

The following example imports the Web service application deployment descriptor for the ADFBCHelloWorld application that was previously exported into all of the servers in the domain, and restarts the application.
The following example imports the Web service application deployment descriptor for the ADFBCHelloWorld application that was previously exported into all of the servers in the domain, but does not restart the application automatically. This example shows the commands you need to enter to restart the application manually.

```plaintext
wls:/wls-domain/serverConfig> savePddToAllAppInstancesInDomain
'ADFBCHelloWorld', '/tmp/exportPDD.jar', false

saving pdd to  com.bea:ServerRuntime=ManagedServer,Name=ADFBCHelloWorld,
Location=ManagedServer,Type=ApplicationRuntime
saving pdd to  com.bea:ServerRuntime=ManagedServer2,Name=ADFBCHelloWorld,
Location=ManagedServer2,Type=ApplicationRuntime
application  ADFBCHelloWorld  PDD has been reset, please restart application now to uptake changes!

wls:/wls-domain/serverConfig> stopApplication('ADFBCHelloWorld')

wls:/wls-domain/serverConfig> startApplication('ADFBCHelloWorld')
```

### 14.7 Token Issuer Trust Configuration Commands

Use the WLST commands listed in Table 14–7 to view and define trusted issuers, trusted distinguished name (DN) lists, and token attribute rule filters for SAML and JWT signing certificates.

**Note:** The commands in this section apply to Oracle Infrastructure Web Services only.

To view the help for the WLST commands described in this section, connect to a running instance of the server and enter `help('wsmManage')`. 
14.7.1 deleteWSMTokenIssuerTrust

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: Token Issuer Trust Configuration

Use with WLST: Online

14.7.1.1 Description
Delete a trusted token issuer and its associated trusted DN list. Supported values for a SAML assertion or JWT token type are dns.sv, dns.hok, or dns.jwt.

14.7.1.2 Syntax
`deleteWSMTokenIssuerTrust(type, issuer)`

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>The type of SAML assertion or JWT tokens the trusted issuer issues:</td>
</tr>
<tr>
<td></td>
<td>- dns.sv – SAML sender vouches client list</td>
</tr>
<tr>
<td></td>
<td>- dns.hok – SAML HOK or Bearer</td>
</tr>
<tr>
<td></td>
<td>- dns.jwt – JWT token.</td>
</tr>
</tbody>
</table>
14.7.1.3 Examples
In the following example, the issuer www.yourcompany.com and the DN list in the
dns.sv trusted SAML sender vouches client list for the issuer are deleted:

```shell
wls:/wls-domain/serverConfig>deleteWSMTokenIssuerTrust('dns.sv', 'www.yourcompany.com')
```

14.7.2 deleteWSMTokenIssuerTrustAttributeRule

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: Token Issuer Trust Configuration
Use with WLST: Online

14.7.2.1 Description
Delete the token attribute rule associated with a trusted DN.

14.7.2.2 Syntax
`deleteWSMTokenIssuerTrustAttributeRule(dn)`

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dn</td>
<td>The DN of the token signing certificate used as the identifier for the token rule that is to be deleted.</td>
</tr>
</tbody>
</table>

14.7.2.3 Examples
In the following example, the token attribute rule associated with the CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle, C=US' trusted DN is deleted.

```shell
wls:/wls-domain/serverConfig>deleteWSMTokenIssuerTrustAttributeRule('CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle, C=US')
```

14.7.3 displayWSMTokenIssuerTrust

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: Token Issuer Trust Configuration
Use with WLST: Online
14.7.3.1 Description
Display a trusted token issuer and its associated trusted DN list. Supported values for a SAML assertion or JWT token type are dns.hok, dns.sv, or dns.jwt. The issuer argument is optional. If the issuer and type is specified and exists in the trusted issuer list for the type, then the associated DN lists for the issuer is displayed. If issuer is not set, then all trusted issuers of the given type are listed.

14.7.3.2 Syntax
displayWSMTokenIssuerTrust(type, issuer)

Arguments | Definition
--- | ---
type | The type of SAML assertion or JWT tokens the trusted issuer issues:  
- dns.sv – SAML sender vouches client list  
- dns.hok – SAML HOK or Bearer  
- dns.jwt – JWT token.
issuer | Optional. The issuer whose trusted DN list is displayed (for example, SAML assertion or JWT token). If not set, the list of all the trusted issuers is displayed.

14.7.3.3 Examples
In the following example, the DN lists in the dns.sv trusted SAML sender vouches client list for the www.oracle.com trusted issuer are displayed:

wls:/wls-domain/serverConfig>displayWSMTokenIssuerTrust('dns.sv', 'www.oracle.com')

In the following example, the names of all trusted SAML issuers associated with the dns.sv trusted SAML sender vouches client list are displayed:

wls:/wls-domain/serverConfig>displayWSMTokenIssuerTrust('dns.sv', None)

14.7.4 setWSMTokenIssuerTrust

Note: This command applies to Oracle Infrastructure Web services only.

Command Category: Token Issuer Trust Configuration
Use with WLST: Online

14.7.4.1 Description
Configure a trusted token issuer and define trusted keys or a trusted DN list for the issuer. Supported values for a SAML assertion or JWT token type are dns.hok, dns.sv, or dns.jwt. The trustedKeyIDs argument is optional. If you do not set this argument, only the trusted issuer will be set for the specified type.

This command can be used to specify the DN list associated with a trusted token issuer, update the list, or delete the list. See the following examples.

14.7.4.2 Syntax
setWSMTokenIssuerTrust(type, issuer, trustedKeyIDs)
### 14.7.4.3 Examples

In the following example, `CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle, C=US` is set as a DN in the `dns.sv` DN list for the `www.oracle.com` trusted issuer:

```
```

In the following example, the name `CN=orcladmin, OU=Doc, O=Oracle, C=US` in added to the `dns.sv` DN list for the `www.oracle.com` trusted issuer:

```
```

In the following example, the list of DN values in the `dns.sv` DN list is removed from the `www.oracle.com` trusted issuer:

```
wl:wl-domain/serverConfig>setWSMTokenIssuerTrust('dns.sv', 'www.oracle.com', [])
```

### 14.7.5 setWSMTokenIssuerTrustAttributeFilter

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>issuer</td>
<td>The name of the trusted issuer, for example <code>www.oracle.com</code>.</td>
</tr>
<tr>
<td>type</td>
<td>The type of SAML assertion or JWT tokens the trusted issuer issues:</td>
</tr>
<tr>
<td></td>
<td>- <code>dns.sv</code> – SAML sender vouches client list</td>
</tr>
<tr>
<td></td>
<td>- <code>dns.hok</code> – SAML HOK or Bearer</td>
</tr>
<tr>
<td></td>
<td>- <code>dns.jwt</code> – JWT token.</td>
</tr>
<tr>
<td>trustedKeyIDs</td>
<td>Optional. An array of DN’s for token signing certificates associated with the issuer for the specified type. This is a comma-separated list with the format <code>['CN=name1', 'CN=name2', 'CN=name3', ...]</code>. If you enter an empty set <code>[]</code> the list of DN values will be deleted for the issuer.</td>
</tr>
</tbody>
</table>

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: Token Issuer Trust Configuration

Use with WLST: Online

#### 14.7.5.1 Description

Specify token attribute filtering rules for a trusted DN list. For each trusted DN configured for an issuer, a token attribute filtering rule can be configured and applied. Each rule has two parts: a name ID and an attributes part for attributes in a SAML assertion or JWT token. The name ID and each attribute can contain a filter with multiple value patterns.

To remove the list of filters for an attribute for the signing certificate, use an empty set `[]` for the value of `filters`. 

Note: You must first use the `setWSMTokenIssuerTrust` command to configure a list of trusted DN names for an issuer.

### 14.7.5.2 Syntax

```
setWSMTokenIssuerTrustAttributeFilter(dn, attr-name, filters)
```

### Argument | Definition
---|---
**dn** | The DN of a token signing certificate.

| **attr-name** | The name of the user attribute for which the filtering will be applied. The value can be as follows:
| | ■ name-id

| **filters** | List of filters for the attribute. The list has the format `['value1', 'value2', 'value3', ...]`. Each value can be an exact name or a name pattern with a wildcard character "*".
| | When name-id is selected for the attr-name argument, then the value of the subject name ID in the incoming SAML assertion must match one of the specified values to go through. If no values are specified, then any value for the subject name ID will go through.

### 14.7.5.3 Examples

In the following example, the name ID `yourTrustedUser` is set as a trusted user for the `weblogic` trusted DN:

```
> wls:/wls-domain/serverConfig>setWSMTokenIssuerTrustAttributeFilter('CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle, C=US','name-id', ['yourTrustedUser'])
```

Starting Operation `setWSMTokenIssuerTrustAttributeFilter` ...
The token attribute filter are successfully set

In the following example, the name IDs `jdoe` is added to the list of trusted users for the `weblogic` trusted DN:

```
> wls:/wls-domain/serverConfig>setWSMTokenIssuerTrustAttributeFilter('CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle, C=US','name-id', ['yourTrustedUser', 'jdoe'])
```

In the following example, the list of trusted users for the `weblogic` trusted DN is removed:

```
> wls:/wls-domain/serverConfig>setWSMTokenIssuerTrustAttributeFilter('CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle, C=US','name-id', [])
```

### 14.7.6 setWSMTokenIssuerTrustAttributeMapping

Note: This command applies to Oracle Infrastructure Web services only.

Command Category: Token Issuer Trust Configuration
Use with WLST: Online

14.7.6.1 Description
Specify token attribute mapping rules for a trusted DN list. For each trusted DN configured for a token issuer, a token attribute mapping rule can be configured and applied. Each rule has two parts: a name ID and an attributes part for attributes associated with a SAML assertion or JWT token.

For a trusted DN, a token attribute mapping rule sets the mapping for the value of an attribute as specified by the attr-name argument. The user-attribute argument is optional and indicates the local user attribute it corresponds to. The user-mapping-attribute argument is optional and indicates the user attribute to be used in the system to authenticate the users. If the attribute as identified by attr-name exists for a token attribute rule for the DN, the mapping is overwritten by the new value.

For example, in federated environments, where the user subject ID (for example, mail) in the token is different from the user attribute (for example, uid) for authenticating the same user, the name ID and each attribute can map the local user attribute for the subject name ID to the local user attribute to authenticate a trusted user.

---

**Note:** You must first use the setWSMTokenIssuerTrust command to configure a list of trusted DN names for an issuer.

14.7.6.2 Syntax

```
setTokenIssuerTrustAttributeMapping(DN, attr-name, user-attribute=None, user-mapping-attribute=None)
```

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DN</td>
<td>The DN of a token signing certificate.</td>
</tr>
<tr>
<td>attr-name</td>
<td>The name of the use attribute for which the mapping will be applied. The value can be as follows:</td>
</tr>
<tr>
<td></td>
<td>- name-id</td>
</tr>
<tr>
<td>user-attribute</td>
<td>Optional. The local name of the user attribute in the local identity store that the subject name ID corresponds to. The value can be as follows:</td>
</tr>
<tr>
<td></td>
<td>- mail</td>
</tr>
<tr>
<td>user-mapping-attribute</td>
<td>Optional. The value of the local name of the user attribute in the local identity store that the subject name ID maps to for authentication. The value can be as follows:</td>
</tr>
<tr>
<td></td>
<td>- uid</td>
</tr>
</tbody>
</table>

14.7.6.3 Examples

In the following example, the mail attribute for the Subject ID in the token is mapped to the uid attribute.

```
wls:/base_domain/serverConfig>setTokenIssuerTrustAttributeMapping('CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle, C=US', 'name-id', 'mail', 'uid')
```

Starting Operation setWSMTokenIssuerTrustAttributeMapping ...
The token attribute mapping are successfully set
In the following example, the local user attribute for the Subject ID in the token is mapped to the `uid` attribute.

```
wlst:/base_domain/serverConfig>setTokenIssuerTrustAttributeMapping('CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle, C=US', 'name-id', '', 'uid')
```

### 14.7.7 revokeWSMTokenIssuerTrust

**Command Category:** Token Issuer Trust Configuration

**Use with WLST:** Online

#### 14.7.7.1 Description
Revokes trust by removing all trusted issuers and associated configurations (DNs and token attribute rules). The issuers specified in the optional exclude list will not be removed. If no argument is passed, then all trusted issuers and the associated configuration are removed.

#### 14.7.7.2 Syntax

```
revokeWSMTokenIssuerTrust(excludeIssuers=None)
```

#### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>excludeIssuers</td>
<td>Optional. The list of issuers for which trust configuration should not be removed.</td>
</tr>
</tbody>
</table>

#### 14.7.7.3 Examples

In the following example, all trusted issuer configurations are removed except for `www.oracle.com` and `www.yourcompany.com`, which have been excluded:

```
wls:/wls-domain/serverConfig>revokeWSMTokenIssuerTrust(['www.oracle.com','www.yourcompany.com'])
```

*Starting Operation* `revokeWSMTokenIssuerTrust` ...

*Configuration for trusted issuers successfully removed.*

In the following example, all trusted issuer configurations are removed:

```
wls:/wls-domain/serverConfig>revokeWSMTokenIssuerTrust()
```

*Starting Operation* `revokeWSMTokenIssuerTrust` ...

*Configuration for trusted issuers successfully removed.*

### 14.7.8 exportWSMTokenIssuerTrustMetadata

**Note:** This command applies to Oracle Infrastructure Web services only.
Command Category: Token Issuer Trust Configuration

Use with WLST: Online

14.7.8.1 Description

Export all the trust configurations (issuer, DNs, and token attribute rules) for all trusted issuers. The trust configuration will be exported to an XML file identified by the specified location. The trust configuration for the issuers specified in the exclude list will not be exported. If no argument is passed, the trust configuration for all trusted issuers will be exported.

14.7.8.2 Syntax

exportWSMTokenIssuerTrustMetadata(trustFile, excludeIssuers= None)

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>trustFile</td>
<td>The location of the file where the exported metadata will be stored.</td>
</tr>
<tr>
<td>excludeIssuers</td>
<td>Optional. The list of issuers for which trust configuration should not be exported.</td>
</tr>
</tbody>
</table>

14.7.8.3 Examples

In the following example, all trusted issuer configurations are exported to the specified XML file except for www.oracle.com and www.yourcompany.com, which have been excluded:

```
wlst:/wls-domain/serverConfig>exportWSMTokenIssuerTrustMetadata(trustFile='/tmp/trustData.xml', ['www.oracle.com', 'www.yourcompany.com'])
```

Starting Operation exportWSMTokenIssuerTrustMetadata ...
Configuration for trusted issuers successfully exported.

In the following example, all specified trusted issuer configurations are exported to the specified XML file:

```
wlst:/wls-domain/serverConfig>exportWSMTokenIssuerTrustMetadata(trustFile='/tmp/trustData.xml')
```

Starting Operation exportWSMTokenIssuerTrustMetadata ...
Configuration for trusted issuers successfully exported.

14.7.9 importWSMTokenIssuerTrustMetadata

Note: This command applies to Oracle Infrastructure Web services only.

Command Category: Token Issuer Trust Configuration
Use with WLST: Online

14.7.9.1 Description

Import the trust configurations (issuers, DNs, and token attribute rules) for all trusted issuers. The trust configuration will be imported from an XML file identified by the specified location.
14.7.9.2 Syntax

importWSMTokenIssuerTrustMetadata(trustFile)

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>trustFile</td>
<td>The location of the file where the imported metadata will be stored.</td>
</tr>
</tbody>
</table>

14.7.9.3 Examples

In the following example, all trusted issuer configurations are imported from the specified XML file:

```
wls:/wls-domain/serverConfig>importWSMTokenIssuerTrustMetadata(trustFile='/tmp/trustData.xml')
```

Starting Operation importWSMTokenIssuerTrustMetadata ...
Configuration for trusted issuers successfully imported.

14.8 Diagnostic Commands

Use the WLST command in this section to check the status of the WSM components that are required for proper functioning of the product.

14.8.1 checkWSMStatus

Command Category: Diagnostic

Use with WLST: Online

14.8.1.1 Description

Check the status of the WSM components that are required for proper functioning of the product. The WSM components that are checked are the policy manager (wsm-pm), the agent (agent), and the credential store and keystore configuration. The status of the components can be checked together or individually.

**Note:** The Policy Manager (wsm-pm) application must be deployed and running for the check status tool to function correctly.

14.8.1.2 Syntax

```
checkWSMStatus([component=None],[address=None],[verbose=true])
```

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>component</td>
<td>Optional. All checks will be performed if no value is specified. Valid options are:</td>
</tr>
<tr>
<td></td>
<td>■ wsm-pm—Policy Manager. Checks the configuration state of the policy manager component.</td>
</tr>
<tr>
<td></td>
<td>■ agent—Enforcement Agent. Checks status of end-to-end service-side enforcement through the wsm agent component. The enforcement check is specific only to the environment from which the command is run.</td>
</tr>
<tr>
<td></td>
<td>■ credstore—Credential Store. Checks whether the credentials are configured for the keystore password, signing, and encryption certificates in the keystore.</td>
</tr>
</tbody>
</table>
14.8.1.3 Examples

In the following example, the checkWSMStatus command is run without arguments. The status of the credential store, policy manager, and enforcement agent is returned.

wls:/base_domain/serverConfig> checkWSMStatus()

Credential Store Configuration:

PASSED.

Message(s):
  keystore.pass.csf.key : Property is configured and its value is "keystore-csf-key".
    Description: The "keystore.pass.csf.key" property points to the CSF alias that is mapped to the username and password of the keystore. Only the password is used; username is redundant in the case of the keystore.
  keystore-csf-key : Credentials configured.
  keystore.sig.csf.key : Property is configured and its value is "sign-csf-key".
    Description: The "keystore.sig.csf.key" property points to the CSF alias that is mapped to the username and password of the private key that is used for signing.
    sign-csf-key : Credentials configured.
    Sign Key : Key configured.
      Alias - orakey
      Alias - CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle, C=US
      Expiry - June 28, 2020 11:17:12 AM PDT
  keystore.enc.csf.key : Property is configured and its value is "enc-csf-key".
    Description: The "keystore.enc.csf.key" property points to the CSF alias that is mapped to the username and password of the private key that is used for decryption.
    enc-csf-key : Credentials configured.
    Encrypt Key : Key configured.
      Alias - orakey
      Alias - CN=weblogic, OU=Orakey Test Encryption Purposes Only, O=Oracle, C=US
      Expiry - June 28, 2020 11:17:12 AM PDT

Policy Manager:

PASSED.

Message(s):
  OWSM Policy Manager connection state is OK.
  OWSM Policy Manager connection URL is "host.example.com:1234".

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>address</td>
<td>Optional. The HTTP URL of the host running the Policy Manager wsm-pm application. This value is required for checking enforcement through an agent component, for example, checkWSMStatus('agent', '<a href="http://localhost:7001">http://localhost:7001</a>') The address is not required in the WebLogic Server domain where auto-discovery is present.</td>
</tr>
<tr>
<td>verbose</td>
<td>Optional. If the value of this flag is true, then the detailed messages (including stack trace, if any) are displayed. Default is false.</td>
</tr>
</tbody>
</table>
Enforcement Agent:

PASSED.

Message(s):
   Enforcement is successful.
Service URL:
http://host.example.com:7001/Diagnostic/DiagnosticService?wsdl

In the following example, the credential store key keystore-csf-key is deleted and the checkWSMStatus command is rerun for the credential store credstore. The status check fails because the csf-key keystore-csf-key is not present in the credential store:

wls:/base_domain/serverConfig> deleteCred(map="oracle.wsm.security", key="keystore-csf-key")
wls:/base_domain/serverConfig> checkWSMStatus('credstore')

Credential Store Configuration:

FAILED.

Message(s):
   keystore.pass.csf.key : Property is configured and its value is "keystore-csf-key".
      Description: The 'keystore.pass.csf.key' property points to the CSF alias that is mapped to the username and password of the keystore. Only the password is used; username is redundant in the case of the keystore.
   keystore-csf-key : Credentials not configured.

Credential Store Diagnostic Messages:
   Message(s):
      The csf-key keystore-csf-key is not present in the credential store.

Perform the following steps to update the credential store (using WLST commands):
1. connect()
2. createCred(map="oracle.wsm.security", key="keystore-csf-key", user="keystore-csf-key", password="<keystore-password>", desc="Keystore Password CSF Key")

NOTE:- All the above commands are based on the Domain level configurations. The actual csf key may be overridden at runtime due to config override. See Documentation for more details.

In the following example, the csf-key keystore-csf-key is configured and the checkWSMStatus command is rerun. The configuration check passes.

wls:/base_domain/serverConfig> createCred(map="oracle.wsm.security", key="keystore-csf-key", user="keystore-csf-key", password="welcome1", desc="Keystore Password CSF Key")
Already in Domain Runtime Tree

wls:/base_domain/serverConfig> checkWSMStatus('credstore')

Credential Store Configuration:

PASSED.

Message(s):
   keystore.pass.csf.key : Property is configured and its value is "keystore-csf-key".
Description: The "keystore.pass.csf.key" property points to the CSF alias that is mapped to the username and password of the keystore. Only the password is used; username is redundant in the case of the keystore.

Keystore-CSF Key: Credentials configured.
Keystore.sig.csf.key: Property is configured and its value is "sign-csf-key".

Description: The "keystore.sig.csf.key" property points to the CSF alias that is mapped to the username and password of the private key that is used for signing.

Sign CSF Key: Credentials configured.
Sign Key: Key configured.

O=Oracle, C=US
Expiry - June 28, 2020 11:17:12 AM PDT
Keystore.Enc.csf.key: Property is configured and its value is "enc-csf-key".

Description: The "keystore.enc.csf.key" property points to the CSF alias that is mapped to the username and password of the private key that is used for decryption.

Encrypt CSF Key: Credentials configured.
Encrypt Key: Key configured.

O=Oracle, C=US
Expiry - June 28, 2020 11:17:12 AM PDT
true

The following example checks the enforcement status of the agent component at the URL http://localhost:7001.

wls:/test_domain1/serverConfig> checkWSMStatus('agent','http://localhost:7001')

Enforcement Agent:

Note: Enforcement might succeed if OWSM Policy Manager is down due to policy caching. For such scenarios wsm-pm test must be run prior to this test.

PASSED.

Message(s):
Enforcement is successful.
Service URL: http://localhost:7001/Diagnostic/DiagnosticService?wsdl

14.9 JKS Keystore Configuration Commands

Use the WLST commands listed in Table 14–8 to view and manage JKS keystore credentials and certificates.

---

Note: The commands in this section apply to Oracle Infrastructure Web Services only.

To view the help for the WLST commands described in this section, connect to a running instance of the server and enter help('wsmManage').

---
### 14.9.1 deleteWSMKeyStoreEntry

#### Note:
This command applies to Oracle Infrastructure Web services only.

Command Category: JKS Keystore Management

Use with WLST: Online

#### 14.9.1.1 Description
Delete a single `KeyStore.TrustedCertificateEntry` entry from the keystore. You cannot delete the `keyStore.PrivateKeyEntry`.

#### 14.9.1.2 Syntax

```
deleteWSMKeyStoreEntry(alias)
```

#### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alias</td>
<td>Alias of the certificate to be deleted.</td>
</tr>
</tbody>
</table>

#### 14.9.1.3 Examples

In this example, the alias for a key store entry, `testalias1`, is deleted from the keystore.

```
wls:/base_domain/serverConfig> deleteWSMKeyStoreEntry('testalias')
Starting Operation deleteWSMKeyStoreEntry ...
```
Certificate for alias 'testalias' successfully deleted.

14.9.2 deleteWMKeyStoreEntries

---

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: JKS Keystore Management
Use with WLST: Online

**14.9.2.1 Description**
Delete all KeyStore.TrustedCertificateEntry entries from the keystore except those identified by the aliases in the exclusion list. If no argument is passed then all the KeyStore.TrustedCertificateEntry entries will be deleted.

**14.9.2.2 Syntax**
deleteWMKeyStoreEntries(exclusionList=None)

**Arguments**

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>exclusionList</strong></td>
</tr>
<tr>
<td>Optional. List of aliases for the certificate that should not be deleted.</td>
</tr>
</tbody>
</table>

**14.9.2.3 Examples**
In this example, all key store entries are deleted from the keystore, except for the testalias and testalias2 aliases, which are specified on the exclusion list:

```
> deleteWMKeyStoreEntries(['testalias', 'testalias2'])
```

Starting Operation deleteWMKeyStoreEntries ...
Certificate(s) deleted successfully.

In this example, all key store entries are deleted from the keystore:

```
> deleteWMKeyStoreEntries()
```

14.9.3 exportWMCertificate

---

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: JKS Keystore Management
Use with WLST: Online

**14.9.3.1 Description**
Export a trusted certificate or a certificate chain associated with a private key indicated by the specified alias. The certificate will be exported to the specified location.

- If the type argument is Certificate:
If the alias is pointing to KeyStore.TrustedCertificateEntry, it will return
the trusted certificate associated with the entry.

If the alias is pointing to KeyStore.PrivateKeyEntry, it will return the first
certificate in the certificate chain.

If the alias does not point to either KeyStore.TrustedCertificateEntry or
KeyStore.PrivateKeyEntry, it will return an error message.

If the type argument is PKCS7:

- If the alias is pointing to a KeyStore.PrivateKeyEntry, it will return the
certificate chain associated with the entry in PKCS7 format.
- If the alias does not point to KeyStore.PrivateKeyEntry, it will return an
error message.

If the type argument is set to an invalid value, an error message is returned.

14.9.3.2 Syntax

eexportWSMCertificate(alias, certFile, type)

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alias</td>
<td>Alias of the certificate to be exported.</td>
</tr>
<tr>
<td>certFile</td>
<td>Location of the file where the exported certificate will be stored.</td>
</tr>
<tr>
<td>type</td>
<td>Type of keystore entry to be exported. Valid values are: Certificate for exporting KeyStore.TrustedCertificateEntry. PKCS7 for exporting a certificate chain corresponding to a keyStoreKeyStore.PrivateKeyEntry specified by the alias in PKCS7 format.</td>
</tr>
</tbody>
</table>

14.9.3.3 Examples

In this example, the trusted certificate testalias is identified by type as Certificate
and is exported to the specified certificate.cer file:

```
exportWSMCertificate('testalias', '/tmp/certificate.cer', 'Certificate')
```

Starting Operation exportWSMCertificate ...
Certificate for alias 'testalias' successfully exported.

In this example, the certificate chain testalias2 is identified by type as PKCS7 and is
exported to the specified certificatechain.p7b file:

```
exportWSMCertificate('testalias2', '/tmp/certificatechain.p7b', 'PKCS7')
```

14.9.4 importWSMCertificate

**Note:** This command applies to Oracle Infrastructure Web services
only.

Command Category: JKS Keystore Management

Use with WLST: Online
14.9.4.1 Description
Import a trusted certificate or a certificate chain associated with a private key indicated by the specified alias. The Base64 encoded certificate will be imported from the specified location.

14.9.4.2 Syntax
importWSMCertificate(alias, certFile, type, password=None)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alias</td>
<td>Alias of the certificate to be imported.</td>
</tr>
<tr>
<td>certFile</td>
<td>Location of the file from which the Base64 encoded certificate will be imported.</td>
</tr>
<tr>
<td>type</td>
<td>Type of keystore entry to be imported. Valid values are: Certificate for importing KeyStore.TrustedCertificateEntry, PKCS7 for importing a certificate chain corresponding to a keyStoreKeyStore.PrivateKeyEntry specified by the alias in PKCS7 format.</td>
</tr>
<tr>
<td>password</td>
<td>Optional. Password associated with the private key.</td>
</tr>
</tbody>
</table>

14.9.4.3 Examples

In this example, the trusted certificate testalias is identified by type as Certificate and is imported from the specified certificate.cer file:

wls:/base_domain/serverConfig>
importWSMCertificate('testalias','/tmp/certificate.cer','Certificate')

Starting Operation importWSMCertificate ...
Certificate for alias 'testalias' successfully imported.

In this example, the password-protected certificate chain testalias is identified by type as PKCS7 and is imported from the specified certificatechain.p7b file:

wls:/base_domain/serverConfig>
importWSMCertificate('testalias','/tmp/certificatechain.p7b','PKCS7',password='privatekeypassword')

In this example, the certificate chain testalias is identified by type as PKCS7 and is imported from the specified certificatechain.p7b file:

wls:/base_domain/serverConfig>
importWSMCertificate('testalias','/tmp/certificatechain.p7b','PKCS7')

14.9.5 listWSMKeystoreAliases

| Note: This command applies to Oracle Infrastructure Web services only. |

Command Category: JKS Keystore Management
Use with WLST: Online
14.9.5.1 Description
List all the aliases in the keystore.

14.9.5.2 Syntax
listWSMKeystoreAliases()

14.9.5.3 Examples
In this example, all the aliases in the keystore are listed.

```
wls:/base_domain/serverConfig>listWSMKeystoreAliases()

Starting Operation listWSMKeystoreAliases ...

testalias
orakey
testalias2
```

14.9.6 displayWSMCertificate

Displays the string representing the contents of a user's certificate if the alias specifies a KeyStore.TrustedCertificateEntry. Displays the certificates in the chain if the alias points to a certificate chain specified by a KeyStore.PrivateKeyEntry.

---

**Note:** This command applies to Oracle Infrastructure Web services only.

Command Category: JKS Keystore Management

Use with WLST: Online

14.9.6.1 Description

14.9.6.2 Syntax

displayWSMCertificate(alias)

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alias</td>
<td>Alias of the certificate/certificate chain to be displayed.</td>
</tr>
</tbody>
</table>

14.9.6.3 Examples

In this example, the contents of the orakey trusted certificate is displayed.

```
wls:/base_domain/serverConfig>displayWSMCertificate('orakey')

Starting Operation displayWSMCertificate ...
[
  Version: V3
  Subject: CN=OWSM QA, OU=Fusion Middleware, O=Oracle, L=Redwood City, ST=CA, C=US
  Signature Algorithm: SHA1withRSA, OID = 1.2.840.113549.1.1.5

  Key: Sun RSA public key, 1024 bits
  modulus:
```

Note: This command applies to Oracle Infrastructure Web services only.
14.10 OWSM Configuration Commands

Use the WLST commands listed in Table 14–9 to view and configure the OWSM domain.

<table>
<thead>
<tr>
<th>Table 14–9</th>
<th>OWSM Configuration Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Use this command...</strong></td>
<td><strong>To...</strong></td>
</tr>
<tr>
<td>displayConfiguration</td>
<td>Display the full configuration properties and their values and groups for the specified product.</td>
</tr>
<tr>
<td>setConfiguration</td>
<td>Sets the configuration properties of a domain.</td>
</tr>
</tbody>
</table>

14.10.1 displayConfiguration

Command Category: Configuration

Use with WLST: Online

14.10.1.1 Description

Displays the full set of configuration properties, and their values and groups, for the product specified in the context. If a property is not defined in the configuration document associated with the context, then the default value defined for the product is displayed. If a context is not specified, then the set of properties matching the current context is displayed.

14.10.1.2 Syntax

displayConfiguration([context=None])
14.10.1.3 Examples
The following example displays the configuration contained in the configuration document in the repository.

```
wlsl/jrfServer_domain/serverConfig> displayConfiguration()
```

The following example displays the configuration for the `base_domain` domain.

```
wls://jrfServer_domain/serverConfig> displayConfiguration('WLS/base_domain')
```

14.10.2 setConfiguration

Command Category: Configuration

Use with WLST: Online

14.10.2.1 Description
Sets the configuration properties of a domain. The properties are stored in a configuration document for the domain. If a configuration document does not exist, a new one is created.

A new property with values and/or groups of values can be added inside the configuration document. The set of acceptable properties is determined from the default set of properties supported by the product. Specific property values or groups of values can be removed from the configuration document. The configuration document itself is removed if no properties exist in it.

14.10.2.2 Syntax

```
setConfiguration(context, category, name, [group=None], [values=None])
```

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>context</td>
<td>Optional. The context of the configuration document from which property values are displayed. If a context is not specified, then the set of properties matching the current context is displayed. To display the default set of properties along with their values, use &quot;/&quot; as the context value.</td>
</tr>
<tr>
<td>category</td>
<td>The category of the property. This is verified against the default set of properties to ensure it is acceptable for the context. Use the <code>displayConfiguration</code> command to see the category name associated with each property.</td>
</tr>
<tr>
<td>name</td>
<td>The name of the property. This is verified with the default set of properties for acceptability.</td>
</tr>
<tr>
<td>group</td>
<td>Optional. A group containing the set of values to add in a property document. If set to None, then the group will be removed, if such a group exists.</td>
</tr>
<tr>
<td>values</td>
<td>Optional. The array of values to set for a property or group inside the property document.</td>
</tr>
</tbody>
</table>
14.10.2.3 Example
The following example resets the entire configuration for the domain myDomain to its default values.

```
ws:/jrfServer_domain/serverConfig> setConfiguration('/WLS/myDomain')
```

The following command resets the value of the clock.skew property in myDomain to 500.

```
ws:/jrfServer_domain/serverConfig>
setConfiguration('/WLS/myDomain','Agent','clock.skew',None, ['500'])
```

The following command resets the value of the clock.skew property in myDomain to its default value.

```
ws:/jrfServer_domain/serverConfig>
setConfiguration('/WLS/myDomain','Agent','clock.skew',None,None)
```

14.11 Upgrade OWSM Configuration Command
During the patching of an earlier Oracle Fusion Middleware 11g release to Release 11.1.1.9.0, use the WLST command in this section to upgrade the OWSM configuration in a WebLogic Server 11g domain from a previous release (11.1.1.1.0–11.1.1.6.0) to Release 11.1.1.9.0.

The OWSM configuration and policies will be upgraded, such as:

- Any predefined policies that have not been customized for your environment are replaced, and any new policies are automatically added. Note, however, that predefined policies that have been customized and user-created custom policies in the repository are not replaced.
- WS-Trust configuration, such as DNs and trusted issuers.
- Custom roles for secure EJBs.

For more information about patching WebLogic Server 11g domains, see "Post-Patching Tasks for Your WebLogic Server Domain" in the Oracle Fusion Middleware Patching Guide.

14.11.1 upgradeWSM
Command Category: OWSM Configuration Upgrade
Use with WLST: Offline

14.11.1.1 Description
Upgrades the OWSM configuration in a WebLogic Server 11g domain from a previous release (11.1.1.1.0–11.1.1.6.0) to release 11.1.1.9.0. After completing the 11.1.1.9.0 patch installation in your FMW environment, this command must be run from the oracle_common\common\bin directory of release 11.1.1.9.0 installation.

14.11.1.2 Syntax
```
upgradeWSM(domainPath)
```

<table>
<thead>
<tr>
<th>Arguments</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>domainPath</td>
<td>The path of the WebLogic Server 11g domain that is being upgraded.</td>
</tr>
</tbody>
</table>
14.11.1.3 Example

In the following example, the OWSM configuration in a WebLogic Server 11g 11.1.1.5.0 domain is being upgraded to release 11.1.1.9.0.

```
oracle_common\common\bin > upgradeWSM('Oracle_HOME/user_projects/domains/ps5_domain')
```

Starting Operation upgradeWSM ...

Upgrading Oracle Web Services Manager configuration started.

Upgrading Oracle Web Services Manager configuration files started.
Upgrading Oracle Web Services Manager configuration files completed.
Migration of custom roles and policies started.
Migration of custom roles and policies completed.
Oracle Web Services Manager successfully upgraded to version 11.1.1.9.0.

In this example, the upgradeWSM command is being run in a WebLogic Server 11g 11.1.1.5.0 domain that has already been upgraded to release 11.1.1.9.0.

```
wls:/offline>upgradeWSM('Oracle_HOME/user_projects/domains/ps5_domain')
```

Starting Operation upgradeWSM ...

Oracle Web Services Manager is already upgraded to version 11.1.1.9.0. Upgrade is not required.
The Diagnostic Framework aids in capturing relevant and timely diagnostics for critical errors. The diagnostics can be sent to Oracle Support for further analysis. Use the Diagnostic Framework commands to generate incidents, query existing incidents and execute individual diagnostics dumps to gather specific diagnostics data. This chapter provides detailed descriptions of WLST commands for the Diagnostic Framework, including command syntax, arguments and command examples.

For additional information about using the Diagnostic Framework, see "Diagnosing Problems" in the Oracle Fusion Middleware Administrator's Guide.

Table 15–1 lists the different categories of Diagnostic Framework commands.

<table>
<thead>
<tr>
<th>Command Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incident Commands</td>
<td>View problems and incidents and create incidents.</td>
</tr>
<tr>
<td>Diagnostic Dump Commands</td>
<td>Display information about dumps and execute dumps.</td>
</tr>
<tr>
<td>Dump Sampling Commands</td>
<td>Capture samples of diagnostic dumps at specified intervals.</td>
</tr>
</tbody>
</table>

15.1 Incident Commands

Use the commands in Table 15–2 to view problems and incidents and to create incidents.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>createIncident</td>
<td>Create a diagnostic incident.</td>
<td>Online</td>
</tr>
<tr>
<td>getIncidentFile</td>
<td>Retrieves the contents of the specified incident file.</td>
<td>Online</td>
</tr>
<tr>
<td>listADRHomes</td>
<td>List the set of ADR Home paths.</td>
<td>Online</td>
</tr>
</tbody>
</table>
15.1.1 createIncident

Use with WLST: Online

15.1.1.1 Description

Creates a diagnostic incident, using the specified information to determine the set of
diagnostic rules and actions to execute.

15.1.1.2 Syntax


<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adrHome</td>
<td>The path for the ADR Home in which to create the incident. The ADR Home must exist. If this argument is not specified, the default ADR Home is used. The default ADR Home is the following location: ADR_BASE/diag/OFM/domain_name/server_name</td>
</tr>
<tr>
<td>incidentTime</td>
<td>The timestamp at which the incident occurred. If this not specified, the current time is used. You can specify the following: ■ The time of the current day, in the format HH:MM. For example: 19:45 ■ The date and time, in the format MM/DD/YYYY HH:MM</td>
</tr>
<tr>
<td>messageId</td>
<td>The ID of the error message. For example, MDS-50400.</td>
</tr>
<tr>
<td>ecid</td>
<td>The Execution Context ID for the error message.</td>
</tr>
<tr>
<td>appName</td>
<td>The name of the deployed application for which the diagnostics are being gathered. For example, if you have multiple ADF applications deployed, each may register a dump called adf.dump. To execute this command for a specific application, you must specify the application name.</td>
</tr>
<tr>
<td>description</td>
<td>Descriptive text to associate with the incident. This is useful when reviewing the incident at a later time.</td>
</tr>
<tr>
<td>server</td>
<td>The name of the Managed Server from which to collect information. This argument is valid only when you are connected to the Administration Server.</td>
</tr>
</tbody>
</table>

15.1.1.3 Example

The following example creates an incident that is related to messages with the ID MDS-50400:
createIncident(messageId="MDS-50400", description="sample incident")
Incident Id: 55
Problem Id: 4
Problem Key: MDS-50400 [MANUAL]
Incident Time: 25th March 2010 11:55:45 GMT
Error Message Id: MDS-50400
Flood Controlled: false

15.1.2 getIncidentFile
Use with WLST: Online

15.1.2.1 Description
Retrieves the contents of the specified incident file.

15.1.2.2 Syntax
getIncidentFile(id, name [,outputFile] [,adrHome] [,server])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The ID of the incident that you want to retrieve.</td>
</tr>
<tr>
<td>name</td>
<td>The name of the file to retrieve. To find the name of the file, use the</td>
</tr>
<tr>
<td></td>
<td>showIncident command.</td>
</tr>
<tr>
<td>outputFile</td>
<td>The name of the file to which to write the output.</td>
</tr>
<tr>
<td>adrHome</td>
<td>The path for the ADR Home from which to retrieve the information. If this</td>
</tr>
<tr>
<td></td>
<td>argument is not specified, the default ADR Home will be queried.</td>
</tr>
<tr>
<td></td>
<td>The default ADR Home is the following location:</td>
</tr>
<tr>
<td></td>
<td>ADR_BASE/diag/OFM/domain_name/server_name</td>
</tr>
<tr>
<td>server</td>
<td>The name of the Managed Server from which to collect information. This</td>
</tr>
<tr>
<td></td>
<td>argument is valid only when you are connected to the Administration Server.</td>
</tr>
</tbody>
</table>

15.1.2.3 Example
The following example writes the contents of the incident dms_metrics3_i1.dmp to the specified output file:

getIncidentFile(id='1', name='dms_metrics3_i1.dmp', outputFile='/tmp/incident1_dms.txt')
The content of 'dms_metrics3_i1.dmp'is written to /tmp/incident1_dms.txt

15.1.3 listADRHome
Use with WLST: Online

15.1.3.1 Description
Lists the paths of all of the ADR Homes for the server.

15.1.3.2 Syntax
listADRHome([server])
15.1.3.3 Example
The following example lists the paths of the ADR homes:

```
listADRHomes()
diag/ofm/base_domain/WLS_Spaces
diag/ofm/fusionapps/GeneralLedger
```

15.1.4 listIncidents
Use with WLST: Online

15.1.4.1 Description
Lists the set of diagnostic incidents for the given problem ID, if specified, or all available incidents.

15.1.4.2 Syntax
```
listIncidents([id] [, adrHome] [,server])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The ID of the problem for which you want to list the set of diagnostic incidents.</td>
</tr>
<tr>
<td>adoHome</td>
<td>The path for the ADR Home from which to query incidents. If this argument is not specified, the default ADR Home will be queried.</td>
</tr>
<tr>
<td>server</td>
<td>The name of the Managed Server from which to collect information. This argument is valid only when you are connected to the Administration Server.</td>
</tr>
</tbody>
</table>

15.1.4.3 Example
The following example lists the incidents associated with the problem with the ID 1:

```
listIncidents(id="1")
```

<table>
<thead>
<tr>
<th>Incident Id</th>
<th>Problem Key</th>
<th>Incident Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>MDS-50300 [WLS_Spaces] [oracle.mds.repos]</td>
<td>Mon Mar 15 11:22:12 PDT</td>
</tr>
<tr>
<td>2010</td>
<td>MDS-50300 [WLS_Spaces] [oracle.mds.repos]</td>
<td>Thu Mar 11 15:11:35 PDT</td>
</tr>
</tbody>
</table>

15.1.5 listProblems
Use with WLST: Online

15.1.5.1 Description
Lists the set of diagnostic problems associated with the specified ADR Home.
15.1.5.2 Syntax

```
listProblems([adrHome],[server])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>adrHome</td>
<td>The path for the ADR Home from which to query problems. If this argument is not specified, the default ADR Home will be queried. The default ADR Home is the following location: ADR_BASE/diag/OFM/domain_name/server_name</td>
</tr>
<tr>
<td>server</td>
<td>The name of the Managed Server from which to collect information. This argument is valid only when you are connected to the Administration Server.</td>
</tr>
</tbody>
</table>

15.1.5.3 Example

The following example lists the diagnostic problems in the default ADR home:

```
listProblems()
```

<table>
<thead>
<tr>
<th>Problem Id</th>
<th>Problem Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MDS-50300 [WLS_Spaces] [oracle.mds.repos]</td>
</tr>
<tr>
<td>2</td>
<td>JOC-38922 [AdminServer] [oracle.cache.network]</td>
</tr>
</tbody>
</table>

15.1.6 reloadCustomRules

Use with WLST: Online, Offline

15.1.6.1 Description

Reloads all custom diagnostic rules or the specified custom diagnostic rule.

15.1.6.2 Syntax

```
reloadCustomRules([name] [, server])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of a custom diagnostic rule. This argument is optional. If you specify it, only the named rule is reloaded. If you do not specify this argument, all custom diagnostic rules are reloaded. The file containing the custom diagnostic rule must be located in one of the following directories: DOMAIN_HOME/config/fmwconfig/dfw DOMAIN_HOME/config/fmwconfig/servers/server_name/dfw</td>
</tr>
<tr>
<td>server</td>
<td>The name of the server to which to reload the rules. This argument is optional. If you do not specify it, the rules are reloaded to all servers. This option is only valid when you are connected to the Administration Server.</td>
</tr>
</tbody>
</table>

15.1.6.3 Example

The following example reloads the custom diagnostic rule myCustomRules.xml:

```
reloadCustomRules(name='myCustomRules.xml')
```

15.1.7 showIncident

Use with WLST: Online
15.1.7.1 Description
Shows the details of the specified incident.

15.1.7.2 Syntax
showIncident(id, [adrHome][, server])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The ID of the incident that you want to view.</td>
</tr>
<tr>
<td>adrHome</td>
<td>The path for the ADR Home from which to query the incident. If this argument is not specified, the default ADR Home will be queried. The default ADR Home is the following location: ADR_BASE/diag/OFM/domain_name/server_name</td>
</tr>
<tr>
<td>server</td>
<td>The name of the Managed Server from which to collect information. This argument is valid only when you are connected to the Administration Server.</td>
</tr>
</tbody>
</table>

15.1.7.3 Example
The following example displays information about the incident with the ID 10:

```
showIncident(id="10")
```

Incident Id: 10
Problem Id: 1
Problem Key: MDS-50300 [WLS_Spaces] [oracle.mds.repos]
Incident Time: 25th March 2010 10:12:15 GMT
Error Message Id: MDS-50300
Execution Context: 0000ICK4rbYC8xT6uBf9EH1AX1qF000000
Flood Controlled: false
Dump Files:
- dms_ecidctx1_i1.dmp
- jvm_threads2_i1.dmp
- dms_metrics3_i1.dmp
- odl_logs4_i1.dmp
- diagnostic_image_AdminServer_2010_03_25_11_12_15.zip
- readme.txt

15.2 Diagnostic Dump Commands
Use the commands in Table 15–3 to display information about dumps and to execute dumps.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>describeDump</td>
<td>Display a description of the specified diagnostic dump.</td>
<td>Online</td>
</tr>
<tr>
<td>executeDump</td>
<td>Execute the specified diagnostic dump.</td>
<td>Online</td>
</tr>
<tr>
<td>listDumps</td>
<td>Display the set of diagnostic dumps that can be executed.</td>
<td>Online</td>
</tr>
</tbody>
</table>

15.2.1 describeDump
Use with WLST: Online
### 15.2.1 Description
Displays a description of the specified diagnostic dump.

### 15.2.2 Syntax
`describeDump(name [,appName] [.server])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the dump for which to display information.</td>
</tr>
<tr>
<td>appName</td>
<td>The name of the deployed application for which information is gathered.</td>
</tr>
<tr>
<td>server</td>
<td>The name of the Managed Server from which to collect information.</td>
</tr>
</tbody>
</table>

For example, if you have multiple ADF applications deployed, each may register a dump called adf.dump. To execute this command for a specific application, you must specify the application name.

**Example**
The following example displays information about the dump with the name odl.logs. You use the `listDumps` command to retrieve the list of available dumps.

```
describeDump(name="odl.logs")
```

**Name:** odl.logs

**Description:** Dumps recent ODL logs, or logs correlated by ECID

**Manadatory Arguments:**

**Optional Arguments:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECID</td>
<td>String</td>
<td>Execution Context Id to correlate log entries with</td>
</tr>
<tr>
<td>timestamp</td>
<td>String</td>
<td>Timestamp to query logs 5 minutes before/after</td>
</tr>
</tbody>
</table>

### 15.2.2 executeDump
Use with WLST: Online

### 15.2.2.1 Description
Executes the specified diagnostic dump.

### 15.2.2.2 Syntax
`executeDump(name [,args] [,outputFile] [,id] [,adrHome] [,server])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>The name of the diagnostic dump to execute.</td>
</tr>
<tr>
<td>args</td>
<td>Mandatory or optional arguments to pass to the dump.</td>
</tr>
<tr>
<td>outputFile</td>
<td>The name of the file to which to write the dump. If you do not specify this argument, the output is written to the console.</td>
</tr>
<tr>
<td>id</td>
<td>The ID of the incident to which to associate the dump. By default, the specified dump will not be associated with an incident.</td>
</tr>
<tr>
<td>adrHome</td>
<td>The ADR home that contains the incident. If you do not specify this argument, the default ADR home is used. The default ADR Home is the following location: ADR_BASE/diag/OPM/domain_name/server_name</td>
</tr>
</tbody>
</table>
Arguments that are either required or are optional can be specified using the "args" keyword. For example:

executeDump("java.sysprops", args={"prop" : "os.name"})

### 15.2.2.3 Examples

The following example executes the dump with the name jvm.threads and writes it to the file dumpout.txt:

```
executeDump(name="jvm.threads", outputFile="/tmp/dumpout.txt")
```

Diagnostic dump jvm.threads output written to /tmp/dumpoutput.txt

The following example executes the dump with the name jvm.threads and the Incident ID for 33 and writes it to the file dumpout.txt:

```
executeDump(name="jvm.threads", outputFile="/tmp/dumpout.txt", id="33")
```

Diagnostic dump jvm.threads output associated with incident 33 in ADR Home diag/ofm/base_domain/AdminServer

The following example executes a dump with the argument prop set to the value os.name:

```
executeDump(name="java.sysprops", args={"prop" : "os.name"})
```

### 15.2.3 listDumps

Use with WLST: Online

#### 15.2.3.1 Description

Displays the set of diagnostic dumps that can be executed.

#### 15.2.3.2 Syntax

```
listDumps([appName] [,server])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>The name of a deployed application for which diagnostics are being gathered. For example, if you have multiple ADF applications deployed, each may register a dump called adf.dump. To execute this command for a specific application, you must specify the application name. If you specify this argument, the command returns the dumps for the specified application. If you do not specify this argument, the command returns the system dumps.</td>
</tr>
<tr>
<td>server</td>
<td>The name of the Managed Server from which to collect information. This argument is valid only when you are connected to the Administration Server.</td>
</tr>
</tbody>
</table>

### 15.2.3.3 Example

The following example lists all of the available dumps.
Dump Sampling Commands

Use the command describeDump(name=<dumpName>) for help on a specific dump.

15.3 Dump Sampling Commands

Use the commands in Table 15–4 to capture samples of diagnostic dumps at specified intervals.

<table>
<thead>
<tr>
<th>Table 15–4 Dump Sampling Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use this command...</td>
</tr>
<tr>
<td>addDumpSample</td>
</tr>
<tr>
<td>enableDumpSampling</td>
</tr>
<tr>
<td>getSamplingArchives</td>
</tr>
<tr>
<td>isDumpSamplingEnabled</td>
</tr>
<tr>
<td>listDumpSamples</td>
</tr>
<tr>
<td>removeDumpSample</td>
</tr>
<tr>
<td>updateDumpSample</td>
</tr>
</tbody>
</table>

15.3.1 addDumpSample

Use with WLST: Online

15.3.1.1 Description

Creates dump samplings for Diagnostic Framework dumps.

15.3.1.2 Syntax

addDumpSample(sampleName, diagnosticDumpName [, appName], samplingInterval, rotationCount [, dumpedImplicitly] [, toAppend] [, args] [, server=])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>sampleName</td>
<td>The name of the sampling.</td>
</tr>
<tr>
<td>diagnosticDumpName</td>
<td>The name of the diagnostic dump to be sampled.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. The name of the application associated with the specified diagnostic dump. If you do not specify appName, the diagnostic dump has a scope of system.</td>
</tr>
<tr>
<td>samplingInterval</td>
<td>The sampling interval in seconds. If you specify zero or a negative value, sampling is suspended.</td>
</tr>
</tbody>
</table>
## 15.3.1.3 Example

The following example adds a sampling for the dump dms.metrics:

```python
addDumpSample(sampleName='dms_metrics', diagnosticDumpName='dms.metrics',
               samplingInterval=300, rotationCount=10)
```

```
dms_metrics is added
```

### 15.3.2 enableDumpSampling

#### Use with WLST: Online

**Description**

Enables or disables all dump samplings. This command affects all configured dump samplings.

**Syntax**

```python
enableDumpSampling(enable [,server])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>enable</td>
<td>A Boolean value that specifies whether to enable or disable dump samplings. Valid values are true and false.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. The name of the server for which to enable or disable dump sampling. If you do not specify this parameter, this command enables or disables the dump sampling for the Administration Server.</td>
</tr>
</tbody>
</table>

**Example**

The following example disables all dump samplings:

```python
enableDumpSampling(enable=false)
```
15.3.3 getSamplingArchives

Use with WLST: Online

15.3.3.1 Description
Collects all dump samplings in a zip file containing the individual sampling files and a readme file. This method is particularly useful in dealing with binary format dumps.

15.3.3.2 Syntax
getSamplingArchives([sampleName,] outputFile [,server])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>sampleName</td>
<td>Optional. The name of a particular dump sampling that you want to retrieve. If you do not specify this argument, the command returns all dump samplings.</td>
</tr>
<tr>
<td>outputFile</td>
<td>The absolute path of the file to which the dump samplings will be written.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. The name of the server from which to collect the information. If you do not specify this parameter, this command collects the dump samples for the Administration Server.</td>
</tr>
</tbody>
</table>

15.3.3.3 Example
The following example retrieves the dump sampling for the dump JVMThreadDump:

getSamplingArchives(sampleName="JVMThreadDump", outputFile="/tmp/jvm_dump.zip")
wrote 63518 bytes to /tmp/jvm_dump.zip

The following shows the contents of the zip file:

```
unzip -l jvm_dump.zip
Archive: jvm_dump.zip
  Length  Date   Time    Name
---------  ----    ----    ----
 508780  08-21-12 07:25   dfw_samplingArchive1065570966467923683.JVMThreadDump.dmp
  840    08-21-12 07:25   dfw_samplingArchive7749640004639161119.readme.txt
---------  ----    ----    ----
 509620                   2 files
```

15.3.4 isDumpSamplingEnabled

Use with WLST: Online

15.3.4.1 Description
Lists whether dump sampling is enabled or disabled.

15.3.4.2 Syntax
isDumpSamplingEnabled([server])
### 15.3.4.3 Example

The following example lists the whether dump sampling is enabled or disabled for the server `soa_server1`:

```java
isDumpSamplingEnabled(server="soa_server1")
```

Location changed to domainRuntime tree. This is a read-only tree with DomainMBean as the root. For more help, use `help(domainRuntime)`

true

### 15.3.5 listDumpSamples

Use with WLST: Online

#### 15.3.5.1 Description

Lists all dump samplings, a specified dump sampling, or all dump samplings associated with a specified server.

#### 15.3.5.2 Syntax

```java
listDumpSamples([sampleName] [, server])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>sampleName</code></td>
<td>Optional. The name of the sampling.</td>
</tr>
<tr>
<td><code>server</code></td>
<td>Optional. The name of the server for which to list the dump samplings. If you do not specify this parameter, this command lists the dump samplings for the Administration Server.</td>
</tr>
</tbody>
</table>

#### 15.3.5.3 Example

The following example lists all dump samplings associated with the server `soa_server1`:

```java
listDumpSamples(server="soa_server1")
```

Location changed to domainRuntime tree. This is a read-only tree with DomainMBean as the root. For more help, use `help(domainRuntime)`

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>: JVMThreadDump</td>
</tr>
<tr>
<td>Dump Name</td>
<td>: jvm.threads</td>
</tr>
<tr>
<td>Application Name</td>
<td>: jvm.threads</td>
</tr>
<tr>
<td>Sampling Interval</td>
<td>: 1800</td>
</tr>
<tr>
<td>Rotation Count</td>
<td>: 5</td>
</tr>
<tr>
<td>Dump Implicitly</td>
<td>: false</td>
</tr>
<tr>
<td>Append Samples</td>
<td>: true</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>: JavaClassHistogram</td>
</tr>
<tr>
<td>Dump Name</td>
<td>: jvm.classhistogram</td>
</tr>
<tr>
<td>Application Name</td>
<td>: jvm.classhistogram</td>
</tr>
<tr>
<td>Sampling Interval</td>
<td>: 1800</td>
</tr>
<tr>
<td>Rotation Count</td>
<td>: 5</td>
</tr>
<tr>
<td>Dump Implicitly</td>
<td>: false</td>
</tr>
<tr>
<td>Append Samples</td>
<td>: true</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
</table>
Sampling Interval : 60
Rotation Count : 10
Dump Implicitly : true
Append Samples : true
Dump Arguments : context=true, timing=true

15.3.6 removeDumpSample
Use with WLST: Online

15.3.6.1 Description
Removes the dump sampling.

15.3.6.2 Syntax
removeDumpSample(sampleName [,server])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>sampleName</td>
<td>The name of the sampling to be removed.</td>
</tr>
<tr>
<td>server</td>
<td>Optional. The name of the server from which to remove the sampling. If you do not specify this parameter, the dump sampling is removed from the Administration Server.</td>
</tr>
</tbody>
</table>

15.3.6.3 Example
The following example removes the dump sampling named HTTPSampling, associated with the server soa_server1:

removeDumpSample(sampleName="HTTPSampling", server="soa_server1")

Removed HTTPSampling

15.3.7 updateDumpSample
Use with WLST: Online

15.3.7.1 Description
Updates the specified dump sampling, modifying the settings of the sampling. You cannot change the name of the sampling. Modifications take affect at the next sampling interval.

15.3.7.2 Syntax
updateDumpSample(sampleName [, appName], samplingInterval, rotationCount [,dumpedImplicitly] [, toAppend] [, arg], [ , server])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>sampleName</td>
<td>The name of the sampling.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. The name of the application associated with the specified diagnostic dump. If you do not specify appName, the diagnostic dump has a scope of system.</td>
</tr>
<tr>
<td>samplingInterval</td>
<td>Optional. The sampling interval in seconds. If you specify zero or a negative value, sampling is suspended.</td>
</tr>
</tbody>
</table>
15.3.7.3 Example

The following example updates the dump sampling HTTPSampling, modifying the sampling interval, rotation count, and server.

updateDumpSample(sampleName="HTTPSampling", samplingInterval=200, rotationCount=5, server="soa_server1")

HTTPSampling is updated
This chapter provides detailed descriptions of custom WLST commands for Oracle Fusion Middleware Information Rights Management, including command syntax, arguments and command examples.

The following sections describe the Oracle Fusion Middleware Information Rights Management custom WLST commands in detail. Topics include:

- Section 16.1, "Overview of WLST IRM Commands"
- Section 16.2, "General Server Commands"
- Section 16.3, "Migration Commands"
- Section 16.4, "Test Content Commands"
- Section 16.5, "Languages Support Commands"
- Section 16.6, "Oracle IRM Desktop Installers Commands"

For additional information about Oracle Information Rights Management, see Oracle IRM Administrator’s Guide.

---

**Note:** To use these commands, you must invoke WLST from the Oracle home in which the component has been installed. See "Using Custom WLST Commands" in the Oracle Fusion Middleware Administrator’s Guide.

### 16.1 Overview of WLST IRM Commands

WLST IRM commands are divided into the following categories:

<table>
<thead>
<tr>
<th>Command Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Server Commands</td>
<td>Make general changes to Oracle IRM Server settings.</td>
</tr>
<tr>
<td>Migration Commands</td>
<td>Back up and migrate Oracle IRM Server user data.</td>
</tr>
<tr>
<td>Test Content Commands</td>
<td>Set up test content for users of Oracle IRM Desktop.</td>
</tr>
<tr>
<td>Languages Support Commands</td>
<td>Set up languages support for users of Oracle IRM Server.</td>
</tr>
<tr>
<td>Oracle IRM Desktop Installers Commands</td>
<td>Set up software installation support for Oracle IRM Desktop.</td>
</tr>
</tbody>
</table>
16.2 General Server Commands

Use the WLST commands listed in Table 16–2 to make general changes to Oracle IRM Server settings.

Table 16–2  WLST General Server Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>addIRMRefreshPeriod</td>
<td>Create a new refresh period.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMRefreshPeriod</td>
<td>Display an existing refresh period.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMRefreshPeriods</td>
<td>Display all the refresh periods.</td>
<td>Online</td>
</tr>
<tr>
<td>removeIRMRefreshPeriod</td>
<td>Remove an existing refresh period.</td>
<td>Online</td>
</tr>
<tr>
<td>updateIRMRefreshPeriod</td>
<td>Update an existing refresh period.</td>
<td>Online</td>
</tr>
<tr>
<td>addIRMSyncWindow</td>
<td>Create a new sync window.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMSyncWindow</td>
<td>Display an existing sync window.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMSyncWindows</td>
<td>Display all the sync windows.</td>
<td>Online</td>
</tr>
<tr>
<td>removeIRMSyncWindow</td>
<td>Remove an existing sync window.</td>
<td>Online</td>
</tr>
<tr>
<td>updateIRMSyncWindow</td>
<td>Update an existing sync window.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMCryptoSchema</td>
<td>Display the cryptography algorithm.</td>
<td>Online</td>
</tr>
<tr>
<td>setIRMCryptoSchema</td>
<td>Set the cryptography algorithm.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMDeviceCount</td>
<td>Display the device count.</td>
<td>Online</td>
</tr>
<tr>
<td>setIRMDeviceCount</td>
<td>Set the device count.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMJournalCleanUp</td>
<td>Display the current report record clean-up values.</td>
<td>Online</td>
</tr>
<tr>
<td>setIRMJournalCleanUp</td>
<td>Set report record clean-up values.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMLicenseStateCleanUp</td>
<td>Display the license state clean-up frequency.</td>
<td>Online</td>
</tr>
<tr>
<td>setIRMLicenseStateCleanUp</td>
<td>Set the license state clean-up frequency.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMPrivacyURL</td>
<td>Display the URL of the privacy statement page.</td>
<td>Online</td>
</tr>
<tr>
<td>setIRMPrivacyURL</td>
<td>Set the URL of the privacy statement page.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMKeyStore</td>
<td>Display the type and location of the Oracle IRM keystore.</td>
<td>Online</td>
</tr>
<tr>
<td>setIRMKeyStore</td>
<td>Set the type and location of the Oracle IRM keystore.</td>
<td>Online</td>
</tr>
</tbody>
</table>

16.2.1 addIRMRefreshPeriod

Online command that creates a new refresh period.

16.2.1.1 Description

This command creates a refresh period. A refresh period is the maximum length of time that a user can use rights before the rights are refreshed from the server.

16.2.1.2 Syntax

addIRMRefreshPeriod(duration,dtype)
16.2.1.3 Examples
The following example creates a refresh period of 5 hours:

```
wlsl/base_domain/serverConfig> addIRMRefreshPeriod(5,"HOURS")
```

The following example creates a refresh period of 50 minutes:

```
wlsl/base_domain/serverConfig> addIRMRefreshPeriod(50,"MINUTES")
```

16.2.2 getIRMRefreshPeriod
Online command that displays an existing refresh period.

16.2.2.1 Description
This command displays the refresh period that is present at the specified index. A refresh period is the maximum length of time that a user can use rights before the rights are refreshed from the server.

16.2.2.2 Syntax
```
getIRMRefreshPeriod(pindex)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pindex</td>
<td>Specifies the index of the refresh period.</td>
</tr>
</tbody>
</table>

16.2.2.3 Examples
The following example displays the refresh period that is present at index zero:

```
wls:/base_domain/serverConfig> getIRMRefreshPeriod(0)
```

The following example displays the refresh period that is present at index one:

```
wls:/base_domain/serverConfig> getIRMRefreshPeriod(1)
```

16.2.3 getIRMRefreshPeriods
Online command that displays all existing refresh periods.

16.2.3.1 Description
This command displays all existing refresh periods. A refresh period is the maximum length of time that a user can use rights before the rights are refreshed from the server.

16.2.3.2 Syntax
```
getIRMRefreshPeriods()
```

16.2.3.3 Example
```
wls:/base_domain/serverConfig> getIRMRefreshPeriods()
```
16.2.4 removeIRMRefreshPeriod

Online command that removes an existing refresh period.

16.2.4.1 Description
This command removes a refresh period that is present at the specified index. A
refresh period is the maximum length of time that a user can use rights before the
rights are refreshed from the server.

16.2.4.2 Syntax
removeIRMRefreshPeriod(pindex)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pindex</td>
<td>Specifies the index of the refresh period.</td>
</tr>
</tbody>
</table>

16.2.4.3 Examples
The following example removes the refresh period that is present at index zero:

```
wls:/base_domain/serverConfig> removeIRMRefreshPeriod(0)
```

The following example removes the refresh period that is present at index one:

```
wls:/base_domain/serverConfig> removeIRMRefreshPeriod(1)
```

16.2.5 updateIRMRefreshPeriod

Online command that updates an existing refresh period.

16.2.5.1 Description
This command updates an existing refresh period. A refresh period is the maximum
length of time that a user can use rights before the rights are refreshed from the server.

16.2.5.2 Syntax
updateIRMRefreshPeriod(pindex,duration,dtype)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>pindex</td>
<td>Specifies the index of the refresh period.</td>
</tr>
<tr>
<td>duration</td>
<td>Specifies the value of the refresh period. Integer.</td>
</tr>
</tbody>
</table>
| dtype         | Specifies the unit of the refresh period. 'MINUTES', 'HOURS', 'DAYS',
                      'MONTHS' or 'YEARS'. |

16.2.5.3 Examples
The following example updates the refresh period at index zero to have a duration of 5
hours:

```
wls:/base_domain/serverConfig> updateIRMRefreshPeriod(0,5,"HOURS")
```

The following example updates the refresh period at index zero to have a duration of
50 minutes:

```
wls:/base_domain/serverConfig> updateIRMRefreshPeriod(0,50,"MINUTES")
```
16.2.6 addIRMSSyncWindow

Online command that creates a sync window.

16.2.6.1 Description
This command creates a sync window. A sync window is a period during which Oracle IRM Desktop will attempt to contact the server to synchronize rights.

16.2.6.2 Syntax
addIRMSSyncWindow(day, stHrs, stMins, endHrs, endMins)

16.2.6.3 Example
The following example creates a sync window that will result in Oracle IRM Desktop attempting to contact the server between 9.30am and 6.30pm on Mondays:

```
wls:/base_domain/serverConfig> addIRMSSyncWindow("MONDAY", 9, 30, 6, 30)
```

16.2.7 getIRMSSyncWindow

Online command that displays an existing sync window.

16.2.7.1 Description
This command displays a sync window that is present at the specified index. A sync window is a period during which Oracle IRM Desktop will attempt to contact the server to synchronize rights.

16.2.7.2 Syntax
getIRMSSyncWindow(sindex)

16.2.7.3 Examples
The following example displays the sync window at index zero:

```
wls:/base_domain/serverConfig> getIRMSSyncWindow(0)
```

The following example displays the sync window at index one:

```
wls:/base_domain/serverConfig> getIRMSSyncWindow(1)
```

16.2.8 getIRMSSyncWindows

Online command that displays all existing sync windows.
16.2.8.1 Description
This command displays all existing sync windows. A sync window is a period during which Oracle IRM Desktop will attempt to contact the server to synchronize rights.

16.2.8.2 Syntax
getIRMSyncWindows()

16.2.8.3 Example
wls:/base_domain/serverConfig> getIRMSyncWindows()

16.2.9 removeIRMSyncWindow
Online command that removes an existing sync window.

16.2.9.1 Description
This command removes a sync window that is present at the specified index. A sync window is a period during which Oracle IRM Desktop will attempt to contact the server to synchronize rights.

16.2.9.2 Syntax
removeIRMSyncWindow(sindex)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>sindex</td>
<td>Specifies the index of the sync window.</td>
</tr>
</tbody>
</table>

16.2.9.3 Examples
The following example removes the sync window at index zero:

wls:/base_domain/serverConfig> removeIRMSyncWindow(0)

The following example removes the sync window at index one:

wls:/base_domain/serverConfig> removeIRMSyncWindow(1)

16.2.10 updateIRMSyncWindow
Online command that updates an existing sync window.

16.2.10.1 Description
This command updates an existing sync window. A sync window is a period during which Oracle IRM Desktop will attempt to contact the server to synchronize rights.

16.2.10.2 Syntax
updateIRMSyncWindow(indexOfDay, day, stHrs, stMins, endHrs, endMins)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>indexOfDay</td>
<td>Specifies the index of the sync window.</td>
</tr>
<tr>
<td>day</td>
<td>Specifies the weekday. 'MONDAY', 'TUESDAY', etc.</td>
</tr>
<tr>
<td>stHrs</td>
<td>Specifies the start hours. Integer.</td>
</tr>
<tr>
<td>stMins</td>
<td>Specifies the start minutes. Integer.</td>
</tr>
<tr>
<td>endHrs</td>
<td></td>
</tr>
<tr>
<td>endMins</td>
<td></td>
</tr>
</tbody>
</table>
16.2.10.3 Example
The following example updates the sync window at index zero so that Oracle IRM Desktop will attempt to contact the server between 9.30am and 5.30pm on Mondays:

```plaintext
wls:/base_domain/serverConfig> updateIRMSyncWindow(0,"MONDAY",9,30,5,30)
```

16.2.11 getIRMCryptoSchema
Online command that displays the cryptography algorithm.

16.2.11.1 Description
This command displays the cryptography algorithm currently applied to files that are sealed using Oracle IRM.

16.2.11.2 Syntax
```plaintext
getIRMCryptoSchema()
```

16.2.11.3 Example
```plaintext
wls:/base_domain/serverConfig> getIRMCryptoSchema()
```

16.2.12 setIRM CryptoSchema
Online command that sets the cryptography algorithm.

16.2.12.1 Description
This command sets the cryptography algorithm that will be applied to files that are sealed using Oracle IRM. The default of AES128 is recommended.

16.2.12.2 Syntax
```plaintext
setIRM CryptoSchema(cryptID)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>cryptID</td>
<td>Specifies the name of the cryptography algorithm. Possible algorithm names are AES128, AES256, AES128-FIPS, AES256-FIPS, DES3-FIPS.</td>
</tr>
</tbody>
</table>

16.2.12.3 Example
The following example sets the cryptography algorithm used for Oracle IRM communications to AES128:

```plaintext
wls:/base_domain/serverConfig> setIRM CryptoSchema("AES128")
```

16.2.13 getIRMDeviceCount
Online command that displays the device count.
16.2.13.1 Description
This command displays the maximum number of devices on which a user can open a sealed document at one time. The value applies to all users, and does not differ for individual users.

16.2.13.2 Syntax
getIRMDeviceCount()

16.2.13.3 Example
wls:/base_domain/serverConfig> getIRMDeviceCount()

16.2.14 setIRMDeviceCount
Online command that sets the device count.

16.2.14.1 Description
This command sets the maximum number of devices on which a user can open a sealed document at one time. The value applies to all users. The device count is normally kept low (1 or 2) to make it difficult to circumvent document access restrictions by sharing passwords.

16.2.14.2 Syntax
setIRMDeviceCount(devCount)

16.2.14.3 Example
The following example sets the device count to 2:
wls:/base_domain/serverConfig> setIRMDeviceCount(2)

16.2.15 getIRMJournalCleanUp
Online command that displays the current report record clean-up values.

16.2.15.1 Description
This command displays the report record clean-up values. The values show how often report record clean-ups are performed, and the maximum age of the report records before they are deleted.

16.2.15.2 Syntax
getIRMJournalCleanUp()

16.2.15.3 Example
wls:/base_domain/serverConfig> getIRMJournalCleanUp()

16.2.16 setIRMJournalCleanUp
Online command that sets report record clean-up values.
16.2.16.1 Description
This command sets how often report record clean-ups are performed, and the maximum age of report records before they are deleted.

16.2.16.2 Syntax
setIRMJournalCleanUp(clDuration, clUnitType, retDuration, retUnitType)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>clDuration</td>
<td>Specifies the value for how often report record clean-ups are performed. Integer.</td>
</tr>
<tr>
<td>clUnitType</td>
<td>Specifies the unit for how often report record clean-ups are performed. 'MINUTES', 'HOURS', 'DAYS', 'MONTHS', 'YEARS'.</td>
</tr>
<tr>
<td>retDuration</td>
<td>Specifies the value for the maximum age of report records before they are deleted. Integer.</td>
</tr>
<tr>
<td>retUnitType</td>
<td>Specifies the unit for the maximum age of report records before they are deleted. 'MINUTES', 'HOURS', 'DAYS', 'MONTHS', 'YEARS'.</td>
</tr>
</tbody>
</table>

16.2.16.3 Example
The following example runs report record clean-ups every 5 days, and deletes report records that are 6 months old:

wls:/base_domain/serverConfig> setIRMJournalCleanUp(5, "DAYS", 6, "MONTHS")

16.2.17 getIRMLicenseStateCleanUp
Online command that displays the license state clean-up frequency.

16.2.17.1 Description
This command displays the license state clean-up frequency (the frequency at which license records will be deleted).

16.2.17.2 Syntax
getIRMLicenseStateCleanUp() 

16.2.17.3 Example
wls:/base_domain/serverConfig> getIRMLicenseStateCleanUp()

16.2.18 setIRMLicenseStateCleanUp
Online command that sets the license state clean-up frequency.

16.2.18.1 Description
This command sets the license state clean-up frequency (the frequency at which license records will be deleted).

16.2.18.2 Syntax
setIRMLicenseStateCleanUp(duration, unitType)
### 16.2.18 setIRMLicenseStateCleanUp

#### Description
This command sets the frequency at which license records will be deleted from the cache.

#### Syntax
```
setIRMLicenseStateCleanUp(duration, unitType)
```

#### Examples
- The following example sets the frequency at which license records will be deleted to 10 hours:
  ```
wls:/base_domain/serverConfig> setIRMLicenseStateCleanUp(10, "HOURS")
  ```
- The following example sets the frequency at which license records will be deleted to 50 minutes:
  ```
wls:/base_domain/serverConfig> setIRMLicenseStateCleanUp(50, "MINUTES")
  ```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>duration</td>
<td>Specifies the value of the frequency at which license records will be deleted. Integer.</td>
</tr>
<tr>
<td>unitType</td>
<td>Specifies the unit for the frequency at which license records will be deleted. 'MINUTES', 'HOURS', 'DAYS', 'MONTHS', 'YEARS'.</td>
</tr>
</tbody>
</table>

### 16.2.19 getIRMPrivacyURL

#### Description
Online command that displays the URL of the privacy statement page.

#### Syntax
```
getIRMPrivacyURL()
```

#### Example
- The following example sets the URL of the privacy policy page to "http://irm.example.com/":
  ```
wls:/base_domain/serverConfig> getIRMPrivacyURL("http://irm.example.com")
  ```

### 16.2.20 setIRMPrivacyURL

#### Description
Online command that sets the URL of the privacy statement page.

#### Syntax
```
setIRMPrivacyURL(privacyURL)
```

#### Example
- The following example sets the URL of the privacy policy page to "http://irm.example.com/":
  ```
wls:/base_domain/serverConfig> setIRMPrivacyURL("http://irm.example.com")
  ```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>privacyURL</td>
<td>Specifies the URL of the privacy statement page.</td>
</tr>
</tbody>
</table>
16.2.21  getIRMKeyStore

Online command that displays the type and location of the Oracle IRM keystore.

16.2.21.1  Description

This command displays the type and location of the Oracle IRM keystore.

16.2.21.2  Syntax

getIRMKeyStore()

16.2.21.3  Examples

wls:/base_domain/serverConfig> getIRMKeyStore()

16.2.22  setIRMKeyStore

Online command that sets the type and location of the Oracle IRM keystore.

16.2.22.1  Description

This command sets the type and location of the Oracle IRM keystore. You should not normally need to change the keystore type and location from the default (type JKS at location ${domain.home}/config/fmwconfig/irm.jks).

16.2.22.2  Syntax

setIRMKeyStore()

You will be prompted to provide the following arguments:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>KeyStore Type</td>
<td>Specifies the type of the keystore.</td>
</tr>
<tr>
<td>KeyStore Location</td>
<td>Specifies the location of the keystore.</td>
</tr>
</tbody>
</table>

16.2.22.3  Example

The following example sets the keystore type to JCEKS and the keystore location to D:/exampledir/:

wls:/base_domain/serverConfig> setIRMKeyStore()
Enter KeyStore Type: JCEKS
Enter KeyStore Location: D:/exampledir/

16.3  Migration Commands

Use the WLST commands listed in Table 16–3 to set up import and export of user data between instances of Oracle IRM Server.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>setIRMExportFolder</td>
<td>Set or clear the data export folder location.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMExportFolder</td>
<td>Display the value for the data export folder.</td>
<td>Online</td>
</tr>
<tr>
<td>setIRMImportFolder</td>
<td>Set or clear the data import folder location.</td>
<td>Online</td>
</tr>
</tbody>
</table>
16.3.1 setIRMExportFolder

Online command that sets or clears the data export folder location.

16.3.1.1 Description
This command sets or clears the location of the folder used for data export.

16.3.1.2 Syntax

\[ \text{setIRMExportFolder(folder)} \]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>\text{folder}</td>
<td>Specifies the data export folder value.</td>
</tr>
</tbody>
</table>

16.3.1.3 Example

\text{wls:/base_{domain}/serverConfig> setIRMExportFolder("export")}

16.3.2 getIRMExportFolder

Online command that displays the value of the data export folder.

16.3.2.1 Description
This command displays the location of the folder used for data export.

16.3.2.2 Syntax

\[ \text{getIRMExportFolder()} \]

16.3.2.3 Example

\text{wls:/base_{domain}/serverConfig> getIRMExportFolder()}

16.3.3 setIRMImportFolder

Online command that sets or clears the data import folder location.

16.3.3.1 Description
This command sets or clears the location of the folder used for data import.

16.3.3.2 Syntax

\[ \text{setIRMImportFolder(folder)} \]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>\text{folder}</td>
<td>Specifies the import folder value.</td>
</tr>
</tbody>
</table>
16.3.3 Example

wls:/base_domain/serverConfig> setIRMImportFolder("import")

16.3.4 getIRMImportFolder

Online command that displays the value of the data import folder.

16.3.4.1 Description

This command displays the location of the folder used for data import.

16.3.4.2 Syntax

getIRMImportFolder()

16.3.4.3 Example

wls:/base_domain/serverConfig> getIRMImportFolder()

16.4 Test Content Commands

Use the WLST commands listed in Table 16–4 to set up test content for users of Oracle IRM Desktop.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>addIRMTestContent</td>
<td>Create a new test content instance.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMTestContent</td>
<td>Display details for an existing test content instance.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMTestContents</td>
<td>Display details of all existing test content instances.</td>
<td>Online</td>
</tr>
<tr>
<td>removeIRMTestContent</td>
<td>Remove an existing test content instance.</td>
<td>Online</td>
</tr>
<tr>
<td>updateIRMTestContent</td>
<td>Update an existing text content instance.</td>
<td>Online</td>
</tr>
</tbody>
</table>

16.4.1 addIRMTestContent

Online command that creates a new test content instance.

16.4.1.1 Description

This command creates a test content instance. Test content instances identify an item of test content, usually an image file. Test content is shown in a sealed document when Oracle IRM Desktop successfully connects to Oracle IRM Server through the client test facility.

16.4.1.2 Syntax

addIRMTestContent(uri,localeKeys,testNames)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>uri</td>
<td>Specifies the URI of the test content (for example, an image file).</td>
</tr>
</tbody>
</table>
16.4.1.3 Examples

The following example creates a test content instance comprising an image named exampleImage.jpg at http://irm.example.com, for use with English installations, and showing the name 'Test Content':

```wls:/base_domain/serverConfig>
addIRMTestContent("http://irm.example.com/exampleImage.jpg","en","Test Content")
```

The following example creates a test content instance comprising an image named exampleImage.jpg at http://irm.example.com, for use with English and French installations, and showing the names 'Test Content (en)' and 'Test Content (fr)'

```wls:/base_domain/serverConfig>
addIRMTestContent("http://irm.example.com/exampleImage.jpg","en,fr","Test Content (en),Test Content (fr)")
```

16.4.2 getIRMTestContent

Online command that displays the details of an existing test content instance.

16.4.2.1 Description

This command displays the details of the test content instance that is present at the specified index. Test content instances identify an item of test content, usually an image file. Test content is shown in a sealed document when Oracle IRM Desktop successfully connects to Oracle IRM Server through the client test facility.
16.4.2.2 Syntax
getIRMTestContent(tindex)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>tindex</td>
<td>Specifies the index of the test content instance.</td>
</tr>
</tbody>
</table>

16.4.2.3 Examples
The following example displays the details of the test content instance at index zero:

    wls:/base_domain/serverConfig> getIRMTestContent(0)

The following example displays the details of the test content instance at index one:

    wls:/base_domain/serverConfig> getIRMTestContent(1)

16.4.3 getIRMTestContents
Online command that displays all the test content instances.

16.4.3.1 Description
This command displays all the test content instances. Test content instances identify an item of test content, usually an image file. Test content is shown in a sealed document when Oracle IRM Desktop successfully connects to Oracle IRM Server through the client test page.

16.4.3.2 Syntax
getIRMTestContents()

16.4.3.3 Example

    wls:/base_domain/serverConfig> getIRMTestContents()

16.4.4 removeIRMTestContent
Online command that removes an existing test content instance.

16.4.4.1 Description
This command removes the test content instance that is present at the specified index. Test content instances identify an item of test content, usually an image file.

16.4.4.2 Syntax
removeIRMTestContent(tindex)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>tindex</td>
<td>Specifies the index of test content.</td>
</tr>
</tbody>
</table>

16.4.4.3 Examples
The following example removes the test content instance at index zero:

    wls:/base_domain/serverConfig> removeIRMTestContent(0)

The following example removes the test content instance at index one:
16.4.5 updateIRMTestContent

Online command that updates an existing test content instance.

16.4.5.1 Description
This command updates an existing test content instance. Test content instances identify an item of test content, usually an image file. Test content is shown in a sealed document when Oracle IRM Desktop successfully connects to Oracle IRM Server through the client test facility.

16.4.5.2 Syntax
updateIRMTestContent(tindex, uri, localeKeys, testNames)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>tindex</td>
<td>Specifies the index of the test content instance. Integer.</td>
</tr>
<tr>
<td>uri</td>
<td>Specifies the URI of the test content (for example, an image file).</td>
</tr>
<tr>
<td>localeKeys</td>
<td>Specifies the locale(s) associated with this test content instance. Must be from the list of two-letter language codes given in Table 16–5 (for example, 'fr' for French). If there is more than one supported locale for an instance, the two-letter codes must be listed as comma-separated values.</td>
</tr>
<tr>
<td>testNames</td>
<td>Specifies the name(s) associated with this test content instance. If there is more than one name for a URI, they must be specified as comma-separated values.</td>
</tr>
</tbody>
</table>

16.4.5.3 Examples
The following example updates a test content instance by changing the image to exampleImage.jpg at http://irm.example.com, for use with English installations, and showing the name 'Test Content':

wls:/base_domain/serverConfig> updateIRMTestContent(0,"http://irm.example.com/exampleImage.jpg","en","Test Content")

The following example updates a test content instance by changing the image to exampleImage.jpg at http://irm.example.com, for use with English and French installations, and showing the names 'Test Content (English)' and 'Test Content (French)'

wls:/base_domain/serverConfig> updateIRMTestContent(0,"http://irm.example.com/exampleImage.jpg","en,fr","Test Content (English),Test Content (French)")

16.5 Languages Support Commands
Use the WLST commands listed in Table 16–6 to set up languages support for users of Oracle IRM Server.
### 16.5.1 addIRMTranslation

Online command that creates a new language support instance.

#### 16.5.1.1 Description
This command creates a new language support instance. Each language support instance provides the facility in Oracle IRM Server to add names and descriptions in one or more languages (in addition to the default language).

#### 16.5.1.2 Syntax
```
addIRMTranslation(transList)
```

#### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>transList</td>
<td>Specifies the supported language(s). Must be from the list of two-letter language codes given in Table 16–5 (for example, 'fr' for French). If there is more than one supported language for an instance, the two-letter codes must be listed as comma-separated values.</td>
</tr>
</tbody>
</table>

#### 16.5.1.3 Examples
The following example creates a language support instance that will enable users of Oracle IRM Server to add names and descriptions in French (in addition to their default language):

```
wls:/base_domain/serverConfig> addIRMTranslation("fr")
```

The following example creates a language support instance that will enable users of Oracle IRM Server to add names and descriptions in French and Arabic (in addition to their default language):

```
wls:/base_domain/serverConfig> addIRMTranslation("fr,ar")
```

### 16.5.2 getIRMDefaultTranslation

Online command that displays the default language.

#### 16.5.2.1 Description
This command displays the default language.
16.5.2.2 Syntax
getIRMDefaultTranslation()

16.5.2.3 Example
wls:/base_domain/serverConfig> getIRMDefaultTranslation()

16.5.3 getIRMTranslations
Online command that displays all the language support instances.

16.5.3.1 Description
This command displays all the language support instances. Each language support instance provides the facility in Oracle IRM Server to add names and descriptions in one or more languages (in addition to the default language).

16.5.3.2 Syntax
getIRMTranslations()

16.5.3.3 Example
wls:/base_domain/serverConfig> getIRMTranslations()

16.5.4 removeIRMTranslation
Online command that removes an existing language support instance.

16.5.4.1 Description
This command removes the language support instance that is present at the specified index. Each language support instance provides the facility in Oracle IRM Server to add names and descriptions in one or more languages (in addition to the default language).

16.5.4.2 Syntax
removeIRMTranslation(tindex)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>tindex</td>
<td>Specifies the index of the language support instance.</td>
</tr>
</tbody>
</table>

16.5.4.3 Examples
The following example removes the language support instance at index zero:

wls:/base_domain/serverConfig> removeIRMTranslation(0)

The following example removes the language support instance at index one:

wls:/base_domain/serverConfig> removeIRMTranslation(1)

16.5.5 setIRMTranslations
Online command that sets the default language, and sets a language support instance for one or more languages in addition to the default language.
16.5.5.1 Description
This command sets the default language, and sets a language support instance for one or more languages in addition to the default language. Each language support instance provides the facility in Oracle IRM Server to add names and descriptions in one or more languages (in addition to the default language).

16.5.5.2 Syntax
setIRMTranslations(defaultTrans,transList)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultTrans</td>
<td>Specifies the default language. Language code (for example, 'en' for English).</td>
</tr>
<tr>
<td>transList</td>
<td>Specifies the supported language(s). Must be from the list of two-letter language codes given in Table 16–5 (for example, 'fr' for French). If there is more than one supported language for an instance, the two-letter codes must be listed as comma-separated values.</td>
</tr>
</tbody>
</table>

16.5.5.3 Examples
The following example enables users of Oracle IRM Server to enter names and descriptions in English as the default language, and additionally to enter names and descriptions in French:

```
wlst:/base_domain/serverConfig> setIRMTranslations("en","fr")
```

The following example enables users of Oracle IRM Server to enter names and descriptions in English as the default language, and additionally to enter names and descriptions in French and Arabic:

```
wlst:/base_domain/serverConfig> setIRMTranslations("en","fr,ar")
```

16.6 Oracle IRM Desktop Installers Commands
Use the WLST commands listed in Table 16–7 to set up installation support for Oracle IRM Desktop software.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>addIRMDownload</td>
<td>Create a new installer.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMDownload</td>
<td>Display the details for an existing installer.</td>
<td>Online</td>
</tr>
<tr>
<td>getIRMDownloads</td>
<td>Display the details for all installers.</td>
<td>Online</td>
</tr>
<tr>
<td>removeIRMDownload</td>
<td>Remove an existing installer.</td>
<td>Online</td>
</tr>
<tr>
<td>updateIRMDownload</td>
<td>Update an existing installer.</td>
<td>Online</td>
</tr>
</tbody>
</table>

16.6.1 addIRMDownload
Online command that creates a new installer.
16.6.1 Description
This command creates a new installer. Each installer identifies the locale and URI of software for installing Oracle IRM Desktop, and displays a name and version number that enables users of Oracle IRM Server to select the installer.

16.6.1.2 Syntax
addIRMDownload(locale,name,version,uri)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>locale</td>
<td>Specifies the locale of the installer. Must be from the list of two-letter language codes given in Table 16–5 (for example, 'en' for English).</td>
</tr>
<tr>
<td>name</td>
<td>Specifies the name for the installer.</td>
</tr>
<tr>
<td>version</td>
<td>Specifies the version of the installer. This is a label for the installer, and is not verified against the associated installation software.</td>
</tr>
<tr>
<td>uri</td>
<td>Specifies the URI of Oracle IRM Desktop installation software.</td>
</tr>
</tbody>
</table>

16.6.1.3 Example
The following example creates an installer for English language installation software at http://irm.example.com/, with the name 'Oracle IRM Desktop' and the version number 11.1.1.0.0 visible to users of Oracle IRM Server when they select this installer:

```bash
wls:/base_domain/serverConfig> addIRMDownload("en","Oracle IRM Desktop","11.1.1.0.0","http://irm.example.com")
```

16.6.2 getIRMDownload
Online command that displays the details for an existing installer.

16.6.2.1 Description
This command displays the details for an installer that is present at the specified index. Each installer identifies the locale and URI of software for installing Oracle IRM Desktop, and displays a name and version number that enables users of Oracle IRM Server to select the installer.

16.6.2.2 Syntax
getIRMDownload(dindex)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>dindex</td>
<td>Specifies the index of the download.</td>
</tr>
</tbody>
</table>

16.6.2.3 Examples
The following example displays the details for the installer at index zero:

```bash
wls:/base_domain/serverConfig> getIRMDownload(0)
```

The following example displays the details for the installer at index one:

```bash
wls:/base_domain/serverConfig> getIRMDownload(1)
```

16.6.3 getIRMDownloads
Online command that displays the details of all installers.
16.6.3.1 Description
This command displays the details of all installers. Each installer identifies the locale and URI of software for installing Oracle IRM Desktop, and displays a name and version number that enables users of Oracle IRM Server to select the installer.

16.6.3.2 Syntax
getIRMDownloads()

16.6.3.3 Example
wls:/base_domain/serverConfig> getIRMDownloads()

16.6.4 removeIRMDownload
Online command that removes an existing installer.

16.6.4.1 Description
Removes the installer that is present at the specified index. Each installer identifies the locale and URI of software for installing Oracle IRM Desktop, and displays a name and version number that enables users of Oracle IRM Server to select the installer.

16.6.4.2 Syntax
removeIRMDownload(dindex)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>dindex</td>
<td>Specifies the index of the download.</td>
</tr>
</tbody>
</table>

16.6.4.3 Examples
The following example removes the installer at index zero:
wls:/base_domain/serverConfig> removeIRMDownload(0)

The following example removes the installer at index one:
wls:/base_domain/serverConfig> removeIRMDownload(1)

16.6.5 updateIRMDownload
Online command that updates an existing installer.

16.6.5.1 Description
This command updates an existing installer. Each installer identifies the locale and URI of software for installing Oracle IRM Desktop, and displays a name and version number that enables users of Oracle IRM Server to select the installer.

16.6.5.2 Syntax
updateIRMDownload(dindex,locale,name,version,uri)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>dindex</td>
<td>Specifies the index of the installer. Integer.</td>
</tr>
</tbody>
</table>
### 16.6.5.3 Example

The following example updates the installer for index zero. After the update, the installation software is English language and is located at http://irm.example.com/. The name ‘Oracle IRM Desktop (English)’ and the version number 11.1.1.1.0.0 will be visible to users of Oracle IRM Server when they select this installer.

```
wls:/base_domain/serverConfig> updateIRMDownload(0,"en", "Oracle IRM Desktop (English)", "11.1.1.1.0.0", "http://irm.example.com/")
```
This chapter provides detailed descriptions of custom WLST commands for Oracle WebCenter: Imaging, including command syntax, arguments and command examples. The following sections describe the WLST commands that are specific to Oracle WebCenter: Imaging. Topics include:

- Section 17.1, "Overview of Imaging WLST Command Categories"
- Section 17.2, "Diagnostic Commands"
- Section 17.3, "Imaging Configuration Commands"

### 17.1 Overview of Imaging WLST Command Categories

WLST commands specific to Imaging are divided into the following categories.

<table>
<thead>
<tr>
<th>Command category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diagnostic Commands</td>
<td>Return workflow agent and other processing information.</td>
</tr>
<tr>
<td>Imaging Configuration Commands</td>
<td>Configure settings specific to Imaging and Process Management.</td>
</tr>
</tbody>
</table>

### 17.2 Diagnostic Commands

Use the Imaging WLST diagnostic commands, listed in table Table 17–2, to list and organize processing failures during workflow processes.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>clearIPMWorkflowFaults</td>
<td>Clear processing failures that occurred during workflow agent processing.</td>
<td>Online</td>
</tr>
<tr>
<td>listIPMWorkflowFaults</td>
<td>Provide details of processing failures that occurred during workflow agent processing.</td>
<td>Online</td>
</tr>
<tr>
<td>repairIPMWorkflowFaults</td>
<td>Repair processing failures that occurred during workflow agent processing.</td>
<td>Online</td>
</tr>
<tr>
<td>sumIPMWorkflowFaults</td>
<td>Count processing failures during workflow agent processing, grouped by choice of date, application ID, or batch ID.</td>
<td>Online</td>
</tr>
</tbody>
</table>
17.2.1 clearIPMWorkflowFaults

Command Category: Diagnostic Commands
Use with WLST: Online

17.2.1.1 Description
Clear processing failures that have occurred during workflow agent processing.

17.2.1.2 Syntax

\[
\text{clearIPMWorkflowFaults}(\text{startDate}, \text{endDate}, \text{appId}, \text{batchId})
\]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>Optional. The start of the date range for which error details should be repaired, in yyyy-MM-dd format.</td>
</tr>
<tr>
<td>endDate</td>
<td>Optional. The end of the date range for which error details should be repaired, in yyyy-MM-dd format.</td>
</tr>
<tr>
<td>appId</td>
<td>Optional. The application ID for which error details should be repaired, in yyyy-MM-dd format.</td>
</tr>
<tr>
<td>batchId</td>
<td>Optional. The batch ID for which error details should be repaired.</td>
</tr>
</tbody>
</table>

17.2.1.3 Example
The following example clears the faults within the specified parameters.

\[
\begin{align*}
\text{clearIPMWorkflowFaults}(\text{startDate}=&\text{"2009-06-01"}, \text{endDate}=&\text{"2009-06-02"}) \\
\text{clearIPMWorkflowFaults}(\text{appId}=3) \\
\text{clearIPMWorkflowFaults}(\text{batchId}=15) \\
\text{clearIPMWorkflowFaults}(\text{startDate}=&\text{"2009-06-01"}, \text{endDate}=&\text{"2009-06-02"}, \text{appId}=3)
\end{align*}
\]

17.2.2 listIPMWorkflowFaults

Command Category: Diagnostic Commands
Use with WLST: Online

17.2.2.1 Description
List details on processing failures that have occurred during workflow agent processing.

17.2.2.2 Syntax

\[
\text{listIPMWorkflowFaults}(\text{startDate}, \text{endDate}, \text{appId}, \text{batchId})
\]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>Optional. The start of the date range for which error details should be repaired, in yyyy-MM-dd format.</td>
</tr>
</tbody>
</table>
17.2.2.3 Example
The following example lists the faults within the specified parameters.

```
listIPMWorkflowFaults(startDate="2009-06-01", endDate="2009-06-02")
listIPMWorkflowFaults(appId=3)
listIPMWorkflowFaults(batchId=15)
listIPMWorkflowFaults(startDate="2009-06-01", endDate="2009-06-02", appId=3)
```

17.2.3 repairIPMWorkflowFaults
Command Category: Diagnostic Commands
Use with WLST: Online

17.2.3.1 Description
Repair processing failures that have occurred during workflow agent processing.

17.2.3.2 Syntax
```
repairIPMWorkflowFaults([startDate], [endDate], [appId], [batchId])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>startDate</td>
<td>Optional. The start of the date range for which error details should be repaired, in yyyy-MM-dd format.</td>
</tr>
<tr>
<td>endDate</td>
<td>Optional. The end of the date range for which error details should be repaired, in yyyy-MM-dd format.</td>
</tr>
<tr>
<td>appId</td>
<td>Optional. The application ID for which error details should be repaired.</td>
</tr>
<tr>
<td>batchId</td>
<td>Optional. The batch ID for which error details should be repaired.</td>
</tr>
</tbody>
</table>

17.2.3.3 Example
The following example repairs the faults within the specified parameters.

```
repairIPMWorkflowFaults(startDate="2009-06-01", endDate="2009-06-02")
repairIPMWorkflowFaults(appId=3)
repairIPMWorkflowFaults(batchId=15)
repairIPMWorkflowFaults(startDate="2009-06-01", endDate="2009-06-02", appId=3)
```

17.2.4 sumIPMWorkflowFaults
Command Category: Diagnostic Commands
Use with WLST: Online
17.2.4.1 Description
Provides a count of processing failures that have occurred during workflow agent processing. The results are grouped by date, application ID, or batch ID.

17.2.4.2 Syntax
sumIPMWorkflowFaults(group)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>groupOption</td>
<td>Required. One of the following:</td>
</tr>
<tr>
<td></td>
<td>■ DATE: Returns fault counts grouped by date.</td>
</tr>
<tr>
<td></td>
<td>■ APPID: Returns fault counts grouped by application ID.</td>
</tr>
<tr>
<td></td>
<td>■ BATCHID: Returns fault counts grouped by batch ID.</td>
</tr>
</tbody>
</table>

17.2.4.3 Example
The following example returns all workflow faults grouped first by date, then by applications ID, then again grouped by batch ID.

sumIPMWorkflowFaults(group='DATE')
sumIPMWorkflowFaults(group='APPID')
sumIPMWorkflowFaults(group='BATCHID')

17.2.5 resetIpmDMSMetrics
Command Category: Diagnostic Commands
Use with WLST: Online

17.2.5.1 Description
Resets all Dynamic Monitoring Server (DMS) metrics associated with I/PM to zero. This is generally done if the administrator finds that historical performance data is skewing the current results.

17.2.5.2 Syntax
resetIpmDMSMetrics()

17.2.5.3 Example
The following example resets all DMS metrics to zero.

resetIpmDMSMetrics()

17.3 Imaging Configuration Commands
Use the Imaging configuration commands, listed in Table 17–3, to list and set configuration values specific to Imaging.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>createIPMConnection</td>
<td>Creates a new Imaging connection from a connection definition file.</td>
<td>Online</td>
</tr>
</tbody>
</table>
17.3.1 createIPMConnection

Command Category: Imaging Configuration Commands
Use with WLST: Online

17.3.1.1 Description
Creates a new Imaging connection from a connection definition file. The connection definition file is an XML file that describes a single Imaging connection definition using the Connection element type from the Imaging ConnectionService web services API schema definition. This schema is available from a running Imaging server using the following URL:

http://ipm_server_machine:ipm_server_port/imaging/ws/ConnectionService?xsd=1

For more information about the connection definition file format, see the Oracle Fusion Middleware Administering Oracle WebCenter Content: Imaging.

17.3.1.2 Syntax

```
createIPMConnection(connectionFile)
```

**Argument** | **Definition**
---|---
connectionFile | Required. A full path to the connection definition file's location on the Imaging server Node. Must be enclosed in single or double quotes.

17.3.1.3 Example
The following example creates a connection based on the specified attribute.

```
createIPMConnection(connectionFile="/home/ipmuser/localCSConnection.xml")
```
17.3.2 getIPMConfig

Command Category: Imaging Configuration Commands

Use with WLST: Online

17.3.2.1 Description

Gets an Imaging configuration setting value. The command is equivalent to browsing
the custom mbean hierarchy to the Imaging config mbean and using the standard
WLST set command to set an mbean attribute.

17.3.2.2 Syntax

getIPMConfig(attrName)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>attrName</td>
<td>Required. Name of the attribute to be read. Must be enclosed in single</td>
</tr>
<tr>
<td></td>
<td>or double quotes.</td>
</tr>
</tbody>
</table>

17.3.2.3 Example

The following example returns the value for the specified attribute names.

getIPMConfig('AgentUser')
getIPMConfig('CheckInterval')

17.3.3 grantIPMCreAccess

Command Category: Imaging Configuration Commands

Use with WLST: Online

17.3.3.1 Description

Grants CredentialAccessPermissions to Imaging so that it can read credentials from
the credential store. This command is required in configurations where Imaging
managed servers are in a separate domain home from the Administration Server.
When at least one Imaging managed server is in the same domain home as the
Administration Server, this command is not required, as CredentialAccessPermissions
are granted during Imaging startup.

When the Imaging Managed Server is not in the same domain home as the
Administration Server, however, the Imaging startup grant only affects the local
settings. Local settings get overwritten when the Administration Server synchronizes
its copy as the domain wide configuration, so this command updates the
Administration Server configuration such that permissions are distributed correctly to
all domain nodes.

17.3.3.2 Syntax

grantIPMCreAccess()

17.3.3.3 Example

The following example returns a list of all Imaging configuration mbeans.

grantIPMCreAccess()
### 17.3.4 importIPMApplication

**Command Category:** Imaging Configuration Commands  
**Use with WLST:** Online

#### 17.3.4.1 Description
Imports an application definition from a previously exported definition file.

#### 17.3.4.2 Syntax

```
importIPMApplication(exportFile, action, name, repository, securityOption, securityMember, docSecurityOption, docSecurityGroup, storageOption, storageVolume)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>exportFile</td>
<td>Required. A full path to the export definition file's location on the Imaging server node. Must be enclosed in single or double quotes.</td>
</tr>
<tr>
<td>action</td>
<td>Required. The action to be performed. Available actions are:</td>
</tr>
<tr>
<td></td>
<td>- <strong>Add</strong>: Creates a new input. Fails if an application with the same name already exists.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Update</strong>: Modifies and existing input. Fails if an application with the same name does not exist.</td>
</tr>
<tr>
<td></td>
<td>- <strong>AddOrUpdate</strong>: Creates a new application if it does not already exist or updates one that does.</td>
</tr>
<tr>
<td>name</td>
<td>Required. The name of the application being imported from the exported definitions file.</td>
</tr>
<tr>
<td>repository</td>
<td>The name of the repository in which to create the application. Required when adding an application, ignored when updating or modifying an application.</td>
</tr>
<tr>
<td>securityOption</td>
<td>Optional. Specifies how to define security for the imported application as follows:</td>
</tr>
<tr>
<td></td>
<td>- <strong>Existing</strong>: Uses application security as defined in the existing definition. Valid only for an update action.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Imported</strong>: Attempts to use application security as defined in the import file. Fails if any members defined in the import file are invalid.</td>
</tr>
<tr>
<td></td>
<td>- <strong>ValidOnly</strong>: Uses application security as defined in the import file and filters out any invalid members.</td>
</tr>
<tr>
<td></td>
<td>- <strong>CurrentUser</strong>: Sets full permissions to the user used to connect to the server.</td>
</tr>
<tr>
<td></td>
<td>- <strong>User</strong>: Sets full permissions to the user name provided in the securityMember parameter.</td>
</tr>
<tr>
<td></td>
<td>- <strong>Group</strong>: Sets full permissions to the group name provided in the securityMember parameter.</td>
</tr>
<tr>
<td>securityMember</td>
<td>Name of the user or group given full permissions to the application. Valid only when securityOption is set to either <strong>User</strong> or <strong>Group</strong>, otherwise it is ignored.</td>
</tr>
</tbody>
</table>
### 17.3.4.3 Example
The following example updates an existing application named *Invoices*. Note that the repository is listed as *None* because the update action uses the repository specified in the original application.

```java
importIPApplication(exportFile="/home/ipmuser/exportdefinitions.xml",
action="Update", name="Invoices", repository=None, securityOption="Existing")
```

### 17.3.4.4 Example
The following example creates a new application named *Receipts*. Note that the repository is explicitly named because the add action requires a valid repository be named.

```java
importIPApplication(exportFile="/home/ipmuser/exportdefinitions.xml",
action="Add", name="Receipts", repository="LocalCS", securityOption="ValidOnly")
```

### 17.3.5 importIPInput
Command Category: Imaging Configuration Commands
Use with WLST: Online

#### 17.3.5.1 Description
Imports an input definition from a previously exported definition file.
17.3.5.2 Syntax

```python
importIPMInput(exportFile, action, name, securityOption, securityMember)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>exportFile</code></td>
<td>Required. A full path to the export definition file's location on the Imaging server node. Must be enclosed in single or double quotes.</td>
</tr>
<tr>
<td><code>action</code></td>
<td>Required. The action to be performed. Available actions are:</td>
</tr>
<tr>
<td></td>
<td>■ <strong>Add</strong>: Creates a new input. Fails if an input with the same name already exists.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>Update</strong>: Modifies an existing input. Fails if an input with the same name does not exist.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>AddOrUpdate</strong>: Creates a new application if it does not already exist or updates one that does.</td>
</tr>
<tr>
<td><code>name</code></td>
<td>Required. The name of the input being imported from the exported definitions file.</td>
</tr>
<tr>
<td><code>repository</code></td>
<td>The name of the repository in which to create the application. Required when adding an application, ignored when updating or modifying an application.</td>
</tr>
<tr>
<td><code>securityOption</code></td>
<td>Optional. Specifies how to define security for the imported application as follows:</td>
</tr>
<tr>
<td></td>
<td>■ <strong>Existing</strong>: Uses input security as defined in the existing definition. Valid only for an update action.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>Imported</strong>: Attempts to use input security as defined in the import file. Fails if any members defined in the import file are invalid.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>ValidOnly</strong>: Uses input security as defined in the import file and filters out any invalid members.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>CurrentUser</strong>: Sets full permissions to the user used to connect to the server.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>User</strong>: Sets full permissions to the user name provided in the securityMember parameter.</td>
</tr>
<tr>
<td></td>
<td>■ <strong>Group</strong>: Sets full permissions to the group name provided in the securityMember parameter.</td>
</tr>
<tr>
<td><code>securityMember</code></td>
<td>Name of the user or group given full permissions to the input. Valid only when securityOption is set to either User or Group, otherwise it is ignored.</td>
</tr>
</tbody>
</table>

17.3.5.3 Example

The following example updates an existing input named *Invoices*. Note that the repository is listed as *None* because the update action uses the repository specified in the original application.

```python
importIPMInput(exportFile="/home/ipmuser/exportdefinitions.xml", action="Update", name="Invoices", securityOption="Existing")
```

17.3.5.4 Example

The following example creates a new input named *Receipts*. Note that the repository is explicitly named because the add action requires a valid repository be named.

```python
importIPMInput(exportFile="/home/ipmuser/exportdefinitions.xml", action="Add", name="Receipts", securityOption="ValidOnly")
```
17.3.6  importIPMSearch

Command Category: Imaging Configuration Commands
Use with WLST: Online

17.3.6.1 Description
Import a search definition from a previously exported definition file.

17.3.6.2 Syntax
importIPMSearch(exportFile, action, name, securityOption, securityMember)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>exportFile</td>
<td>Required. A full path to the export definition file's location on the Imaging server node. Must be enclosed in single or double quotes.</td>
</tr>
<tr>
<td>action</td>
<td>Required. The action to be performed. Available actions are:</td>
</tr>
<tr>
<td></td>
<td>■ Add: Creates a new search. Fails if a search with the same name already exists.</td>
</tr>
<tr>
<td></td>
<td>■ Update: Modifies an existing search. Fails if a search with the same name does not exist.</td>
</tr>
<tr>
<td></td>
<td>■ AddOrUpdate: Creates a new search if it does not already exist or updates one that does.</td>
</tr>
<tr>
<td>name</td>
<td>Required. The name of the search being imported from the exported definitions file.</td>
</tr>
<tr>
<td>repository</td>
<td>The name of the repository in which to create the application. Required when adding an application, ignored when updating or modifying an application.</td>
</tr>
<tr>
<td>securityOption</td>
<td>Optional. Specifies how to define security for the imported application as follows:</td>
</tr>
<tr>
<td></td>
<td>■ Existing: Uses search security as defined in the existing definition. Valid only for an update action.</td>
</tr>
<tr>
<td></td>
<td>■ Imported: Attempts to use search security as defined in the import file. Fails if any members defined in the import file are invalid.</td>
</tr>
<tr>
<td></td>
<td>■ ValidOnly: Uses search security as defined in the import file and filters out any invalid members.</td>
</tr>
<tr>
<td></td>
<td>■CurrentUser: Sets full permissions to the user used to connect to the server.</td>
</tr>
<tr>
<td></td>
<td>■ User: Sets full permissions to the user name provided in the securityMember parameter.</td>
</tr>
<tr>
<td></td>
<td>■ Group: Sets full permissions to the group name provided in the securityMember parameter.</td>
</tr>
<tr>
<td>securityMember</td>
<td>Name of the user or group given full permissions to the search. Valid only when securityOption is set to either User or Group, otherwise it is ignored.</td>
</tr>
</tbody>
</table>

17.3.6.3 Example
The following example updates an existing search named Invoices. Note that the repository is listed as None because the update action uses the repository specified in the original application.
importIPMSearch(exportFile="/home/ipmuser/exportdefinitions.xml", action="Update", name="Invoices", securityOption="Existing")

17.3.6.4 Example
The following example creates a new search named Receipts. Note that the repository is explicitly named because the add action requires a valid repository be named.
importIPMSearch(exportFile="/home/ipmuser/exportdefinitions.xml", action="Add", name="Receipts", securityOption="ValidOnly")

17.3.7 listIPMConfig
Command Category: Imaging Configuration Commands
Use with WLST: Online

17.3.7.1 Description
Provides a listing of imaging configuration mbeans. The command is equivalent to browsing the custom mbean hierarchy and listing the Imaging mbean attributes.

17.3.7.2 Syntax
listIPMConfig()

17.3.7.3 Example
The following example returns a list of all Imaging configuration mbeans.
listIPMConfig()

17.3.8 listIPMExportFile
Command Category: Imaging Configuration Commands
Use with WLST: Online

17.3.8.1 Description
Lists the contents of an exported Imaging definitions file.

17.3.8.2 Syntax
listIPMExportFile(exportFile="<path to file>")

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>exportFile</td>
<td>Required. A full path to the export definition file’s location on the Imaging server node. Must be enclosed in single or double quotes.</td>
</tr>
</tbody>
</table>

17.3.8.3 Example
The following example returns the contents of an Imaging definitions file.
lisIPMExportFile(exportFile="/home/ipmuser/exportdefinitions.xml")

17.3.9 refreshIPMSecurity
Command Category: Imaging Configuration Commands
Use with WLST: Online

17.3.9.1 Description
Refreshes security items currently stored in the Imaging database. This is typically done when migrating security to a different policy store and only updates security items found in the new policy store.

17.3.9.2 Syntax
refreshIPMSecurity()

17.3.9.3 Example
The following example refreshes the security items stored in the Imaging database.
refreshIPMSecurity()

17.3.10 setIPMConfig
Command Category: Imaging Configuration Commands
Use with WLST: Online

17.3.10.1 Description
Sets an Imaging configuration setting value. The command is equivalent to browsing the custom mbean hierarchy to the Imaging config mbean and using the standard WLST 'set' command to set an mbean attribute.

17.3.10.2 Syntax
setIPMConfig(attrName, value)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>attrName</td>
<td>Required. Name of the attribute to be set. Must be enclosed in single or double quotes.</td>
</tr>
<tr>
<td>value</td>
<td>Required. Value of the attribute to be set. Only enclosed in single or double quotes if value is a string literal.</td>
</tr>
</tbody>
</table>

17.3.10.3 Example
The following example sets the specified values for the specified attribute names.

setIPMConfig('AgentUser', 'agentadmin')
setIPMConfig('CheckInterval', 30)

17.3.11 submitIPMToWorkflow
Command Category: Imaging Configuration Commands
Use with WLST: Online

17.3.11.1 Description
Submits a document to the workflow agent. Note that a confirmation message is displayed stating that the document has been submitted, however if the document is stored in an application that is not configured with a workflow, no action is taken.
17.3.11.2 Syntax

submitIPMToWorkflow(documentId)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>documentId</td>
<td>Required. The unique document ID of the submitted document.</td>
</tr>
</tbody>
</table>

17.3.11.3 Example

The following example submits a document to a workflow.

submitIPMToWorkflow(documentId="2.IPM_12345")
Oracle Business Process Management Custom WLST Commands

This chapter provides detailed descriptions of custom WLST commands for Business Process Management (BPM), including command syntax, arguments and command examples.

This chapter lists and describes the custom WLST commands for Oracle Business Process Management.

18.1 BPMLifecycleAdmin Command Group

Table 18–1 lists and describes the BPMLifecycleAdmin commands for project lifecycle administration.

<table>
<thead>
<tr>
<th>Use this command…</th>
<th>To…</th>
<th>Use with WLST…</th>
</tr>
</thead>
<tbody>
<tr>
<td>create_public_share</td>
<td>Create a public share.</td>
<td>Offline</td>
</tr>
<tr>
<td>unlock_public_share</td>
<td>Unlock a public share.</td>
<td>Offline</td>
</tr>
<tr>
<td>export_public_share</td>
<td>Export a public share to the file system.</td>
<td>Offline</td>
</tr>
<tr>
<td>delete_public_share</td>
<td>Delete a public share.</td>
<td>Offline</td>
</tr>
<tr>
<td>publish_template</td>
<td>Publish a template to MDS.</td>
<td>Offline</td>
</tr>
<tr>
<td>export_template</td>
<td>Export a template to the file system.</td>
<td>Offline</td>
</tr>
<tr>
<td>delete_template</td>
<td>Delete a template from MDS.</td>
<td>Offline</td>
</tr>
<tr>
<td>connectProjectRepository</td>
<td>Establish the connection to the MDS project repository.</td>
<td>Offline</td>
</tr>
<tr>
<td>disconnectProjectRepository</td>
<td>Close the connection from MDS project repository.</td>
<td>Offline</td>
</tr>
<tr>
<td>importProject</td>
<td>Import a project from disk into the project repository.</td>
<td>Offline</td>
</tr>
<tr>
<td>createProject</td>
<td>Create a public share project based on a Template project.</td>
<td>Offline</td>
</tr>
<tr>
<td>deleteProject</td>
<td>Delete a project from the project repository.</td>
<td>Offline</td>
</tr>
<tr>
<td>exportProject</td>
<td>Export a project stored in the repository to disk.</td>
<td>Offline</td>
</tr>
<tr>
<td>unlockProject</td>
<td>Unlock a project.</td>
<td>Offline</td>
</tr>
</tbody>
</table>
18.1.1 create_public_share

Command Category: BPMLifecycleAdmin Commands
Use with WLST: Offline

18.1.1.1 Description
Use this command to create a public share from a template. The template must exist in MDS.

18.1.1.2 Syntax
create_public_share(composerUser, composerPassword, connectionURL, templateName, publicshareId, mdsconfigLocation, [Override], [oracleHome])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>composerUser</td>
<td>The Business Process Composer user who performs the current operation.</td>
</tr>
<tr>
<td>composerPassword</td>
<td>BPM Composer user’s password</td>
</tr>
<tr>
<td>connectionURL</td>
<td>JNDI connection URL to the security server service in format host:port</td>
</tr>
<tr>
<td>templateName</td>
<td>Name of the template in MDS</td>
</tr>
<tr>
<td>publicshareId</td>
<td>Name of the public share to be created</td>
</tr>
<tr>
<td>mdsconfigLocation</td>
<td>Location of the mds-config.xml to be used to connect to MDS</td>
</tr>
<tr>
<td>projectLocation</td>
<td>The path where the public share will be created. If the path does not exist, it will be created. The root is '/'.</td>
</tr>
<tr>
<td>Override</td>
<td>Enables you to override the public share if a public share exists in MDS with the same name. The template is not overwritten when you execute this command.</td>
</tr>
<tr>
<td>oracleHome</td>
<td>Optional. The Oracle home to be used.</td>
</tr>
</tbody>
</table>

18.1.1.3 Examples
The following example creates a public share named Sample_PublicShare. It is based on the template with name Sample_Template. The name of the public share is Sample_PublicShare, and the location of the mds-config.xml file is /tmp/mds-config.xml.
create_public_share('user_name', 'password', 'host:port', 'Sample_Template', 'Sample_PublicShare', '/tmp/mds-config.xml')

The following example creates a public share named Sample_PublicShare. It is based on the template named Sample_Template that exists in MDS. The public share, not the template, is overridden. The location of the mds-config.xml file is /tmp/mds-config.xml.
create_public_share('user_name', 'password', 'host:port', 'Sample_Template', 'Sample_PublicShare', '/tmp/mds-config.xml', 'true')

18.1.2 unlock_public_share

Command Category: BPMLifecycleAdmin Commands
Use with WLST: Offline
18.1.2.1 Description
Use this command to unlock a public share. For example, when you create a project by using the Ant task `create_public_share` command, the project is created as locked. You can then unlock it by using the `unlock_public_share` command.

A lock is also set by enabling or disabling the check box `enable sharing` in the project creation page in Oracle Business Process Composer.

It is also released when the user publishes a project from Business Process Composer.

The public share must exist in MDS.

18.1.2.2 Syntax
```
unlock_public_share(composerUser, composerPassword, connectionURL, publicshareId, mdsconfigLocation, [oracleHome] )
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>composerUser</td>
<td>The Business Process Composer user who performs the current operation.</td>
</tr>
<tr>
<td>composerPassword</td>
<td>BPM Composer user's password</td>
</tr>
<tr>
<td>connectionURL</td>
<td>JNDI connection URL to the security server service in format <code>host:port</code></td>
</tr>
<tr>
<td>publicshareId</td>
<td>Name of the public share to be unlocked</td>
</tr>
<tr>
<td>mdsconfigLocation</td>
<td>Location of the <code>mds-config.xml</code> to be used to connect to MDS</td>
</tr>
<tr>
<td>oracleHome</td>
<td>Optional. The Oracle home to be used</td>
</tr>
</tbody>
</table>

18.1.2.3 Example
The following example unlocks a public share named `Sample_PublicShare`. The location of the `mds-config.xml` file is `/tmp/mds-config.xml`.

```
unlock_public_share('user_name', 'password', 'host:port', 'Sample_PublicShare', '/tmp/mds-config.xml')
```

18.1.3 export_public_share
Command Category: BPMLifecycleAdmin Commands
Use with WLST: Offline

18.1.3.1 Description
Use this command to export the public share from MDS to the file system.

18.1.3.2 Syntax
```
export_public_share(composerUser, composerPassword, connectionURL, publicshareId, fsLocation, mdsconfigLocation, [oracleHome] )
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>composerUser</td>
<td>The Business Process Composer user who performs the current operation.</td>
</tr>
<tr>
<td>composerPassword</td>
<td>BPM Composer user's password</td>
</tr>
<tr>
<td>connectionURL</td>
<td>JNDI connection URL to the security server service in format <code>host:port</code></td>
</tr>
</tbody>
</table>
18.1.3.3 Example

The following example specifies the public share name as Sample_PublicShare, the file system location as /tmp, and the location of the mds-config.xml file as /tmp/mds-config.xml.

```bash
export_public_share('user_name', 'password', 'host:port', 'Sample_PublicShare', '/tmp', '/tmp/mds-config.xml')
```

18.1.4 delete_public_share

Command Category: BPMLifecycleAdmin Commands

Use with WLST: Offline

18.1.4.1 Description

Use this command to delete a public share from MDS. Executing this command requires that the public share is not locked.

18.1.4.2 Syntax

```bash
delete_public_share(composerUser, composerPassword, connectionURL, publicshareId, mdsconfigLocation, [releaseLock], [oracleHome])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>composerUser</td>
<td>The Business Process Composer user who performs the current operation.</td>
</tr>
<tr>
<td>composerPassword</td>
<td>BPM Composer user’s password</td>
</tr>
<tr>
<td>connectionURL</td>
<td>JNDI connection URL to the security server service in format host:port</td>
</tr>
<tr>
<td>publicshareId</td>
<td>Name of the public share to be deleted</td>
</tr>
<tr>
<td>mdsconfigLocation</td>
<td>Location of the mds-config.xml to be used to connect to MDS</td>
</tr>
<tr>
<td>releaseLock</td>
<td>Optional. If the public share is locked, this lock can be released and the delete operation completed. You can set this attribute to either true or false. If not specified, default value is false.</td>
</tr>
<tr>
<td>oracleHome</td>
<td>Optional. The Oracle home to be used</td>
</tr>
</tbody>
</table>

18.1.4.3 Examples

The following example specifies the name and location of a public share to be deleted.

```bash
delete_public_share('Sample_PublicShare', '/tmp/mds-config.xml')
```

The following example specifies the name and location of a public share to be deleted, and that the public share should be deleted even if locked.

```bash
delete_public_share('user_name', 'password', 'host:port', 'Sample_PublicShare', '/tmp/mds-config.xml', 'true')
```
18.1.5 publish_template

Command Category: BPMLifecycleAdmin Commands

Use with WLST: Offline

18.1.5.1 Description

Use this command to publish the template from the file system to MDS.

18.1.5.2 Syntax

```
publish_template(composerUser, composerPassword, connectionURL, templateName, fsLocation, mdsconfigLocation, [Override], [oracleHome] )
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>composerUser</td>
<td>The Business Process Composer user who performs the current operation.</td>
</tr>
<tr>
<td>composerPassword</td>
<td>BPM Composer user's password</td>
</tr>
<tr>
<td>connectionURL</td>
<td>JNDI connection URL to the security server service in format host:port</td>
</tr>
<tr>
<td>templateName</td>
<td>Name of the template to be published</td>
</tr>
<tr>
<td>fsLocation</td>
<td>File system location of the template project</td>
</tr>
<tr>
<td>mdsconfigLocation</td>
<td>Location of the mds-config.xml to be used to connect to MDS</td>
</tr>
<tr>
<td>projectLocation</td>
<td>The path where the public share will be created. If the path does not exist it will be created. The root is '/'.</td>
</tr>
<tr>
<td>Override</td>
<td>When you publish a template in MDS, this attribute enables you to override an existing template with the same name. Can either be 'true' or 'false'. If not specified, default value is 'false'.</td>
</tr>
<tr>
<td>oracleHome</td>
<td>Optional. The Oracle home to be used</td>
</tr>
</tbody>
</table>

18.1.5.3 Example

The following example publishes a template named `Sample_Template_Name_MDS` to the root folder.

```
f('user_name', 'password', 'host:port','Sample_Template','/tmp/MyTemplate','/','/tmp/mds-config.xml')
```

The following example publishes a template named `Sample_Template_Name_MDS` to the `/WorkingOn/` folder.

```
publish_template('user_name', 'password', 'host:port','Sample_Template','/tmp/MyTemplate','/WorkingOn', '/tmp/mds-config.xml')
```

18.1.6 export_template

Command Category: BPMLifecycleAdmin Commands

Use with WLST: Offline
### 18.1.6.1 Description
Use this command to export the template from MDS to the file system.

### 18.1.6.2 Syntax

```bash
export_template(composerUser, composerPassword, connectionURL, templateName, fsLocation, mdsconfigLocation, [oracleHome] )
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>composerUser</td>
<td>The Business Process Composer user who performs the current operation.</td>
</tr>
<tr>
<td>composerPassword</td>
<td>BPM Composer user's password</td>
</tr>
<tr>
<td>connectionURL</td>
<td>JNDI connection URL to the security server service in format host:port</td>
</tr>
<tr>
<td>templateName</td>
<td>Name of the template to be exported</td>
</tr>
<tr>
<td>fsLocation</td>
<td>File system location where the project is to be downloaded</td>
</tr>
<tr>
<td>mdsconfigLocation</td>
<td>Location of the mds-config.xml to be used to connect to MDS</td>
</tr>
<tr>
<td>oracleHome</td>
<td>Optional. The Oracle home to be used</td>
</tr>
</tbody>
</table>

### 18.1.6.3 Example

The following example specifies the template name as Sample_Template, the file system location as /tmp, and the location of the mds-config.xml file as /tmp/mds-config.xml.

```bash
export_template('user_name', 'password', 'host:port','Sample_Template','/tmp','/tmp/mds-config.xml')
```

### 18.1.7 delete_template

Command Category: BPMLifecycleAdmin Commands

Use with WLST: Offline

#### 18.1.7.1 Description
Use this command to delete the template from MDS.

#### 18.1.7.2 Syntax

```bash
delete_template(composerUser, composerPassword, connectionURL, templateName, mdsconfigLocation, [oracleHome] )
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>composerUser</td>
<td>The Business Process Composer user who performs the current operation.</td>
</tr>
<tr>
<td>composerPassword</td>
<td>BPM Composer user's password</td>
</tr>
<tr>
<td>connectionURL</td>
<td>JNDI connection URL to the security server service in format host:port</td>
</tr>
<tr>
<td>templateName</td>
<td>Name of the template to be deleted</td>
</tr>
<tr>
<td>fsLocation</td>
<td>File system location of the template project</td>
</tr>
<tr>
<td>mdsconfigLocation</td>
<td>Location of the mds-config.xml to be used to connect to MDS</td>
</tr>
</tbody>
</table>
18.1.7.3 Example
The following example deletes the template named Sample_template from MDS.

```java
delete_template('weblogic', 'welcome1', 'host:port', '/Sample_template', '/tmp/mds-config.xml')
```

18.1.8 connectProjectRepository

Command Category: BPMLifecycleAdmin Commands

Use with WLST: Offline

18.1.8.1 Description

Use this command to establish the connection to the MDS project repository.

18.1.8.2 Syntax

```java
connectProjectRepository(connectionURL, mdsConfigLocation, composerUser, composerPassword, [mdsDbUser], [mdsDbPassword])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>connectionURL</td>
<td>JNDI connection URL to the security server service in format: host:port</td>
</tr>
<tr>
<td>mdsConfigLocation</td>
<td>The location of the mds-config.xml to be used to connect to MDS</td>
</tr>
<tr>
<td>composerUser</td>
<td>BPM Composer user. This user will perform the current operation.</td>
</tr>
<tr>
<td>composerPassword</td>
<td>BPM Composer user's password.</td>
</tr>
<tr>
<td>mdsDbUser</td>
<td>Optional. Overrides the mds database username stored in the mds-config.xml file.</td>
</tr>
<tr>
<td>mdsDbPassword</td>
<td>Optional. Overrides the MDS database password stored in the mds-config.xml file.</td>
</tr>
</tbody>
</table>

18.1.8.3 Examples

```java
connectProjectRepository('localhost:7001', '/home/user/mds-config.xml', 'weblogic', 'welcome1')
connectProjectRepository('localhost:7001', '/home/user/mds-config.xml', 'weblogic', 'welcome1', 'DEV_MDS', 'dbPassword')
```

18.1.9 disconnectProjectRepository

Command Category: BPMLifecycleAdmin Commands

Use with WLST: Offline
18.1.9.1 Description
Use this command to close the connection from MDS project repository.

18.1.9.2 Syntax
disconnectProjectRepository()

18.1.9.3 Example
disconnectProjectRepository()

18.1.10 importProject
Command Category: BPMLifecycleAdmin Commands
Use with WLST: Offline

18.1.10.1 Description
Use this command to import a project from disk into the project repository. All projects will be imported as a template—that is, it does not matter if the imported project is a template or public shared.

18.1.10.2 Syntax
importProject(srcDir, toName, [toPath], [override])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>srcDir</td>
<td>The file system location of the template project.</td>
</tr>
<tr>
<td>toName</td>
<td>The file system location of the template project.</td>
</tr>
<tr>
<td>toPath</td>
<td>[Optional] The path where the template will be imported. If the path does not exist it will be created. If not specified, the root '/' will be used.</td>
</tr>
<tr>
<td>override</td>
<td>[Optional] Specifies whether the import should override an existing template with the same name. If not specified, 'false' will be used.</td>
</tr>
</tbody>
</table>

18.1.10.3 Examples
importProject(srcDir='/home/user/myFolder/myTemplate', toName='NewTemplateName')

importProject(srcDir='/home/user/myFolder/myTemplate', toName='NewTemplateName', toPath='/NewFolder', override='true')

18.1.10.4 Security Constraints
None. All users are allowed to import a project template. These templates are imported by default with public permissions granted to all users.

18.1.11 createProject
Command Category: BPMLifecycleAdmin Commands
Use with WLST: Offline

18.1.11.1 Description
Use this command to create a public share project based on a template project.
18.1.11.2 Syntax
createProject(fromId, toName, [toPath])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>fromId</td>
<td>The identifier of the template in MDS.</td>
</tr>
<tr>
<td>toName</td>
<td>The name of the public share to be created.</td>
</tr>
<tr>
<td>toPath</td>
<td>[Optional] The path where the public share will be created. If the path does not exist it will be created. If not specified, the root ‘/’ will be used.</td>
</tr>
</tbody>
</table>

18.1.11.3 Example
createProject(fromId='/SampleTemplate', toName='SamplePublicShare')
createProject(fromId='/folder/SampleTemplate', toName='SamplePublicShare', toPath='/newFolder')

18.1.11.4 Security Constraints
None. All the users are allowed to create public shared based on the templates. All the templates in MDS are public by default.

18.1.12 deleteProject
Command Category: BPMLifecycleAdmin Commands
Use with WLST: Offline

18.1.12.1 Description
Use this command to delete a project from the project repository.

18.1.12.2 Syntax
deleteProject(id, [type], [breakLinks])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>The identifier of the project to be deleted.</td>
</tr>
<tr>
<td>type</td>
<td>[Optional] Specified if the project is a template or a public share. Valid values: 'TEMPLATE', 'PUBLIC_SHARED'. If not specified, 'TEMPLATE' is assumed.</td>
</tr>
<tr>
<td>breakLinks</td>
<td>[Optional] Only valid when deleting templates. If there are public shares linked to this template, specifies whether the links should be removed or not. Default: 'true'</td>
</tr>
</tbody>
</table>

18.1.12.3 Example
deleteProject(id='/SampleTemplate', type='TEMPLATE', breakLinks='true')
deleteProject(id='folder/SamplePublicShare', type='PUBLIC_SHARED')

18.1.12.4 Security Constraints
ADMIN | OWNER
To perform a delete operation, the user must have the role of either ADMIN or OWNER of the project to delete.
18.1.13 exportProject

Command Category: BPMLifecycleAdmin Commands
Use with WLST: Offline

18.1.13.1 Description
Use this command to export a project stored in the repository to disk.

18.1.13.2 Syntax
exportProject(id, toDir, [type])

18.1.13.3 Examples
exportProject(id='/SampleTemplate', toDir='/home/user/myFolder', type='TEMPLATE')

exportProject(id='/aFolder/SamplePublicShare', toDir='/home/user/myFolder', type='PUBLIC_SHARED')

18.1.13.4 Security Constraints
ADMIN | OWNER | EDITOR | VIEWER
The user needs at least VIEWER permission of the project to be exported.

18.1.14 unlockProject

Command Category: BPMLifecycleAdmin Commands
Use with WLST: Offline

18.1.14.1 Description
Unlock a project. Only public share projects can be unlocked.

18.1.14.2 Syntax
unlockProject(id)

18.1.14.3 Example
unlockProject(id='SamplePublicShare')

18.1.14.4 Security Constraints
ADMIN | OWNER | EDITOR
A project can be unlocked only by the same user or application that locked it. The exception is the administrator user who can unlock a project regardless of the user or application that locked it.

An application can be Oracle Business Process Composer, Oracle Business Process Management Studio, or a generic application. WLST functions are considered generic applications.

**Note:** Because there is no WLST lock operation for projects, an owner or editor user cannot release a project by using the WLST unlockProject command if the project was locked by a different application. In this case, only an administrator user can unlock the project.
This chapter provides detailed descriptions of custom WLST commands for Oracle WebCenter Content, including command syntax, arguments and command examples.

The following sections describe the custom WLST commands for Oracle WebCenter Content. These commands enable you to configure and monitor the Oracle WebCenter Content server and the Oracle WebCenter Content Server instance from the command line. Topics include:

- Section 19.1, "Overview of WLST WebCenter Content Command Categories"
- Section 19.2, "WLST WebCenter Content Help"
- Section 19.3, "Getter and Setter Methods Implementation"
- Section 19.4, "Server Configuration Commands"
- Section 19.5, "Email Configuration Commands"
- Section 19.6, "System Status Commands"
- Section 19.7, "General Configuration Commands"
- Section 19.8, "Content Security Configuration Commands"
- Section 19.9, "Component Manager Configuration Commands"
- Section 19.10, "Records Management Configuration Commands"
- Section 19.11, "User Interface Commands"
- Section 19.12, "User Interface Connection Commands"

For additional information about Oracle Webcenter Content and Content Server administration and configuration, see Oracle Fusion Middleware Administrator’s Guide.

**Note:** To use the Oracle Webcenter Content custom commands, you must invoke the WLST script from the Oracle Common home in which the component has been installed. See "Using Custom WLST Commands" in the Oracle Fusion Middleware Administrator’s Guide.

### 19.1 Overview of WLST WebCenter Content Command Categories

WLST WebCenter Content commands are divided into the following categories:
To view the WebCenter Content UCM commands that can be invoked from WLST, enter the following command at the WLST prompt:

```
help('UCM')
```

To view help for a specific Oracle Webcenter Content command, specify the name of the command; for example:

```
help('getUCMServerPort')
```

```
help('wccAdfConfig')
```

### 19.3 Getter and Setter Methods Implementation

The WLST component for Oracle Webcenter Content uses **getter** and **setter** methods to handle a situation where multiple applications register their corresponding Mbeans on a managed server, but WLST can talk to only one application.

#### Getter Method

The **getter** method is designed to handle zero or one argument.

If you do not provide an argument to an WLST WebCenter Content command, then one of two things occurs:

- If only one application has registered its Mbean on the server, then the WLST WebCenter Content command should work successfully and display the output.
- If multiple applications have registered Mbeans on the server, then an error message is displayed to prompt you to enter the specific application name in the argument.
If there is one argument to an WLST WebCenter Content command, then the following occurs:

- You must enter the correct application name when entering an argument. If the name is not entered properly, then an error message is displayed to prompt you to enter the valid application name in the argument.

**Setter Method**
The setter method is designed to handle one or two arguments.

- The first argument is the value to which you want to set the parameter.
- The second argument is the application name, which can be null or a string.

### 19.4 Server Configuration Commands

Use the commands in Table 19–2 to configure the Oracle WebCenter Content Server instance.

Before you use these custom commands, set up the initial WLST connection as follows:

1. Set the environment variable ORACLE_HOME to `<Middleware_Home>/Oracle_ECM1`.
2. Run the WLST script from the following location: `<middleware_home>/Oracle_ECM1/common/bin`.
3. Connect to the WebCenter Content Server instance using the `connect()` command, for example, `connect("weblogic","password","t3://localhost:16200")`.

#### Table 19–2  WLST Server Configuration Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>getUCMHttpServerAddress</td>
<td>Display the HTTP Server Address value.</td>
<td>Online</td>
</tr>
<tr>
<td>getUCMServerPort</td>
<td>Display the Intradoc Server Port configuration parameter.</td>
<td>Online</td>
</tr>
<tr>
<td>setUCMServerPort</td>
<td>Set the Intradoc Server Port configuration parameter.</td>
<td>Online</td>
</tr>
<tr>
<td>getUCMIpAddressFilter</td>
<td>Display the IP Address Filter value.</td>
<td>Online</td>
</tr>
<tr>
<td>setUCMIpAddressFilter</td>
<td>Set the IP Address Filter value.</td>
<td>Online</td>
</tr>
<tr>
<td>getUCMUseSSL</td>
<td>Display the Use SSL value.</td>
<td>Online</td>
</tr>
</tbody>
</table>

### 19.4.1 getUCMHttpServerAddress

Use with WLST: Online

#### 19.4.1.1 Description

Gets the HTTP Server Address value from the `config.cfg` file and displays it.

#### 19.4.1.2 Syntax

`getUCMHttpServerAddress(['appName'])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>
19.4.1.3 Example
The following command displays the Oracle Webcenter Content HTTP server address for the application "Oracle Universal Content Management - Content Server":

    getUCMHttpServerAddress('Oracle Universal Content Management - Content Server')
    server.example.com

19.4.2 getUCMServerPort
Use with WLST: Online

19.4.2.1 Description
Gets the Intradoc Server Port configuration parameter from the config.cfg file and displays it.

19.4.2.2 Syntax
getUCMServerPort([ appName ])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.4.2.3 Example
The following command displays the Intradoc Server Port value for the application "Oracle Universal Content Management - Content Server":

    getUCMServerPort('Oracle Universal Content Management - Content Server')
    4442

19.4.3 setUCMServerPort
Use with WLST: Online

19.4.3.1 Description
Sets the Server Port configuration parameter.

19.4.3.2 Syntax
setUCMServerPort(value, [ appName ])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Server Port number. This number must be a positive integer between 0 and 65535.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.4.3.3 Example
The following command sets the Server Port configuration parameter for the application "Oracle Universal Content Management - Content Server":

    setUCMServerPort(4442,'Oracle Universal Content Management - Content Server')

19.4.4 getUCMIpAddressFilter
Use with WLST: Online
19.4.4.1 Description
Gets the IP Address Filter value from the config.cfg file and displays it.

19.4.4.2 Syntax
getUCMIpAddressFilter(['appName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.4.4.3 Example
The following command displays the IP address filter value for the application "Oracle Universal Content Management - Content Server":

```
getUCMIpAddressFilter('Oracle Universal Content Management - Content Server')
10.131.123.*
```

19.4.5 setUCMIpAddressFilter
Use with WLST: Online

19.4.5.1 Description
Sets the Webcenter Content IP Address Filter value.

19.4.5.2 Syntax
setUCMIpAddressFilter(value, ['appName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>WebCenter Content IP Address Filter number. This number must be of &quot;<em>.</em>.<em>.</em>&quot; format or IPV6 Format. The value must be taken from a list of IP Addresses allowed to communicate with the Content Server instance through the Intradoc Server Port.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.4.5.3 Example
The following command sets the value for the WebCenter Content IP address filter for the application "Oracle Universal Content Management - Content Server":

```
setUCMIpAddressFilter(10.131.123.*, 'Oracle Universal Content Management - Content Server')
```

19.4.6 getUCMUseSSL
Use with WLST: Online

19.4.6.1 Description
Gets the Use SSL value from the config.cfg file and displays it. The value can be True or False.

19.4.6.2 Syntax
getUCMUseSSL(['appName'])
Email Configuration Commands

19.4.6.3 Example
The following command displays the Use SSL value for the application "Oracle Universal Content Management - Content Server":

```plaintext
getUCMUseSSL('Oracle Universal Content Management - Content Server')
True
```

19.5 Email Configuration Commands
Use the commands in Table 19–3 to configure email for the Oracle WebCenter Content Server instance.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>getUCMMailServer</td>
<td>Display the Mail Server value.</td>
<td>Online</td>
</tr>
<tr>
<td>setUCMMailServer</td>
<td>Set the Mail Server value.</td>
<td>Online</td>
</tr>
<tr>
<td>getUCMSmtpPort</td>
<td>Display the SMTP Port value.</td>
<td>Online</td>
</tr>
<tr>
<td>getUCMSysAdminAddr</td>
<td>Display the Admin Address value.</td>
<td>Online</td>
</tr>
<tr>
<td>setUCMSysAdminAddr</td>
<td>Set the Admin Address value.</td>
<td>Online</td>
</tr>
</tbody>
</table>

19.5.1 getUCMMailServer
Use with WLST: Online

19.5.1.1 Description
Gets the Mail Server value from the config.cfg file and displays it.

19.5.1.2 Syntax
getUCMMailServer(['appName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.5.1.3 Example
The following command displays the Mail Server value for the application "Oracle Universal Content Management - Content Server":

```plaintext
getUCMMailServer('Oracle Universal Content Management - Content Server')
mymailserver.example.com
```

19.5.2 setUCMMailServer
Use with WLST: Online
19.5.2.1 Description
Sets the Mail Server value in the config.cfg file.

19.5.2.2 Syntax
setUCMMailServer(value,['appName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Value for the Mail Server. The value is the name of the mail server that the Content Server instance uses to send SMTP based email.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.5.2.3 Example
The following command sets the value for the Mail Server for the application "Oracle Universal Content Management - Content Server":
setUCMMailServer(mymailserver.example.com,'Oracle Universal Content Management - Content Server')

19.5.3 getUCMSmtpPort
Use with WLST: Online

19.5.3.1 Description
Gets the SMTP Port value in the config.cfg file and displays it.

19.5.3.2 Syntax
getUCMSmtpPort(['appName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.5.3.3 Example
The following command displays the SMTP port value for the application "Oracle Universal Content Management - Content Server":
getUCMSmtpPort('Oracle Universal Content Management - Content Server')

4055

19.5.4 getUCMSysAdminAddress
Use with WLST: Online

19.5.4.1 Description
Gets the Admin Address value from the config.cfg file and displays it. The value can be of the form abc@xyz.def.

19.5.4.2 Syntax
getUCMSysAdminAddress(['appName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>
19.5.4.3 Example
The following command displays the Admin Address value for the application "Oracle Universal Content Management - Content Server":

```
getUCMSysAdminAddress('Oracle Universal Content Management - Content Server')
```

19.5.5 setUCMSysAdminAddress
Use with WLST: Online

19.5.5.1 Description
Sets the Admin Address value in the config.cfg file.

19.5.5.2 Syntax
```
setUCMSysAdminAddress(value, ['AppName'])
```

19.5.5.3 Example
The following command sets the Admin Address value for the application "Oracle Universal Content Management - Content Server":

```
setUCMSysAdminAddress(mymail@example.com,'Oracle Universal Content Management - Content Server')
```

19.6 System Status Commands
Use the commands in Table 19–4 to configure additional settings to monitor the WebCenter Content Server instance.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>getUCMCSVersion</td>
<td>Display the version number.</td>
<td>Online</td>
</tr>
<tr>
<td>getUCMServerUptime</td>
<td>Display the uptime value.</td>
<td>Online</td>
</tr>
</tbody>
</table>

19.6.1 getUCMCSVersion
Use with WLST: Online

19.6.1.1 Description
Gets the version number of the Content Server running instance.

19.6.1.2 Syntax
```
getUCMCSVersion(['appName'])
```
19.6.1.3 Example
The following command displays the version number of the active instance of the application "Oracle Universal Content Management - Content Server":

```java
getUCMCSVersion('Oracle Universal Content Management - Content Server')
```

11g R1

19.6.2 getUCMServerUptime
Use with WLST: Online

19.6.2.1 Description
Gets the amount of time the Content Server instance has been up.

19.6.2.2 Syntax

```java
getUCMServerUptime(['appName'])
```

19.6.2.3 Example
The following command displays the amount of time the application "Oracle Universal Content Management - Content Server" has been up:

```java
getUCMServerUptime('Oracle Universal Content Management - Content Server')
```

00H:01 Min:12 Sec

19.7 General Configuration Commands
Use the commands in Table 19–5 to configure general configuration options for the Oracle WebCenter Content Server instance.

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

Table 19–5   WLST General Configuration Options Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>getUCMOverRideFormat</td>
<td>Display the OverRideFormat value.</td>
<td>Online</td>
</tr>
<tr>
<td>setUCMOverRideFormat</td>
<td>Set the OverRideFormat value.</td>
<td>Online</td>
</tr>
<tr>
<td>getUCMDownloadApplet</td>
<td>Display the DownloadApplet value.</td>
<td>Online</td>
</tr>
<tr>
<td>setUCMDownloadApplet</td>
<td>Set the DownloadApplet value.</td>
<td>Online</td>
</tr>
<tr>
<td>getUCMMultiUpload</td>
<td>Display the MultiUpload value.</td>
<td>Online</td>
</tr>
<tr>
<td>setUCMMultiUpload</td>
<td>Set the MultiUpload value.</td>
<td>Online</td>
</tr>
<tr>
<td>getUCMUseAccounts</td>
<td>Display the UseAccount value.</td>
<td>Online</td>
</tr>
<tr>
<td>setUCMUseAccounts</td>
<td>Set the UseAccount value.</td>
<td>Online</td>
</tr>
<tr>
<td>getUCMIsAutoNumber</td>
<td>Display the AutoNumber value.</td>
<td>Online</td>
</tr>
</tbody>
</table>
19.7.1 **getUCMOverRideFormat**

Use with WLST: Online

19.7.1.1 **Description**

Gets the value from the `config.cfg` file, indicating whether the OverrideFormat parameter is set. The OverrideFormat parameter enables users to choose the application format of their content items.

19.7.1.2 **Syntax**

```
getUCMOverRideFormat('appName')
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>appName</code></td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.7.1.3 **Example**

The following command displays the OverRideFormat parameter value:

```
getUCMOverRideFormat()
```

true

19.7.2 **setUCMOverRideFormat**

Use with WLST: Online

19.7.2.1 **Description**

Sets the OverrideFormat parameter to enable or disable the ability for users to choose the application format of their content items.

19.7.2.2 **Syntax**

```
setUCMOverRideFormat('value', ['appName'])
```
19.7.2.3 Example
The following command enables the OverRideFormat parameter:

```
getUCMOverRideFormat('True')
```

19.7.3 getUCMDownloadApplet
Use with WLST: Online

19.7.3.1 Description
Gets the value from the `config.cfg` file indicating whether the DownloadApplet parameter is set. DownloadApplet enables users to download multiple files from a search results page.

19.7.3.2 Syntax
```
getUCMDownloadApplet()  # appName
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.7.3.3 Example
The following command gets the value for the DownloadApplet parameter:

```
getUCMDownloadApplet()
```
```
true
```

19.7.4 setUCMDownloadApplet
Use with WLST: Online

19.7.4.1 Description
Sets the DownloadApplet parameter value to enable or disable the ability for users to download multiple files from a search results page.

19.7.4.2 Syntax
```
setUCMDownloadApplet('value', ['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Specifies whether to enable or disable the DownloadApplet option. Values can be: Yes, No, True, False, 1, 0.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>
19.7.4.3 Example
The following command sets the value for the DownloadApplet to enable the functionality:

```
setUCMDownloadApplet('Yes')
```

19.7.5 getUCMMultiUpload
Use with WLST: Online

19.7.5.1 Description
Gets the value from the config.cfg file indicating whether the MultiUpload parameter is set. MultiUpload allows multiple files to be zipped and checked in as a single content item.

19.7.5.2 Syntax

```
getUCMMultiUpload(['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.7.5.3 Example
The following command displays the value for the MultiUpload parameter:

```
getUCMMultiUpload()  
true
```

19.7.6 setUCMMultiUpload
Use with WLST: Online

19.7.6.1 Description
Sets the MultiUpload parameter value to allow or disallow multiple files to be zipped and checked in as a single content item.

19.7.6.2 Syntax

```
setUCMMultiUpload('value', ['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Specifies whether to enable or disable the MultiUpload option. Values can be: Yes, No, True, False, 1, 0.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.7.6.3 Example
The following command sets MultiUpload to allow multiple files to be zipped and check in as a single content item:

```
setUCMMultiUpload('1')
```

19.7.7 getUCMUseAccounts
Use with WLST: Online
19.7.7.1 Description
Gets the value from the `config.cfg` file indicating whether the UseAccounts parameter is set. UseAccounts enables the use of accounts in Oracle WebCenter Content.

19.7.7.2 Syntax
`getUCMUseAccounts(["appName"])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.7.7.3 Example
The following command displays the value for the UseAccounts option:
```
getUCMUseAccounts()  
True
```

19.7.8 setUCMUseAccounts
Use with WLST: Online

19.7.8.1 Description
Sets the UseAccounts parameter value to enable to disable the use of accounts in Oracle WebCenter Content.

19.7.8.2 Syntax
`setUCMUseAccounts('value', ['appName'])`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Specifies whether to enable or disable the UseAccounts option. Values can be: Yes, No, True, False, 1, 0.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.7.8.3 Example
The following command sets UseAccounts to enable accounts in Oracle WebCenter Content:
```
setUCMUseAccounts('True')
```

19.7.9 getUCMIsAutoNumber
Use with WLST: Online

19.7.9.1 Description
Gets the value from the `config.cfg` file indicating whether the IsAutoNumber parameter is set. The IsAutoNumber parameter enables automatic numbering of Content IDs.

19.7.9.2 Syntax
`getUCMIsAutoNumber(['appName'])`
### 19.7.9.3 Example

The following command displays the value for the IsAutoNumber parameter:

```ruby
getUCMIsAutoNumber()
```

True

### 19.7.10 setUCMIsAutoNumber

Use with WLST: Online

#### 19.7.10.1 Description

Sets the IsAutoNumber parameter value to enable or disable automatic numbering of Content IDs.

#### 19.7.10.2 Syntax

```ruby
setUCMIsAutoNumber('value','appName')
```

#### Argument | Definition
--- | ---
value | Specifies whether to enable or disable the AutoNumber option. Values can be: Yes, No, True, False, 1, 0.
appName | Optional. Name of the deployed application.

#### 19.7.10.3 Example

The following command sets IsAutoNumber to enable automatic numbering of Content IDs:

```ruby
setUCMIsAutoNumber('True')
```

### 19.7.11 getUCMAutoNumberPrefix

Use with WLST: Online

#### 19.7.11.1 Description

Gets the value from the `config.cfg` file for the AutoNumberPrefix parameter. The prefix is used in all automatically numbered content IDs for newly checked-in files, if the AutoNumber parameter is enabled.

#### 19.7.11.2 Syntax

```ruby
getUCMAutoNumberPrefix('appName')
```

#### Argument | Definition
--- | ---
appName | Optional. Name of the deployed application.

#### 19.7.11.3 Example

The following command displays the value for the AutoNumberPrefix parameter:

```ruby
getUCMAutoNumberPrefix()
```
19.7.12 setUCMAutoNumberPrefix

Use with WLST: Online

19.7.12.1 Description
Sets the AutoNumberPrefix parameter value to a prefix used in all automatically numbered content IDs for newly checked-in files (if the AutoNumber parameter is enabled).

19.7.12.2 Syntax

```python
setUCMUseAutoNumberPrefix('value', ['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>The prefix used in all automatically numbered content IDs for newly checked-in files. Only applies if the AutoNumber parameter is enabled.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.7.12.3 Example

The following command sets the AutoNumberPrefix:

```
setUCMAutoNumberPrefix('dadvm10231usor')
```

19.7.13 getUCMMajorRevLabelSeq

Use with WLST: Online

19.7.13.1 Description

Gets the value from the `config.cfg` file for MajorRevLabelSeq, which defines the major sequence for revision numbers. MajorRevLabelSeq is the first part of the Revision Label.

19.7.13.2 Syntax

```python
getUCMMajorRevLabelSeq(['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.7.13.3 Example

The following command displays the value for the MajorRevLabelSeq parameter:

```
getUCMMajorRevLabelSeq()
```

A1

19.7.14 setUCMMajorRevLabelSeq

Use with WLST: Online
19.7.14 General Configuration Commands

19.7.14.1 Description
Sets the value for the MajorRevLabelSeq parameter, which defines the major sequence for revision numbers. MajorRevLabelSeq is the first part of the Revision Label.

19.7.14.2 Syntax
setUCMMajorRevLabelSeq('value',['appName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Value can be any of the following: A through D, 1 through 8.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.7.14.3 Example
The following command sets MajorRevLabelSeq to 'A1':
setUCMMajorRevLabelSeq('A1')

19.7.15 getUCMMinorRevLabelSeq
Use with WLST: Online

19.7.15.1 Description
Gets the value from the config.cfg file for MinorRevLabelSeq, which defines the minor sequence for revision numbers. MinorRevLabelSeq is the second part of the Revision Label.

19.7.15.2 Syntax
getUCMMinorRevLabelSeq(['appName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.7.15.3 Example
The following command displays the value for the MinorRevLabelSeq parameter:
getUCMMinorRevLabelSeq()
b2

19.7.16 setUCMMinorRevLabelSeq
Use with WLST: Online

19.7.16.1 Description
Sets the value for the MinorRevLabelSeq parameter, which defines the minor sequence for revision numbers. MinorRevLabelSeq is the second part of the Revision Label.

19.7.16.2 Syntax
setUCMMinorRevLabelSeq('value',['appName'])
19.7.16.3 Example
The following command sets MinorRevLabelSeq to 'b2':
setUCMMinorRevLabelSeq('b2')

19.7.17 getUCMJspServerEnabled
Use with WLST: Online

19.7.17.1 Description
Gets the value from the config.cfg file indicating whether the JspServerEnabled is set. If the parameter is turned on, Content Server can execute Java Server Pages. The Java Server Pages must be checked in to Content Server.

19.7.17.2 Syntax
getUCMJspServerEnabled(['appName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.7.17.3 Example
The following command displays the value for the JspServerEnabled parameter:
getUCMJspServerEnabled()
true

19.7.18 setUCMJspServerEnabled
Use with WLST: Online

19.7.18.1 Description
Sets the value for the JspServerEnabled parameter. If the parameter is turned on, Content Server can execute Java Server Pages. The Java Server Pages must be checked in to Content Server.

19.7.18.2 Syntax
setUCMJspServerEnabled('value', ['appName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Specifies whether to enable or disable the parameter. Values can be: Yes, No, True, False, 1, 0.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.7.18.3 Example
The following command sets JspServerEnabled to 'true':
setUCMJspServerEnabled('true')

19.7.19 getUCMJspEnabledGroups
Use with WLST: Online

19.7.19.1 Description
Gets the value from the config.cfg file for the JspEnabledGroups parameter and lists is. JspEnabledGroups lists security groups enabled for Java Server Page functionality. Security groups have certain permissions for contributors and administrators.

19.7.19.2 Syntax
getUCMJspEnabledGroups(['appName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.7.19.3 Example
The following command displays the value for the JspEnabledGroups parameter:

getUCMJspEnabledGroups()
group1
Jsp

19.7.20 setUCMJspEnabledGroups
Use with WLST: Online

19.7.20.1 Description
Sets the value for the JspEnabledGroups parameter, which specifies security groups to be enabled for Java Server Page functionality.

19.7.20.2 Syntax
setUCMJspEnabledGroups('value','appName')

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Specifies the security groups to be enabled for Java Server Page functionality.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.7.20.3 Example
The following command sets JspEnabledGroups to 'group1':

setUCMJspEnabledGroups('group1')

19.8 Content Security Configuration Commands
Use the commands in Table 19–6 to configure content security options for the Oracle WebCenter Content Server instance.
19.8.1 getUCMCopyAccess
Use with WLST: Online

19.8.1.1 Description
Gets the value from the config.cfg file and displays it. When CopyAccess is enabled, users with Read privilege on a content item can get a copy of the native file.

19.8.1.2 Syntax
getUCMCopyAccess()

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.8.1.3 Example
The following command displays the CopyAccess value:

getUCMCopyAccess()  
True

19.8.2 setUCMCopyAccess
Use with WLST: Online

19.8.2.1 Description
Sets the CopyAccess value in the config.cfg file. When CopyAccess is enabled, users with Read privilege on a content item can get a copy of the native file.

19.8.2.2 Syntax
setUCMCopyAccess('value', ['appName'])
19.8.2.3 Example

The following command sets the value to enable the CopyAccess parameter:

```
setUCMCopyAccess('True')
```

19.8.3 getUCMExclusiveCheckout

Use with WLST: Online

19.8.3.1 Description

Gets the value in the `config.cfg` file and displays it. When the ExclusiveCheckout parameter is enabled, Admin privilege is required to check out a content item checked in by another user.

19.8.3.2 Syntax

```
getUCMExclusiveCheckout(['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.8.3.3 Example

The following command displays the value for the ExclusiveCheckout parameter:

```
getUCMExclusiveCheckout()
```

**True**

19.8.4 setUCMExclusiveCheckout

Use with WLST: Online

19.8.4.1 Description

Sets the value in the `config.cfg` file. When the ExclusiveCheckout parameter is enabled, Admin privilege is required to check out a content item checked in by another user.

19.8.4.2 Syntax

```
setUCMExclusiveCheckout('value', ['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Specifies whether to enable or disable the parameter. Values can be: Yes, No, True, False, 1, 0.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>
19.8.4.3 Example
The following command sets the value to enable the ExclusiveCheckout parameter:

```
setUCMExclusiveCheckout('True')
```

19.8.5 getUCMAuthorDelete
Use with WLST: Online

19.8.5.1 Description
Gets the value from the `config.cfg` file and displays it. When the AuthorDelete parameter is enabled, authors are allowed to delete their revisions without having Delete privilege.

19.8.5.2 Syntax
```
getUCMAuthorDelete(['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.8.5.3 Example
The following command displays the AuthorDelete parameter value:

```
getUCMAuthorDelete() 1
```

19.8.6 setUCMAuthorDelete
Use with WLST: Online

19.8.6.1 Description
Sets the AuthorDelete parameter value in the `config.cfg` file. When the AuthorDelete parameter is enabled, authors are allowed to delete their revisions without having Delete privilege.

19.8.6.2 Syntax
```
setUCMAuthorDelete('value',['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Specifies whether to enable or disable the parameter. Values can be: Yes, No, True, False, 1, 0.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.8.6.3 Example
The following command enables the AuthorDelete parameter:

```
setUCMAuthorDelete('1')
```

19.8.7 getUCMShowOnlyKnownAccounts
Use with WLST: Online
19.8.7.1 Description
Gets the value from the `config.cfg` file and displays it. When the ShowOnlyKnownAccounts parameter is enabled, the list of Content Server accounts on a check-in page will contain only globally-predefined accounts.

19.8.7.2 Syntax
```
getUCMShowOnlyKnownAccounts(['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.8.7.3 Example
The following command displays the ShowOnlyKnownAccounts parameter value as enabled:
```
getUCMShowOnlyKnownAccounts()
```
Yes

19.8 setUCMShowOnlyKnownAccounts
Use with WLST: Online

19.8.8.1 Description
Sets the ShowOnlyKnownAccount parameter value in the `config.cfg` file. When the ShowOnlyKnownAccounts parameter is enabled, the list of Content Server accounts on a check-in page will contain only globally-predefined accounts.

19.8.8.2 Syntax
```
setUCMShowOnlyKnownAccounts('value', ['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>
<td>Specifies whether to enable or disable the parameter. Values can be: Yes, No, True, False, 1, 0.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.8.8.3 Example
The following command enables the ShowOnlyKnownAccounts parameter:
```
setUCMShowOnlyKnownAccounts('yes')
```

19.9 Component Manager Configuration Commands
Use the commands in Table 19–7 to configure Component Manager options for the Oracle WebCenter Content Server instance.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>getUCMComponentStatus</td>
<td>Display the status of a component.</td>
<td>Online</td>
</tr>
<tr>
<td>setUCMComponentStatus</td>
<td>Set the status of a component.</td>
<td>Online</td>
</tr>
</tbody>
</table>
### getUCMComponentStatus

Use with WLST: Online

#### Description

Gets the status of a component. The status can be enabled or disabled.

#### Syntax

```python
getUCMComponentStatus('componentName', ['appName'])
```

#### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>componentName</td>
<td>Specifies a valid Content Server component name. For example, 'ContentFolios'.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

#### Example

The following command displays the status of the component 'ContentFolios':

```bash
getUCMComponentStatus('ContentFolios')
```

Status: Enabled

### setUCMComponentStatus

Use with WLST: Online

#### Description

Sets the status of a component in the config.cfg file.

#### Syntax

```python
setUCMComponentStatus('componentName','status', ['appName'])
```

#### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>componentName</td>
<td>Specifies a valid Content Server component name. For example, 'ContentFolios'.</td>
</tr>
<tr>
<td>status</td>
<td>Specifies whether the status of the component is enabled or disabled. Values can be: Enable, Disable.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>
19.9.2.3 Example
The following command sets the status of 'ContentFolios' to 'Enable':

```
setUCMComponentStatus('ContentFolios', 'Enable')
```

19.9.3 installUCMComponent
Use with WLST: Online

19.9.3.1 Description
Installs the component present at the specified file location.

19.9.3.2 Syntax
```
installUCMComponent('filePath', ['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>filePath</td>
<td>Specifies a valid file path to a component.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.9.3.3 Example
The following command installs the component at the location 'C:/manifest.zip':

```
installUCMComponent('C:/manifest.zip')
```

19.9.4 uninstallUCMComponent
Use with WLST: Online

19.9.4.1 Description
Uninstalls the specified component.

19.9.4.2 Syntax
```
uninstallUCMComponent('componentName', ['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>componentName</td>
<td>Specifies a valid Content Server component name. For example, 'ContentFolios'.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.9.4.3 Example
The following command uninstalls the component named 'ContentFolios':

```
uninstallUCMComponent('ContentFolios')
```

19.9.5 downloadUCMComponent
Use with WLST: Online

19.9.5.1 Description
Downloads the specified component to the specified file location.
19.9.5.2 Syntax

downloadUCMComponent('componentName','filePath',["appName"])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>componentName</td>
<td>Specifies a valid Content Server component name. For example, 'ContentFolios'.</td>
</tr>
<tr>
<td>filePath</td>
<td>Specifies a valid file path to a component.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.9.5.3 Example
The following command downloads the component 'ContentFolios' to 'C:/manifest.zip':

downloadUCMComponent('ContentFolios', 'C:/manifest.zip')

19.9.6 getUCMComponentConfig
Use with WLST: Online

19.9.6.1 Description
Gets the configuration of the specified component and displays it.

19.9.6.2 Syntax

getUCMComponentConfig('componentName',["appName"])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>componentName</td>
<td>Specifies a valid component name. For example, 'ContentFolios'.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.9.6.3 Example
The following command displays the configuration for the component 'ContentFolios':

getUCMComponentConfig('ContentFolios')

19.9.7 updateUCMComponentConfig
Use with WLST: Online

19.9.7.1 Description
Sets the specified component with the configuration options provided.

19.9.7.2 Syntax

updateUCMComponentConfig('componentName','updateParams',["appName"])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>componentName</td>
<td>Specifies a valid component name. For example, 'Folders_g'.</td>
</tr>
<tr>
<td>updateParams</td>
<td>Specifies valid parameters for the component.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>
19.9.7.3 Example
The following command sets configuration parameters for the component 'Folders_g':

```
updateUCMComponentConfig('Folders_g','GetCopyAccess:true,CollectionHiddenMeta:xHidden')
```

19.10 Records Management Configuration Commands
Use the commands in Table 19–8 to configure records management options for the Oracle WebCenter Content Server instance.

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>getRMLevel</td>
<td>Display the type of records management configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>getRMConfigurationLevel</td>
<td>Display the records management configuration level.</td>
<td>Online</td>
</tr>
<tr>
<td>getRMFeatures</td>
<td>Display records management features.</td>
<td>Online</td>
</tr>
<tr>
<td>getRMDispositionActions</td>
<td>Display records management dispositions actions.</td>
<td>Online</td>
</tr>
<tr>
<td>rmUpdate</td>
<td>Update the records management configuration.</td>
<td>Online</td>
</tr>
<tr>
<td>addOutgoingProvider</td>
<td>Add an outgoing provider for the Adapter server used to store records management content.</td>
<td>Online</td>
</tr>
<tr>
<td>registerSource</td>
<td>Register the source for the Adapter server repository so records management can find it.</td>
<td>Online</td>
</tr>
</tbody>
</table>

19.10.1 getRMLevel
Use with WLST: Online

19.10.1.1 Description
Gets the type of records management configuration and displays it. The type can have the following values:

- none
- standalone
- adapter

19.10.1.2 Syntax
```
getRMLevel(['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.10.1.3 Example
The following command displays the type of records management configuration:
```
getRMLevel()
```
19.10.2 getRMConfigurationLevel

Use with WLST: Online

19.10.2.1 Description

Gets the records management configuration level and displays it. The configuration level can have the following values:

- **minimal**: Enables minimal amount of functionality and excludes some disposition actions and most of the application features. This is the default when the software is enabled.

- **typical**: Enables all disposition actions and all features except for DoD Configuration, Classified Topics, FOIA/PA tracking (Freedom of Information Act/Privacy Act), and E-mail.

- **dod2**: Enables the features from a Typical installation with the addition of DoD Configuration and E-mail.

- **dodclassified**: Enables all features except for FOIA/PA.

- **custom**: Enables the ability to choose a variety of features. Some disposition actions are dependent on other actions. If an action is selected, dependent actions are also automatically selected.

19.10.2.2 Syntax

getRMConfigurationLevel(['appName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.10.2.3 Example

The following command displays the type of records management configuration:

getRMConfigurationLevel(['appName'])

minimal

19.10.3 getRMFeatures

Use with WLST: Online

19.10.3.1 Description

Gets a list of records management features and displays it. The list can have the following values:

- **feature_related_content**
- **feature_audit_trigger**
- **feature_subject_to_review**
- **feature_revision_dates**
- **feature_security_markings**
- **feature_email_fields**
19.10.3.2 Syntax
getRMFeatures(['appName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.10.3.3 Example
The following command displays the records management features:

getRMFeatures()
feature_related_content
feature_audit_trigger
feature_subject_to_review

19.10.4 getRMDispositionActions
Use with WLST: Online

19.10.4.1 Description
Gets the values for records management disposition actions and displays them. The list can have the following values:

- actions_activate
- actionsObsolete
- actions_cancel
- actions_rescind
- actions_expire
- actions_cutoff
- actions_approve_deletion
- actions_destroy

19.10.4.2 Syntax
getRMDispositionActions(['appName'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.10.4.3 Example
The following command displays a list of records management disposition actions:

getRMDispositionActions()
actions_active
actionsObsolete
actions_cancel
19.10.5 rmUpdate

Use with WLST: Online

19.10.5.1 Description
Sets the records management configuration with feature and disposition actions and enables it.

19.10.5.2 Syntax

\[
\text{rmUpdate}(\text{urmLevel}, \text{level}, \text{featuresList}, \text{dispositionActionsList}, \text{enableRTMandURMAgent}, ['\text{appName}'])
\]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>urmLevel</td>
<td>Specifies a valid records management configuration type.</td>
</tr>
<tr>
<td>level</td>
<td>Specifies a valid records management level.</td>
</tr>
<tr>
<td>featuresList</td>
<td>Specifies valid records management features.</td>
</tr>
<tr>
<td>dispositionActionsList</td>
<td>Specifies valid disposition actions.</td>
</tr>
<tr>
<td>enableRTMandURMAgent</td>
<td>Specifies whether to enable the RTM and URM Agent.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.10.5.3 Example
The following command sets the records management configuration and enables it:

\[
\text{rmUpdate}(\text{urmLevel}='\text{standalone}', \text{level}='\text{custom}', \text{featuresList}='\text{feature_related_content:feature_revision_dates}', \text{dispositionActionsList}='\text{actions_activate:actions:obsolete}', \text{enableRTMandURMAgent}='1')
\]

19.10.6 addOutgoingProvider
Use with WLST: Online

19.10.6.1 Description
Defines the outgoing provider that enables the Adapter server to connect to the Content Server instance with records management enabled.

19.10.6.2 Syntax

\[
\text{addOutgoingProvider}(\text{ProviderName}, \text{ProviderDescription}, \text{ServerHostName}, \text{HTTPServerAddress}, \text{ServerPort}, \text{InstanceName}, \text{RelativeWebRoot}, \text{extraUpdateParams}, ['\text{appName}'])
\]

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProviderName</td>
<td>Name of the outgoing provider.</td>
</tr>
<tr>
<td>ProviderDescription</td>
<td>Description of the outgoing provider.</td>
</tr>
<tr>
<td>ServerHostname</td>
<td>Name of the server host.</td>
</tr>
<tr>
<td>HTTPServerAddress</td>
<td>Address of the HTTP server.</td>
</tr>
<tr>
<td>ServerPort</td>
<td>Number of the server port.</td>
</tr>
<tr>
<td>InstanceName</td>
<td>Name of the instance for the Content Server with records management enabled.</td>
</tr>
<tr>
<td>RelativeWebRoot</td>
<td>Name of the relative web root.</td>
</tr>
</tbody>
</table>
19.10.6.3 Example
The following command defines an outgoing provider for the Adapter server to connect to the records management instance:

```
addOutgoingProvider(ProviderName='x',ProviderDescription='y',ServerHostName='localhost',HTTPServerAddress='z',ServerPort='4444',InstanceName='w',RelativeWebRoot='cs',updateParams='RefineryMaxProJobs:100,IsRefinery:1')
```

19.10.7 registerSource
Use with WLST: Online

19.10.7.1 Description
Registration ensures that records management is aware of the Adapter and is ready to manage the stored content in the Adapter server’s repository.

19.10.7.2 Syntax
```
registerSource(ProviderName, SourceName, SourceTableName, SourceDisplayName, ['appName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProviderName</td>
<td>Name of the outgoing provider.</td>
</tr>
<tr>
<td>SourceName</td>
<td>Description of the source in the Adapter repository.</td>
</tr>
<tr>
<td>SourceTableName</td>
<td>Name of the source table in the Adapter repository.</td>
</tr>
<tr>
<td>SourceDisplayName</td>
<td>Display name for the source in the Adapter repository.</td>
</tr>
<tr>
<td>appName</td>
<td>Optional. Name of the deployed application.</td>
</tr>
</tbody>
</table>

19.10.7.3 Example
The following command registers the Adapter server’s repository source for the records management stored content:

```
registerSource(ProviderName='x',SourceName='y',SourceTableName='z',SourceDisplayName='w')
```

19.11 User Interface Commands
Use the commands in Table 19–9 to display and update the configuration of the Oracle WebCenter Content user interface introduced in Oracle WebCenter Content 11g Release 1 (11.1.1.8). For more information about this user interface, see “Getting Started with the WebCenter Content Interface” in Using Oracle WebCenter Content.

These commands can be used only after the `WccADFUI.zip` file has been installed and enabled on Oracle WebCenter Content.
To use these custom commands, you must invoke the WLST script from the appropriate Oracle home. Do not use the WLST script in the WebLogic Server home. For Oracle WebCenter Content user interface commands, the script is located at:

- UNIX: WlstResources/common/bin/wlst.sh
- Windows: WlstResources\common\bin\wlst.cmd

Before you run the WLST script, you must set the WL_HOME environment variable in the UNIX shell or Windows command prompt.

- UNIX: export WL_HOME=/Oracle/WccUI/wlserver_10.3
- Windows: set WL_HOME=C:\Oracle\WccUI\wlserver_10.3

Configuration changes made using these WLST commands are only effective after you restart the Managed Server on which the WebCenter Content - Web UI application is deployed.

### Table 19–9  WLST WebCenter Content User Interface Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>displayWccAdfConfig</td>
<td>Display the configuration of the WebCenter Content user interface application.</td>
<td>Online</td>
</tr>
<tr>
<td>updateWccAdfConfig</td>
<td>Update the configuration of the WebCenter Content user interface application.</td>
<td>Online</td>
</tr>
</tbody>
</table>

#### 19.11.1 displayWccAdfConfig

Use with WLST: Online

**19.11.1.1 Description**

Displays the configuration of the Oracle WebCenter Content user interface application.

**19.11.1.2 Syntax**

`displayWccAdfConfig(appName='value', [attrName='value'])`

**Argument**

<table>
<thead>
<tr>
<th>Definition</th>
<th>Argument</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of the deployed application.</td>
<td>appName value</td>
</tr>
<tr>
<td>Optional. Name of the attribute to display If omitted all attributes are displayed.</td>
<td>attrName value</td>
</tr>
</tbody>
</table>

**19.11.1.3 Examples**

The following command displays all configuration attributes for the Oracle WebCenter Content user interface deployed with the application name "Oracle WebCenter Content - Web UI".

```
wls:/wccadf_domain/serverConfig> displayWccAdfConfig(appName='Oracle WebCenter Content - Web UI')
WccInstanceName = Default
WccInstanceConnectionName = WccAdfDefaultConnection
ProxyContent = true
DocumentPreview = DOCUMENT_VIEWER
DefaultLocale = en_US
DefaultTimeZone = America/Los_Angeles
ApplicationUrl = null
```
ClusterCompatible = false
TemporaryDirectory = null
MaximumUploadedFileSize = 52428800
MaximumWindowsPerSession = 10
WccLoginPageEnabled = true
SkinFamily = wcc-skin
SkinVersion = null
CustomBrandingLogo = null
CustomBrandingTitle = null

The following command displays the value of the configuration attribute named "WccInstanceConnectionName".

```
wlsci:205218834710/205218835220/oracle/webcenter/content/serverconfig> displayWccAdfConfig(appName='Oracle WebCenter Content - Web UI', attrName='WccInstanceConnectionName')
Attribute WccInstanceConnectionName value is: WccAdfDefaultConnection
```

**19.11.2 updateWccAdfConfig**

Use with WLST: Online

**19.11.2.1 Description**

Updates the configuration of the Oracle WebCenter Content user interface application. The command can be used to update a single attribute or multiple attributes.

**19.11.2.2 Syntax**

```
updateWccAdfConfig(appName, [wccInstanceName], [wccInstanceConnectionName],
[proxyContent], [documentPreview], [defaultLocale], [defaultTimeZone],
[applicationUrl], [clusterCompatible], [temporaryDirectory],
[maximumUploadedFileSize], [maximumWindowsPerSession],
[wccLoginPageEnabled], [skinFamily], [skinVersion], [customBrandingLogo],
[customBrandingTitle])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>Name of the deployed application.</td>
</tr>
<tr>
<td>wccInstanceName</td>
<td>Reserved for future use.</td>
</tr>
<tr>
<td>wccInstanceConnectionName</td>
<td>Name of the Content Server connection in Connection Architecture (connections.xml).</td>
</tr>
<tr>
<td>proxyContent</td>
<td>Controls how a browser retrieves document content, including the native file, the web-viewable rendition, thumbnails, and attachments. Must be &quot;true&quot; or &quot;false&quot;. If &quot;true&quot; the browser requests this content from the Oracle WebCenter Content user interface, which in turn retrieves it from Content Server. If &quot;false&quot; the browser requests this content directly from Content Server.</td>
</tr>
</tbody>
</table>
| documentPreview           | Controls the preview displayed for a document. Must be one of the following case-sensitive values:  
  ■ DOCUMENT_VIEWER: Use the document viewer (if available)  
  ■ WEB_VIEWABLE: Display the web-viewable rendition (if available)  
  ■ NONE: No preview |

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>defaultLocale</td>
<td>Locale to use if a user's locale cannot be otherwise determined. Must be in the format used by the Java class java.util.Locale (for example, &quot;en_US&quot;).</td>
</tr>
<tr>
<td>defaultTimeZone</td>
<td>Time zone to use if the user's time zone cannot be otherwise determined. Must be in the format used by the Java method java.util.TimeZone#getTimeZone (for example, &quot;America/Los_Angeles&quot;)</td>
</tr>
<tr>
<td>applicationUrl</td>
<td>Specifies the scheme, hostname, and port of URLs generated by the Oracle WebCenter Content user interface (for example, <a href="https://wcc.example.com:16225">https://wcc.example.com:16225</a>). Optional. If not set, the scheme, hostname, and port are determined from the HTTP request.</td>
</tr>
<tr>
<td>clusterCompatible</td>
<td>Specifies whether the Oracle WebCenter Content user interface supports session replication among nodes in a cluster. Either &quot;true&quot; or &quot;false&quot;. Must be &quot;true&quot; if session replication is enabled on the Java EE Web Container.</td>
</tr>
<tr>
<td>temporaryDirectory</td>
<td>Base directory for temporary files created by the Oracle WebCenter Content user interface, such as in-progress uploads. This directory should be empty, on a fast file system with adequate free space, and readable and writable by only the owner of the process running the Oracle WebCenter Content user interface. The Oracle WebCenter Content user interface will create and manage a subdirectory structure. If clusterCompatible is &quot;true&quot; this directory must also be on a filesystem shared across all nodes in the cluster. Oracle recommends setting this attribute even if clusterCompatible is &quot;false&quot;. If not set, the value of the Java system property java.io.tmpdir is used.</td>
</tr>
<tr>
<td>maximumUploadedFileSize</td>
<td>Maximum size of uploaded files (in bytes). If less than zero, there is no maximum size. If zero, file upload is disabled.</td>
</tr>
<tr>
<td>maximumWindowsPerSession</td>
<td>Maximum number of active windows (or browser tabs, depending on browser configuration) per session. When this limit is reached, the least recently used window expires and subsequent interaction with that window displays an error message. The Oracle WebCenter Content user interface main page, wccmain, does not contribute to the session's window count and only expires when the session expires or is logged out. The document properties page, wccdoc, counts as two windows if documentPreview is &quot;DOCUMENT_VIEWER&quot;, but counts as one window if documentPreview is &quot;WEB_VIEWABLE&quot; or &quot;NONE&quot;. Use this attribute to control the maximum application server memory consumed by each session. Set to 0 to not limit the number of windows per session.</td>
</tr>
<tr>
<td>wccLoginPageEnabled</td>
<td>Specifies whether the &quot;wcclogin&quot; page is enabled or disabled. Either &quot;true&quot; or &quot;false&quot;. When Single Sign-On (SSO) authentication is enabled, the attribute value should be set to &quot;false&quot; to prevent users from using the &quot;wcclogin&quot; page instead of SSO.</td>
</tr>
<tr>
<td>skinFamily</td>
<td>Name of the skin family. The &lt;skin-family&gt; element in trinidad-config.xml is set to the value of this attribute.</td>
</tr>
</tbody>
</table>
19.11.2.3 Examples

The following command sets the proxyContent attribute to "true" for the Oracle WebCenter Content user interface deployed with the application name "Oracle WebCenter Content - Web UI". With this setting, browsers will retrieve document content from the Oracle WebCenter Content user interface, rather than directly from Content Server.

```
updateWccAdfConfig(appName='Oracle WebCenter Content - Web UI', proxyContent=true)
```

The following command sets the defaultLocale attribute to "en_US" and the defaultTimeZone attribute to "America/Los_Angeles". If the Oracle WebCenter Content user interface cannot determine a user's preferred locale and time zone (for example, based on user preferences), it will default to the English language and display times using the America/Los_Angeles time zone.

```
updateWccAdfConfig(appName='Oracle WebCenter Content - Web UI', defaultLocale='en_US', defaultTimeZone='America/Los_Angeles')
```

The following command sets the temporaryDirectory attribute to "/tmp/wcc". The Oracle WebCenter Content user interface will store temporary files in this directory on the application server.

```
updateWccAdfConfig(appName='Oracle WebCenter Content - Web UI',
temporaryDirectory='/tmp/wcc')
```

The following command sets the wccLoginPageEnabled property to "false". This prevents users from authenticating to the Oracle WebCenter Content user interface using its "wcclogin" page, requiring them to use Single Sign-On (SSO).

```
updateWccAdfConfig(appName='Oracle WebCenter Content - Web UI',
wccLoginPageEnabled=false)
```

19.12 User Interface Connection Commands

Use the commands in Table 19–10 to manage connections for the Oracle WebCenter Content user interface introduced in WebCenter Content 11g Release 1 (11.1.1.8). For more information about this user interface, see "Getting Started with the WebCenter Content Interface" in Using Oracle WebCenter Content.

These commands can be used only after the WccADFUI.zip file has been installed and enabled on Oracle WebCenter Content. To use these custom commands, you must invoke the WLST script from the appropriate Oracle home. Do not use the WLST script in the WebLogic Server home. For the Oracle WebCenter Content user interface commands, the script is located at:

- **UNIX**: ConnArchWlstResources/common/bin/manageconnwlst.sh
- **Windows**: ConnArchWlstResources\common\bin\manageconnwlst.cmd
Before you run the WLST script, you must set the WL_HOME environment variable in the UNIX shell or Windows command prompt.

- UNIX: export WL_HOME=/Oracle/WccUI/wlserver_10.3
- Windows: set WL_HOME=C:\Oracle\WccUI\wlserver_10.3

Configuration changes made using these WLST commands are only effective after you restart the Managed Server on which the WebCenter Content - Web UI application is deployed.

### Table 19–10 WLST WebCenter Content User Interface Connection Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>createRIDCConnection</td>
<td>Create a new RIDC connection.</td>
<td>Online</td>
</tr>
<tr>
<td>updateRIDCConnection</td>
<td>Update existing RIDC connection properties.</td>
<td>Online</td>
</tr>
<tr>
<td>listRIDCConnections</td>
<td>List all the RIDC connection Mbeans present in an application.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteRIDCConnection</td>
<td>Remove an existing RIDC connection.</td>
<td>Online</td>
</tr>
<tr>
<td>displayRIDCConnection</td>
<td>Get all the attributes and their values associated with an RIDC connection Mbean with an option to get the value for a given attribute name.</td>
<td>Online</td>
</tr>
</tbody>
</table>

#### 19.12.1 createRIDCConnection

Use with WLST: Online

#### 19.12.1.1 Description

Creates a new RIDC connection reference and adds the connection name to the mds layer.

#### 19.12.1.2 Syntax

`createRIDCConnection('appName','connName')`

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>java.lang.String</td>
<td>Name of the application for which the connection will be created.</td>
</tr>
<tr>
<td>connName</td>
<td>java.lang.String</td>
<td>Name of the connection which is the placeholder for RIDC connection reference.</td>
</tr>
</tbody>
</table>

#### 19.12.1.3 Example

The following command creates an RIDC connection 'WccAdfDefaultConnection' in the application 'Oracle WebCenter Content - Web UI':

`createRIDCConnection('Oracle WebCenter Content - Web UI','WccAdfDefaultConnection')`

#### 19.12.2 updateRIDCConnection

Use with WLST: Online
19.12.2.1 Description
Updates the connection properties for an existing RIDC connection reference. These changes and additions are read as runtime data and saved in the mds layer.

19.12.2 Syntax
updateRIDCConnection(appName, connName, ['connUrl'], ['connSockettimeout'], ['connPoolMethod'], ['connPoolSize'], ['connWaitTime'], ['credUsername'], ['credPassword'], ['credAppidkey'], ['credImpersonationAllowed'], ['jaxwsStack'], ['jaxwsPolicy'], ['jaxwsJpsconfigfile'], ['jaxwsSkipstackoptimizations'], ['jaxwsServerInstancename'], ['jaxwsRegisteridentityswitchfilter'], ['httpLibrary'], ['idcsAlgorithm'], ['idcsKeystoreFile'], ['idcsKeystorePassword'], ['idcsKeystoreAlias'], ['idcsKeystoreAliasPassword'], ['idcsTrustmanagerFile'], ['idcsTrustmanagerPassword'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>java.lang.String</td>
<td>Name of the application for which the connection has to be updated.</td>
</tr>
<tr>
<td>connName</td>
<td>java.lang.String</td>
<td>Name of the connection which will be placeholder for RIDC connection properties.</td>
</tr>
<tr>
<td>connURL</td>
<td>java.lang.String</td>
<td>RIDC connection URL property.</td>
</tr>
<tr>
<td>connSockettimeout</td>
<td>java.lang.Integer</td>
<td>RIDC connection socket timeout property in seconds.</td>
</tr>
<tr>
<td>connPoolMethod</td>
<td>java.lang.String</td>
<td>RIDC connection pool method property.</td>
</tr>
<tr>
<td>connPoolSize</td>
<td>java.lang.Integer</td>
<td>RIDC connection pool size property.</td>
</tr>
<tr>
<td>connWaitTime</td>
<td>java.lang.Integer</td>
<td>RIDC connection wait time property.</td>
</tr>
<tr>
<td>credUsername</td>
<td>java.lang.String</td>
<td>RIDC authorization credential username property.</td>
</tr>
<tr>
<td>credPassword</td>
<td>java.lang.String</td>
<td>RIDC authorization credential password property.</td>
</tr>
<tr>
<td>credAppidkey</td>
<td>java.lang.String</td>
<td>RIDC authorization credential appid key property.</td>
</tr>
<tr>
<td>credImpersonationAllowed</td>
<td>java.lang.Boolean</td>
<td>Credential impersonation allowed property.</td>
</tr>
<tr>
<td>jaxwsStack</td>
<td>java.lang.String</td>
<td>JaxWs protocol stack property.</td>
</tr>
<tr>
<td>jaxwsPolicy</td>
<td>java.lang.String</td>
<td>JaxWs protocol policy property.</td>
</tr>
<tr>
<td>jaxwsJpsconfigfile</td>
<td>java.lang.String</td>
<td>JaxWs protocol Jps config file property.</td>
</tr>
<tr>
<td>jaxwsSkipstackoptimizations</td>
<td>java.lang.Boolean</td>
<td>JaxWs protocol skip stack optimizations property.</td>
</tr>
<tr>
<td>jaxwsServerInstancename</td>
<td>java.lang.String</td>
<td>JaxWs protocol server instance name property.</td>
</tr>
<tr>
<td>httpLibrary</td>
<td>java.lang.String</td>
<td>Http protocol library property.</td>
</tr>
</tbody>
</table>
19.12.2.3 Example
The following command updates the Intradoc Protocol Connection with a 90 second
socket timeout.

```
updateRIDCConnection('Oracle WebCenter Content - Web UI','WccAdfDefaultConnection',
                connUrl='idc://contentserver:4444',credUsername='weblogic',connSockettimeout=90)
```

19.12.3 listRIDCConnections
Use with WLST: Online

19.12.3.1 Description
Lists all the RIDC connection Mbeans present in an ADF connection architecture based
application.

19.12.3.2 Syntax
```
listRIDCConnections('appName')
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>java.lang.String</td>
<td>Name of the application for which the connection mbeans has to be returned.</td>
</tr>
</tbody>
</table>

19.12.3.3 Example
The following command returns all the connection Mbeans with connection type
'RIDC' present in application 'Oracle WebCenter Content - Web UI'.

```
listRIDCConnections('Oracle WebCenter Content - Web UI')
```

19.12.4 deleteRIDCConnection
Use with WLST: Online

19.12.4.1 Description
Lists all the RIDC connection Mbeans present in an ADF connection architecture based
application.

19.12.4.2 Syntax
```
deleteRIDCConnection('appName','connName')
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>java.lang.String</td>
<td>Name of the application for which the connection mbeans has to be returned.</td>
</tr>
</tbody>
</table>
| connName         | java.lang.String| Name of the connection Mbean to be deleted.}
19.12.4.3 Example
The following command removes the connection name 'WccAdfDefaultConnection' present in application 'Oracle WebCenter Content - Web UI'.

```
deleteRIDCConnection('Oracle WebCenter Content - Web UI','WccAdfDefaultConnection')
```

19.12.5 displayRIDCConnection

Use with WLST: Online

19.12.5.1 Description

Gets all the attribute name-value pairs present in a given connection Mbean of an application, with an option to get the value of an attribute present in a given connection Mbean of an application.

19.12.5.2 Syntax

```
displayRIDCConnection(appName,connName,
['attrName'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appName</td>
<td>java.lang.String</td>
<td>Name of the application for which the connection Mbean attribute value is desired.</td>
</tr>
<tr>
<td>connName</td>
<td>java.lang.String</td>
<td>Name of the connection that contains the mbean property.</td>
</tr>
<tr>
<td>attrName</td>
<td>java.lang.String</td>
<td>Optional. Attribute name in connection Mbean for which value is desired.</td>
</tr>
</tbody>
</table>

19.12.5.3 Examples

The following command retrieves the connection property value of 'PropConnectionUrl'.

```
displayRIDCConnection('Oracle WebCenter Content - Web UI','WccAdfDefaultConnection','PropConnectionUrl')
```

The following command retrieves all the connection properties present in connection 'WccAdfDefaultConnection'.

```
displayRIDCConnection('Oracle WebCenter Content - Web UI','WccAdfDefaultConnection')
```
This chapter provides detailed descriptions of custom WLST commands for Oracle WebCenter Enterprise Capture, including command syntax, arguments and command examples.

The following sections describe the custom WLST commands for Oracle WebCenter Enterprise Capture. These commands enable you to access and modify various configuration parameters for Oracle WebCenter Enterprise Capture from the command line. Topics include:

- Section 20.1, "Overview of WLST Oracle WebCenter Enterprise Capture Command Categories"
- Section 20.2, "Configuration Commands"

To display all the supported Capture commands that can be invoked from WLST, enter the following command at the WLST prompt:

```python
help('capture')
```

For information about Enterprise Capture workspace console use, see Oracle Fusion Middleware Managing Oracle WebCenter Enterprise Capture.

---

**Note:** To use the Oracle WebCenter Enterprise Capture custom commands, you must invoke the WLST script from the Oracle Common home in which the component has been installed. Either connect to the Oracle WebLogic Server managed server (default port 16400) on which the Oracle WebCenter Enterprise Capture application is deployed, or connect to the Oracle WebLogic Server administration server and change your location to domainRuntime by executing the command: `domainRuntime()`

See "Using Custom WLST Commands" in Oracle Fusion Middleware Administrator’s Guide.

---

### 20.1 Overview of WLST Oracle WebCenter Enterprise Capture Command Categories

WLST Oracle WebCenter Enterprise Capture commands are covered in the following category:
Table 20–1  WLST Oracle WebCenter Enterprise Capture Command Categories

<table>
<thead>
<tr>
<th>Command Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Commands</td>
<td>View and manage configuration for Oracle WebCenter Enterprise Capture.</td>
</tr>
</tbody>
</table>

20.2 Configuration Commands

Use the commands in Table 20–2 to configure Oracle WebCenter Enterprise Capture.

Table 20–2  WLST Server Oracle WebCenter Enterprise Capture Configuration Commands

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td>listWorkspaces</td>
<td>List all of the Capture workspaces to which the administrator or workspace manager has access.</td>
<td>Online</td>
</tr>
<tr>
<td>listBatches</td>
<td>List all of the batches within a specified workspace.</td>
<td>Online</td>
</tr>
<tr>
<td>exportBatch</td>
<td>Export a batch including images and metadata to a ZIP file.</td>
<td>Online</td>
</tr>
<tr>
<td>exportWorkspace</td>
<td>Export an entire Capture workspace including documents, metadata, scripts, profiles and jobs to an XML file.</td>
<td>Online</td>
</tr>
<tr>
<td>importWorkspace</td>
<td>Import a workspace from a XML file generated with the exportWorkspace command.</td>
<td>Online</td>
</tr>
<tr>
<td>unlockBatch</td>
<td>Delete the batch lock record for the specified batch.</td>
<td>Online</td>
</tr>
<tr>
<td>listLockedBatches</td>
<td>Display the list of locked batches.</td>
<td>Online</td>
</tr>
<tr>
<td>listCaptureConfig</td>
<td>Display the list of all Enterprise Capture configuration attributes with their values.</td>
<td>Online</td>
</tr>
<tr>
<td>getCaptureConfig</td>
<td>Fetch the value of the attribute specified as an argument.</td>
<td>Online</td>
</tr>
<tr>
<td>setCaptureConfig</td>
<td>Set the value of the Enterprise Capture attribute to the provided value.</td>
<td>Online</td>
</tr>
<tr>
<td>scanForClientBundles</td>
<td>Scan the client bundle directory for updated bundles.</td>
<td>Online</td>
</tr>
<tr>
<td>deleteBundle</td>
<td>Delete an existing client bundle.</td>
<td>Online</td>
</tr>
<tr>
<td>setObjectProperty</td>
<td>Set the value of a property defined in a Capture object which could be a client profile, processor job, or a commit profile.</td>
<td>Online</td>
</tr>
<tr>
<td>getObjectProperty</td>
<td>Print the value of a specific property of a Capture object which could be a client profile, processor job, or a commit profile.</td>
<td>Online</td>
</tr>
<tr>
<td>setObjectCredentials</td>
<td>Set the user name and password associated with a Capture object (client profile, processor job, or commit profile).</td>
<td>Online</td>
</tr>
<tr>
<td>updateScript</td>
<td>Update a script within a workspace.</td>
<td>Online</td>
</tr>
</tbody>
</table>
20.2.1 listWorkspaces

Use with WLST: Online.

20.2.1.1 Description
Lists all Capture workspaces to which the administrator or workspace manager has access.

20.2.1.2 Syntax
listWorkspaces()

20.2.1.3 Example
The following example lists all of the workspaces, preceded by their IDs.

```java
listWorkspaces()
4 bills_workspace
2 certificates_workspace
5 receipts_workspace
```

20.2.2 listBatches

Use with WLST: Online.

20.2.2.1 Description
Lists all batches for the workspace specified by its ID. Each batch listed has five columns (displayed in this order): Batch ID, Batch Name, Created By (user), Last Updated By (user), and Last Modified (date).

20.2.2.2 Syntax
listBatches (WORKSPACE_ID)

20.2.2.3 Example
The following example lists the batches for the workspace ID 3.

```java
wls://base_domain/domainRuntime> listBatches(1)
14 ABI1 <anonymous> <anonymous> 4/26/13
15 ABI2 <anonymous> <anonymous> 4/26/13
1 BCP11 captureuser <anonymous> 4/17/13
```

20.2.3 exportBatch

Use with WLST: Online.

20.2.3.1 Description
Exports a batch including images and metadata to a ZIP file containing an XML file and the batch item files.
20.2.3.2 Syntax

```
exportBatch(BATCH_ID, 'FILE_NAME')
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BATCH_ID</td>
<td>ID number for a batch. Can be optionally specified within single quotes. For example, both 3 and '3' are valid arguments for a batch with ID 3.</td>
</tr>
<tr>
<td>FILE_NAME</td>
<td>Name of the ZIP file to which the batch will be exported. The file name must be specified along with its complete path.</td>
</tr>
</tbody>
</table>

**Note:** The path should be separated by forward slashes "/" regardless of the operating system.

20.2.3.3 Example

The following example exports a batch.

```
exportBatch(8, '/home/abc/batch8.zip')
Batch successfully exported
```

20.2.4 exportWorkspace

Use with WLST: Online.

20.2.4.1 Description

Exports an entire workspace including documents, metadata, scripts, profiles and jobs to an XML file.

20.2.4.2 Syntax

```
exportWorkspace(WORKSPACE_ID, 'FILE_NAME')
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKSPACE_ID</td>
<td>ID number for a workspace. Can be optionally specified within single quotes. For example, both 3 and '3' are valid arguments for a workspace with ID 3.</td>
</tr>
<tr>
<td>FILE_NAME</td>
<td>Name of the file to which the workspace will be exported. The file name must be specified along with its complete path.</td>
</tr>
</tbody>
</table>

**Note:** The path should be separated by forward slashes "/" regardless of the operating system.

20.2.4.3 Example

The following example exports the workspace with the ID of 3.

```
exportWorkspace(3, '/home/abc/workspace3.xml')
Workspace successfully exported
```

20.2.5 importWorkspace

Use with WLST: Online.
20.2.5.1 Description
Imports a workspace from a XML file generated with the `exportWorkspace` command. The workspace being imported must not exist in the Oracle WebCenter Enterprise Capture database.

20.2.5.2 Syntax
```
importWorkspace('FILE_NAME')
```

### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>FILE_NAME</td>
<td>Name of the XML file from which a workspace is imported. The file name must be specified along with its complete path.</td>
</tr>
</tbody>
</table>

20.2.5.3 Examples

The following example imports a workspace with the file name of `workspace3.xml`.
```
importWorkspace('/home/abc/workspace3.xml')
```

Workspace successfully imported

20.2.6 unlockBatch

Use with WLST: Online.

20.2.6.1 Description

Deletes the batch lock record for the specified batch. The batch will be put into a READY state so that it can be opened by the client.

20.2.6.2 Syntax

```
unlockBatch('BATCH_ID')
```

### Example

The following example unlocks a batch ID 8.
```
unlockBatch('8')
```

Batch successfully unlocked

20.2.7 listLockedBatches

Use with WLST: Online.

20.2.7.1 Description

Displays the list of locked batches. Each batch listed has six columns (displayed in this order): Workspace ID, Batch ID, Batch Name, Created By (user), Workstation (name/IP address), Last Modified (date).

20.2.7.2 Syntax

```
listLockedBatches('WORKSPACE_ID')
```

### Argument Definition

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>BATCH_ID</td>
<td>ID number for a batch. Can be optionally specified within single quotes. For example, both 3 and '3' are valid arguments for a batch with ID 3.</td>
</tr>
</tbody>
</table>

### Example

The following example lists all locked batches.
```
listLockedBatches()'
```

Batch successfully unlocked
### 20.2.7.3 Example
The following example lists locked batches for the workspace ID 3.

```bash
wls:/base_domain/domainRuntime> listLockedBatches(1)
1 14 ABI1 <anonymous> sic01lzz.example.com 4/26/13
1 15 ABI2 <anonymous> slc01lzz.example.xom 4/26/13
1 17 BCP16 <captureuser> 10.159.104.189 4/30/13
```

### 20.2.8 listCaptureConfig
Use with WLST: Online.

#### 20.2.8.1 Description
Displays the list of all Enterprise Capture configuration attributes with their values. If a value cannot be fetched, it will be shown as "VALUE CANNOT BE DISPLAYED". The most likely cause of an unfetched value is insufficient privilege.

#### 20.2.8.2 Syntax
```bash
listCaptureConfig()
```

#### 20.2.8.3 Example
The following example lists Enterprise Capture configuration attributes and values.

```bash
listCaptureConfig()
CaptureSystemID CAPTURE_01
BatchRefreshLimit 5
```

### 20.2.9 getCaptureConfig
Use with WLST: Online.

#### 20.2.9.1 Description
Fetches the value of the attribute specified as an argument. Use the listCaptureConfig command to list possible attribute names.

#### 20.2.9.2 Syntax
```bash
getCaptureConfig(['NAME_OF_ATTRIBUTE'])
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKSPACE_ID</td>
<td>ID number for a workspace. Can be optionally specified within single quotes. For example, both 3 and '3' are valid arguments for a workspace with ID 3.</td>
</tr>
</tbody>
</table>

#### 20.2.9.3 Example
The following example fetches the value of the attribute for the Enterprise Capture system ID.

```bash
getCaptureConfig('CaptureSystemID')
CAPTURE_01
```
20.2.10 setCaptureConfig

Use with WLST: Online.

20.2.10.1 Description
Sets the value of the Enterprise Capture attribute to the provided value.

20.2.10.2 Syntax

setCaptureConfig(['NAME_OF_ATTRIBUTE'], ['VALUE_OF_ATTRIBUTE'])

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME_OF_ATTRIBUTE</td>
<td>Name of Enterprise Capture configuration attribute to configure.</td>
</tr>
<tr>
<td>VALUE_OF_ATTRIBUTE</td>
<td>Value to set for the Enterprise Capture configuration attribute.</td>
</tr>
</tbody>
</table>

20.2.10.3 Example
The following example sets the CaptureSystemID attribute to the value CAPTURE_02.

setCaptureConfig('CaptureSystemID', 'CAPTURE_02')
Attribute 'CaptureSystemID' changed to 'CAPTURE_02'

20.2.11 scanForClientBundles

Use with WLST: Online.

20.2.11.1 Description
Scans the client bundle directory for bundles. Data for existing bundles will be updated and information for new bundles will be loaded and cached in the database.

20.2.11.2 Syntax

scanForClientBundles()

20.2.11.3 Example
The following example scans for bundles and lists processed bundles.

scanForClientBundles()
Processed capture bundle.jar

20.2.12 deleteBundle

Use with WLST: Online.

20.2.12.1 Description
Deletes an existing bundle.

20.2.12.2 Syntax

deleteBundle('bundlename')

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>bundlename</td>
<td>Name of an existing bundle to delete.</td>
</tr>
</tbody>
</table>
20.2.12.3 Example
The following example deletes the specified bundle.

`deleteBundle('Example')`

20.2.13 `setObjectProperty`
Use with WLST: Online.

20.2.13.1 Description
Sets the value of a property defined in a Capture Object, which could be a client profile, processor job, or a commit profile.

For information on the properties that can be set for each Capture object, see Appendix A.

20.2.13.2 Syntax
`setObjectProperty('<WORKSPACE_NAME>','<CLASS_NAME>','<OBJECT_IDENTIFIER>','<PROPERTY_ASSIGNMENT_STRING>')`

This command requires the following parameters:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKSPACE_NAME</td>
<td>The name of the Capture workspace that contains the objects that will be modified. If the calling user has rights to more than one workspace with the specified name, the first workspace found will be used.</td>
</tr>
<tr>
<td>CLASS_NAME</td>
<td>The name of the class that defines the client profile, processor job, or commit profile.</td>
</tr>
<tr>
<td>OBJECT_IDENTIFIER</td>
<td>A key/value pair consisting of the name of a property that can be used to locate the object and the expected value of that property.</td>
</tr>
<tr>
<td>PROPERTY_ASSIGNMENT_STRING</td>
<td>The name and value of the property to update, separated by an equal sign.</td>
</tr>
</tbody>
</table>

20.2.13.3 Points to Note
The following list provides some additional points to note when you use the `setObjectProperty` command:

- Modifiable objects expose their properties using the standard JavaBean getter/setter pattern. In the example below it is assumed the methods `getProfileDesc` and `setProfileDesc` exist in the Profile class.

- The value being assigned to the property should be compatible with the property’s data type. If a value is not compatible with the property’s data type, an exception will occur.

- The following are valid class names:
  - `oracle.oddc.data.Profile` (Capture object being modified: Client Profile)
  - `oracle.odc.recognition.RecognitionJob` (Capture object being modified: Recognition Processor Job)
  - `oracle.odc.importprocessor.ImportJob` (Capture object being modified: Import Processor Job)
  - `oracle.odc.entity.CommitProfileEntity` (Capture object being modified: Commit Profile)
Configuration Commands

- oracle.odc.docconverter.DocConverterJob (Capture object being modified: Document Conversion Job)

- In cases where a value needs to be set on an object nested within a profile or job, “dot” notation can be used to access the nested property:
  propertyName.propertyName=value.

- Multiple levels of nesting are supported. The following could be used to modify the barcodeName property of a Recognition Processor job’s docTypeBarcode property:
  docTypeBarcode.barcodeName=value.

- To support accessing a property of an object contained in a collection, the following “angle bracket” notation can be used:
  propertyName<identifierPropertyName=value>.propertyName=value.

  - The value between the angle brackets will be used to locate the object to modify within the collection. The identifierPropertyName is the name of a property that can be used to identify the object in the collection and value is the value that property should contain to be considered a match. For example:
    barcodes<barcodeName=barcode 1>.barcodeName=barcode 2.

20.2.13.4 Example

- The following is an example of <Object Identifier> that would apply to client profiles:
  profileName=Commit to CS

- The following is an example of <Property Assignment String> for updating the property profileDesc for a client profile:
  profileDesc=Test Profile Desc

- The following is an example of setting the property <serverURL> associated with a WebCenter Content commit profile named Commit to CS:
  setObjectProperty('TEST_WORKSPACE','corporate.odc.entity.CommitProfileEntity','profileName=Commit to CS','serverURL=http://myhost.example.com:16200/cs/idcplg/')

20.2.14 getObjectProperty

Use with WLST: Online.

20.2.14.1 Description

Prints the value of a specific property of a Capture Object (client profile, processor job, or commit profile).

20.2.14.2 Syntax

getObjectProperty('<WORKSPACE_NAME>','<CLASS_NAME>','<OBJECT_IDENTIFIER>','<PROPERTY_NAME>')</n

This method requires the following parameters:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKSPACE_NAME</td>
<td>The name of the Capture workspace that contains the object that will be modified. If the calling user has rights to more than one workspace with the specified name, the first workspace found will be used.</td>
</tr>
</tbody>
</table>
20.2.14.3 Example

■ The following are valid class names:
  oracle.oddc.data.Profile
  oracle.odc.recognition.RecognitionJob
  oracle.odc.importprocessor.ImportJob
  oracle.odc.entity.CommitProfileEntity
  oracle.odc.docconverter.DocConverterJob

■ The following is an example of <Object Identifier> that would apply to import jobs:
  jobName=Import Email

■ The following is an example for retrieving the value of the property <serverName> associated with an email import job.
  getObjectProperty('TEST-WORKSPACE','oracle.odc.importprocessor.ImportJob','jobName=Import Email','serverName')

20.2.15 setObjectCredentials

Use with WLST: Online.

20.2.15.1 Description

It is used to set the user name and password associated with a profile or job.

20.2.15.2 Syntax

setObjectCredentials('<WORKSPACE_NAME>','<CLASS_NAME>','<OBJECT_IDENTIFIER>','<USER_NAME>','<PASSWORD>')

This method requires the following parameters:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKSPACE_NAME</td>
<td>The name of the Capture workspace that contains the credential that will be modified. If the calling user has rights to more than one workspace with the specified name, the first workspace found will be used.</td>
</tr>
<tr>
<td>CLASS_NAME</td>
<td>The name of the class that defines the profile or the job.</td>
</tr>
</tbody>
</table>
20.2.15.3 Example

- Valid class names include the following:
  
  oracle.odc.importprocessor.ImportJob
  oracle.odc.entity.CommitProfileEntity

- The following is an example for setting credentials associated with an Email Import Job:
  
  setObjectCredentials('TEST-WORKSPACE','oracle.odc.importprocessor.ImportJob','jobName=import Email','xyz@example.com','welcome123')

- The following is an example for setting credentials associated with a WebCenter Imaging commit profile:
  
  setObjectCredentials('TEST-WORKSPACE','oracle.odc.entity.CommitProfileEntity','profileName=commit to Imaging','admin','welcome123')

20.2.16 updateScript

Use with WLST: Online.

20.2.16.1 Description

Updates a script within a workspace.

20.2.16.2 Syntax

updateScript('<WORKSPACE_NAME>','<SCRIPT_NAME>','<SCRIPT_FILE>')</n

This command requires the following parameters:

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>WORKSPACE_NAME</td>
<td>The name of the workspace that contains the script. The name should be used to lookup the unique identifier of the workspace. If the name is not found, an error message is displayed.</td>
</tr>
<tr>
<td>SCRIPT_NAME</td>
<td>The name of the script to update. If the script cannot be found, an error message is displayed.</td>
</tr>
<tr>
<td>SCRIPT_FILE</td>
<td>The absolute path to the script file that should be updated. The script should be updated with the contents of this file.</td>
</tr>
</tbody>
</table>

20.2.16.3 Example

The following example updates a script within a workspace:

updateScript('TEST-WORKSPACE','Client_Script','/scratch/aime/scripts/client.txt')
This chapter describes Oracle Enterprise Scheduler custom WLST commands for getting job request content, managing requests, and managing server and configuration. It includes command syntax, arguments and examples.

Use the Oracle Enterprise Scheduler commands in the categories listed in Table 21–1 to manage configuration, servers, logs, and job requests.

---

**Note:** To use these custom WLST commands, you must invoke the WLST script from the Oracle Common home. See "Using Custom WLST Commands" in the Oracle Fusion Middleware Administrator’s Guide.

---

When running these WLST commands, you must have the following JARs on your classpath:
- `MW_HOME/oracle_common/modules/oracle.jmx_11.1.1/jmxframework.jar`
- `WL_HOME/server/lib/weblogic.jar`
- `MW_HOME/ORACLE_HOME/ess/lib/ess-admin.jar`

### 21.1 Enterprise Scheduler Custom Commands

Use the Enterprise Scheduler commands listed in Table 21–1 to manage the Enterprise Scheduler server, configuration, job requests, and logs. In the **Use with WLST** column, "online" means the command can only be used when connected to a running administration server. "Offline" means the command can only be used when not connected to a running server.

**Table 21–1  Oracle Enterprise Scheduler Management Commands**

<table>
<thead>
<tr>
<th>Use this command...</th>
<th>To...</th>
<th>Use with WLST...</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>essGetRequestContent</code></td>
<td>Get the log and output data files for a request after its execution is completed.</td>
<td>Online</td>
</tr>
<tr>
<td><code>essManageRequest</code></td>
<td>Cancel, recover, or complete request state manually.</td>
<td>Online</td>
</tr>
<tr>
<td><code>essManageRuntimeConfig</code></td>
<td>Add, modify, delete and display various configuration parameters.</td>
<td>Online</td>
</tr>
</tbody>
</table>
### 21.1.1 essGetRequestContent

**Command Category:** ESS  
**Use with WLST:** Online

#### Description
Get the log and output data files for a request after its execution is completed.

#### Syntax

```plaintext
essGetOutputContent(requestId, contentType, logLines, outDir)
```

#### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestId</td>
<td>The request ID.</td>
</tr>
<tr>
<td>contentType</td>
<td>Type of the content to handle. Can be LOG, OUTPUT, BINARY_OUTPUT or TEXT_OUTPUT. By default, the OUTPUT contentType checks for both BINARY_OUTPUT and TEXT_OUTPUT contents.</td>
</tr>
<tr>
<td>logLines</td>
<td>Optional. The number of lines to be read from the request log. Default is 1000.</td>
</tr>
<tr>
<td>outDir</td>
<td>Optional. The absolute path of the output directory to dump the output files into. Default is the current directory.</td>
</tr>
</tbody>
</table>

#### Examples

To get the request log for request ID 123.

```plaintext
essGetOutputContent(123, "LOG")
```

To get all the output of request 123.

```plaintext
essGetOutputContent(123, "OUTPUT")
```

To get all the output of request 123 and save it in directory /tmp.

```plaintext
essGetOutputContent(123, "OUTPUT", outDir="/tmp")
```

To get all the text output of request 123 and save it in directory /tmp.

```plaintext
essGetOutputContent(123, "TEXT_OUTPUT", outDir="/tmp")
```

To get all the binary output of request 123 and save it in directory /tmp.

```plaintext
essGetOutputContent(123, "BINARY_OUTPUT", outDir="/tmp")
```

To get first 100 lines of the request log for request id 123.

```plaintext
essGetOutputContent(123, "LOG", logLines=100)
```
21.1.2 essManageRequest

Command Category: ESS
Use with WLST: Online

21.1.2.1 Description
Cancel, recover, or complete request state manually.

21.1.2.2 Syntax
essManageRequest(requestId, operation, asyncStatus, statusMessage)

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestId</td>
<td>The request ID.</td>
</tr>
<tr>
<td>operation</td>
<td>The operation to perform: CANCEL, RECOVER, or COMPLETE.</td>
</tr>
<tr>
<td>asyncStatus</td>
<td>Mandatory when the COMPLETE operation is specified. The status to set for the given request. Must be one of the following:</td>
</tr>
<tr>
<td></td>
<td>■ BIZ_ERROR</td>
</tr>
<tr>
<td></td>
<td>■ CANCEL</td>
</tr>
<tr>
<td></td>
<td>■ ERROR</td>
</tr>
<tr>
<td></td>
<td>■ ERROR_MANUAL_RECOVERY</td>
</tr>
<tr>
<td></td>
<td>■ PAUSE</td>
</tr>
<tr>
<td></td>
<td>■ SUCCESS</td>
</tr>
<tr>
<td></td>
<td>■ UPDATE</td>
</tr>
<tr>
<td></td>
<td>■ WARNING</td>
</tr>
<tr>
<td>statusMessage</td>
<td>Optional. The qualifying status message to describe the operation.</td>
</tr>
</tbody>
</table>

21.1.2.3 Examples
To cancel request 123.

    essManageRequest(123, "CANCEL")

To recover request 123.

    essManageRequest(123, "RECOVER")

To complete request 123.

    essManageRequest(123, "COMPLETE", asyncStatus="ERROR", statusMessage="Completed by Admin")

21.1.3 essManageRuntimeConfig

Command Category: ESS
Use with WLST: Online

21.1.3.1 Description
Add, modify, delete and display various configuration parameters.

21.1.3.2 Syntax
essManageRuntimeConfig(app, type, operation, name, val)
### 21.1.3.3 Examples

To add an ENV parameter "foo" with value "bar".

```
essManageRuntimeConfig('myapp', 'APP', operation="add", name="foo", val="bar")
```

To get the value of the ENV parameter "foo".

```
essManageRuntimeConfig('myapp', 'APP', operation="get", name="foo")
```

To get the list of all the ENV parameters.

```
essManageRuntimeConfig('myapp', 'APP', operation="getall")
```

To modify the value of the ENV parameter "foo" to "barone".

```
essManageRuntimeConfig('myapp', 'APP', operation="mod", name="foo", val="barone")
```

To delete the ENV parameter "foo".

```
essManageRuntimeConfig('myapp', 'APP', operation="del", name="foo")
```

To show all parameters of type ESS.

```
essManageRuntimeConfig('myapp', 'ESS')
```

### 21.1.4 essManageServer

Command Category: ESS

Use with WLST: Online

#### 21.1.4.1 Description

Start, stop or get status of the Enterprise Scheduler application running on the server. Starting the Enterprise Scheduler application means to start the Enterprise Scheduler processor thread so that request processing can start. Stopping Enterprise Scheduler means to stop or quiesce the Enterprise Scheduler processor so that no new requests are processed.

If connected to the WLS Administration Server in a cluster, this command would operate upon all nodes in the cluster.

#### 21.1.4.2 Syntax

```
essManageServer(operation)
```
21.1.4.3 Examples
To stop Enterprise Scheduler.

```java
essManageServer('STOP')
```

To get the current state of the Enterprise Scheduler processor.

```java
essManageServer('STATUS')
```

21.1.5 `essQueryRequests`

Command Category: ESS

Use with WLST: Online

21.1.5.1 Description
Search and list requests based on hosting application name, state or elapsed time of execution. This command can be used to find long running requests.

21.1.5.2 Syntax

```java
essQueryRequests(app, state, days, hours, minutes)
```

<table>
<thead>
<tr>
<th>Argument</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>Optional. The name of the hosting application.</td>
</tr>
</tbody>
</table>
| state    | Optional. The request state. Can be one of the following (default is `RUNNING`):
  | ■ BLOCKED
  | ■ CANCELLED
  | ■ CANCELLING
  | ■ COMPLETED
  | ■ ERROR
  | ■ ERROR_AUTO_RETRY
  | ■ ERROR_MANUAL_RECOVERY
  | ■ EXPIRED
  | ■ FINISHED
  | ■ HOLD
  | ■ PAUSED
  | ■ PENDING_VALIDATION
  | ■ READY
  | ■ RUNNING
  | ■ SCHEDULE_ENDED
  | ■ SUCCEEDED
  | ■ VALIDATION_FAILED
  | ■ WAIT
  | ■ WARNING |
### 21.1.5.3 Examples

To get all the requests in RUNNING state.

```sh
essQueryRequests()
essQueryRequests{state="RUNNING"}
```

To get all CANCELLED requests.

```sh
essQueryRequests{state="CANCELLED"}
```

To get all requests running for more than 2 days.

```sh
essQueryRequests{days=2}
```

To get all requests running for more than 10 hours.

```sh
essQueryRequests{hours=10}
```

To get all requests running for the application "myapp".

```sh
essQueryRequests{appName="myapp"}
```
A

Capture Object Properties

This appendix lists all the properties that can be set for each Capture object using the setObjectProperty command.

This appendix contains the following sections:

- Section A.1, "Client Profile Object Properties"
- Section A.2, "Import Processor Job Object Properties"
- Section A.3, "Recognition Processor Job Object Properties"
- Section A.4, "Commit Profile Object Properties"
- Section A.5, "Document Conversion Job Object Properties"

A.1 Client Profile Object Properties

The following table lists all the properties that can be set for a Client Profile object (class name: oracle.oddc.data.Profile).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profile Name</td>
<td>profileName</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Description</td>
<td>profileDesc</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Online</td>
<td>profileStatus</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Apply Default Brightness and Contrast</td>
<td>applyBrightness</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Batch Prefix</td>
<td>batchPrefix</td>
<td>String</td>
<td>Any value</td>
</tr>
</tbody>
</table>
| Profile Type              | scanningType                 | int       | 1 - Capture only  
  2 - Capture and Index  
  3 - Index only          |
<p>|                           | maxPages                     | int       | Any value     |
| Default Status            | defaultStatus                | String    | Any value     |
| Batch Priority            | defaultPriority              | int       | Any value     |</p>
<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Document Creation Option</td>
<td>docOption</td>
<td>int</td>
<td>1 - One page (for example, Simplex) 2 - Two pages (for example, Duplex) 3 - Variable number of pages 4 - Prompt the user</td>
</tr>
<tr>
<td>Separator Sheet Byte Threshold</td>
<td>sepByteThreshold</td>
<td>int</td>
<td>Any value</td>
</tr>
<tr>
<td>Blank Page Byte Threshold:</td>
<td>blankByteThreshold</td>
<td>long</td>
<td>Any value</td>
</tr>
<tr>
<td>Database Lookup Profile</td>
<td>dbLookupProfile</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Always Display Records</td>
<td>dbLookupHitList</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Dependent Choice List</td>
<td>pickListRelationshipProfile</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Records Returned Limit</td>
<td>dbLookupMaxRecords</td>
<td>int</td>
<td>Any value</td>
</tr>
<tr>
<td>Brightness</td>
<td>brightness</td>
<td>int</td>
<td>Any value</td>
</tr>
<tr>
<td>Contrast</td>
<td>contrast</td>
<td>int</td>
<td>Any value</td>
</tr>
<tr>
<td>Batch Visibility</td>
<td>batchVisibility</td>
<td>int</td>
<td>0 - User and workstation 1 - User 2 - All users</td>
</tr>
<tr>
<td>Non-Image File Import Action</td>
<td>nonImageImportAction</td>
<td>int</td>
<td>0 - Do not import 1 - Import in native format 2 - Convert to image format</td>
</tr>
<tr>
<td>Additional Batch Prefixes (separate by ;)</td>
<td>filterPrefix</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Days Old: From</td>
<td>filterDaysFrom</td>
<td>int</td>
<td>Any value</td>
</tr>
<tr>
<td>Days Old: To</td>
<td>filterDaysTo</td>
<td>int</td>
<td>Any value</td>
</tr>
<tr>
<td>Batch Statuses</td>
<td>filterStatus</td>
<td>List&lt;String&gt;</td>
<td>Any value</td>
</tr>
<tr>
<td>Priorities</td>
<td>filterPriority</td>
<td>List&lt;Integer&gt;</td>
<td>[0,1,2,3,4,5,6,7,8,9,10]</td>
</tr>
<tr>
<td>Primary Sort</td>
<td>filterPrimarySort</td>
<td>String</td>
<td>BatchName/BatchPageCount/BatchDate/BatchPriority/BatchStatus</td>
</tr>
<tr>
<td>Primary Sort Order</td>
<td>filterPrimaryOrder</td>
<td>int</td>
<td>0 - Ascending 1 - Descending</td>
</tr>
<tr>
<td>Secondary Sort</td>
<td>filterSecondarySort</td>
<td>String</td>
<td>BatchName/BatchPageCount/BatchDate/BatchPriority/BatchStatus</td>
</tr>
<tr>
<td>Secondary Sort Order</td>
<td>filterSecondaryOrder</td>
<td>int</td>
<td>Any value</td>
</tr>
</tbody>
</table>
A.2 Import Processor Job Object Properties

The following section lists all the properties that can be set for an Import Processor Job object (class name: oracle.odc.importprocessor.ImportJob).

This section contains the following topics:

- General Import Job Properties
- Folder Import Job Specific Properties
- List File Import Job Specific Properties
- Email Job Specific Properties

A.2.1 General Import Job Properties

The following table lists the General Import Job properties that can be set (use class name: oracle.odc.importprocessor.ImportJob).
<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online</td>
<td>isJobOnline</td>
<td>Boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Import Job Name</td>
<td>jobName</td>
<td>String</td>
<td>Any value</td>
</tr>
</tbody>
</table>
| Image Down-Sample        | imageDownsample              | Integer   | 0 - None (retain image format)  
                           |               |            | 1 - Down-sample color to 8 bit (gray scale)  
                           |               |            | 2 - Down-sample color or gray scale to black and white |
| JPEG Image Quality       | jpegQuality                  | Integer   | Any value     |
| Batch Prefix             | batchPrefix                  | String    | Any value     |
| Default Batch Status     | defaultBatchStatusID         | String    | Any value     |
| Default Batch Priority   | defaultPriority              | Integer   | Any value     |
| Database Lookup Profile  | dbSearchID                   | String    | Any value     |
| Database Search Field    | dbSearchFieldID              | String    | Any value     |
| Default Document Profile | defaultDocumentTypeID        | String    | Any value     |
| When more than one record is found | searchResultOption | Integer       | 0 - Use first record  
                           |               |            | 1 - Do not select a record |
| Script                   | scriptID                     | String    | Any value     |
| Import Frequency         | importFrequency              | Integer   | 15: 15 seconds  
                           |               |            | 30: 30 seconds  
                           |               |            | 60: 1 minute  
                           |               |            | 300: 5 minutes  
                           |               |            | 900: 15 minutes  
                           |               |            | 1800: 30 minutes  
                           |               |            | 3600: 1 hour  
                           |               |            | -1: Daily |
| Import Source            | importSourceClassName        | String    | oracle.odc.importprocessor.folder.FolderSource  
                           |               |            | /oracle.odc.importprocessor.email.EmailSource  
                           |               |            | /oracle.odc.importprocessor.listfile.ListFileSource |
### Import Processor Job Object Properties

The following table lists the Folder Import Job properties that can be set (use class name: oracle.odc.importprocessor.ImportJob).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Batch Processor Job</td>
<td>batchProcessorJobID</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Import Freq: Every Day: Time Hr:</td>
<td>hour</td>
<td>Integer</td>
<td>Any value</td>
</tr>
<tr>
<td>Import Freq: Every Day: Time Min:</td>
<td>minute</td>
<td>Integer</td>
<td>Any value</td>
</tr>
<tr>
<td>Default Locale</td>
<td>locale</td>
<td>Locale</td>
<td>Any value</td>
</tr>
<tr>
<td>If Image Validation Fails:</td>
<td>imageFailureAction</td>
<td>Integer</td>
<td>0 - Delete the batch 1 - Skip the file</td>
</tr>
<tr>
<td>Default Date Format</td>
<td>defaultDateFormat</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Description</td>
<td>description</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Encoding</td>
<td>encoding</td>
<td>String</td>
<td>Any value</td>
</tr>
</tbody>
</table>

### A.2.2 Folder Import Job Specific Properties

The following table lists the Folder Import Job properties that can be set (use class name: oracle.odc.importprocessor.ImportJob).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import Folder Path</td>
<td>folder</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Process subfolders</td>
<td>processSubfolders</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Create a New Batch</td>
<td>batchCreationOption</td>
<td>int</td>
<td>0 - Per file 1 - Per folder</td>
</tr>
<tr>
<td>File Mask(s)</td>
<td>fileMasks</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Primary Sort Type</td>
<td>primarySortType</td>
<td>FolderSortType</td>
<td>0 - None 1 - File Name 2 - File Extension 3 - File Modified Date</td>
</tr>
<tr>
<td>Primary Sort Order</td>
<td>primarySortOrder</td>
<td>FolderSortOrder</td>
<td>0 - Ascending 1 - Descending</td>
</tr>
<tr>
<td>Secondary Sort Type</td>
<td>secondarySortType</td>
<td>FolderSortType</td>
<td>0 - None 1 - File name 2 - File extension 3 - File modified date</td>
</tr>
<tr>
<td>Secondary Sort Order</td>
<td>secondarySortOrder</td>
<td>FolderSortOrder</td>
<td>0(Ascending) / 1(Descending)</td>
</tr>
</tbody>
</table>
### A.2.3 List File Import Job Specific Properties

The following table lists the List File Import Job properties that can be set (use class name: `oracle.odc.importprocessor.ImportJob`).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ready File</td>
<td>readyFile</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>After Import</td>
<td>postProcessingOption</td>
<td>int</td>
<td>0 - Delete file</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 - Change file extension</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 - Add file prefix</td>
</tr>
<tr>
<td>New Extension</td>
<td>fileExtension</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Prefix</td>
<td>filePrefix</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Delete processed subfolder if empty</td>
<td>deleteSubfolders</td>
<td>boolean</td>
<td>Any value</td>
</tr>
</tbody>
</table>

### A.2.4 Email Job Specific Properties

The following table lists the Email Job properties that can be set (use class name: `oracle.odc.importprocessor.ImportJob`).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import Folder Path</td>
<td>folder</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Create a New Batch:</td>
<td>batchCreationOption</td>
<td>int</td>
<td>0 - Per list file</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 - Per folder</td>
</tr>
<tr>
<td>File Mask(s)</td>
<td>fileMasks</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Process subfolders</td>
<td>processSubfolders</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Field Delimiter</td>
<td>fieldDelimiter</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Maximum Fields Per Document</td>
<td>maxFieldsPerRecord</td>
<td>int</td>
<td>Any value</td>
</tr>
<tr>
<td>Document File Field Position</td>
<td>documentFieldPosition</td>
<td>int</td>
<td>Any value</td>
</tr>
<tr>
<td>After Import</td>
<td>postProcessingOption</td>
<td>int</td>
<td>0 - Delete file</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 - Change file extension</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 - Add file prefix</td>
</tr>
<tr>
<td>New Extension</td>
<td>fileExtension</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Prefix</td>
<td>filePrefix</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Delete document files after successful import</td>
<td>deleteDocumentFiles</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Property Name (on the UI)</td>
<td>Property Name (in the Class)</td>
<td>Data Type</td>
<td>Valid Options</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-----------------------------------------------</td>
<td>---------------</td>
<td>-----------------------------------</td>
</tr>
</tbody>
</table>
| IMAP Connection Security                      | imapConnectSecurity                           | EmailConnectionSecurity | 0 - NONE  
|                                                    |                                               |                | 1 - STARTTLS                         |
|                                                    |                                               |                | 2 - SSL_TLS                           |
| Email Server Name (DNS name or IP address)     | serverName                                    | String         | Any value                          |
| Port                                            | port                                          | int            | Any value                          |
| Folders to Process                             | folderList                                    | String         | Any value                          |
| Message Filters: From Address                  | filterFromAddressEnabled                      | boolean        | Any value                          |
| Message Filters: From Address Field Contains   | filterFromAddress                             | String         | Any value                          |
| Message Filters: Subject                       | filterSubjectEnabled                          | boolean        | Any value                          |
| Message Filters: Subject Field Contains        | filterSubject                                 | String         | Any value                          |
| Message Filters: Message Body                  | filterMessageBodyEnabled                      | boolean        | Any value                          |
| Message Filters: Message Body Field Contains   | filterMessageBody                             | String         | Any value                          |
| Include attachments matching these mask(s)     | attachmentsMask                               | String         | Any value                          |
| Exclude attachments matching these mask(s)     | attachmentsExcludeMask                        | String         | Any value                          |
| Email Message Options                          | includeMessageBodyOption                      | int            | 0 - Import as text  
|                                                    |                                               |                | 1 - Import in EML format            |
| Import message body file                       | includeMessageBodyFile                        | boolean        | Any value                          |
| Import entire message in EML format            | includeEntireEMLFile                          | boolean        | Any value                          |
| Document Ordering                              | emailDocumentOrder                            | List<Integer>  | 0 - Message body (text file)  
|                                                    |                                               |                | 1 - Message body (EML file)          |
|                                                    |                                               |                | 2 - Email message (EML file)       |
|                                                    |                                               |                | 3 - Attachment(s)                   |
| Include when no attachments exist              | includeMessageBodyWithoutAttachments          | boolean        | Any value                          |
A.3 Recognition Processor Job Object Properties

The following table lists all the properties that can be set for a Recognition Processor Job object (class name: oracle.odc.recognition.RecognitionJob).

This section contains the following topics:

- General Recognition Job Properties
- BarcodeDefinition Class Properties
- DocumentDefinition Class Properties
- RecognitionJobField Class Properties
- SeparatorDefinition Class Properties

### A.3.1 General Recognition Job Properties

The following table lists the General Recognition Job properties that can be set (use class name: oracle.odc.recognition.RecognitionJob).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
</table>
| Upon Failed Import        | postProcessingFailureOption | int       | 0 - Delete the message  
                                 |                              |           | 1 - Move the message to another folder  
                                 |                              |           | 2 - Do not delete the message |
| Upon Successful Import:   | postProcessingSuccessOption | int       | 0 - Delete message  
                                 |                              |           | 1 - Move message to another folder |
| Folder Name:              | postProcessingFailureMoveFolder | String   | Any value |
| Folder Name               | postProcessingSuccessMoveFolder | String   | Any value |
| Search Operator           | searchOperator              | int       | 0 - AND  
                                 |                              |           | 1 - OR |
| Determined based on email importance | determineBatchPriority | boolean  | Any value |

Email Importance Low: Batch Priority

- batchPriorityLow | int | Any value |

Email Importance Normal: Batch Priority

- batchPriorityNormal | int | Any value |

Email Importance High: Batch Priority

- batchPriorityHigh | int | Any value |

Email Accounts to Process

- emailAccounts | List<String> | Any value |
<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recognition Job Name</td>
<td>name</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Description</td>
<td>description</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Script</td>
<td>scriptID</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Online</td>
<td>online</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Bar Codes</td>
<td>barcodes</td>
<td>List&lt;BarcodeDefinition&gt;</td>
<td>Any value</td>
</tr>
<tr>
<td>Enable Auto-Detect</td>
<td>autoDetectBarcodes</td>
<td>Boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Validate optional checksum</td>
<td>validateCheckSum</td>
<td>Boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Bar Code Recognition</td>
<td>symbologies</td>
<td>List&lt;Integer&gt;</td>
<td>[0,1,2,3,4,5,6,7,8,9] (Auto-Detectable 1-D Symbologies) / [10,11,12,13,14,15,16,17,18] (Other 1-D Symbologies) / [19,20,21] (2-D Symbologies)</td>
</tr>
<tr>
<td>Organize documents based on</td>
<td>batchOrganization</td>
<td>Integer</td>
<td>0 - Fixed number of pages per document 1 - None: Do not perform document organization 2 - Same bar code value on each page 3 - Separator pages 4 - Hierarchical separator pages</td>
</tr>
<tr>
<td>Number of Pages Per Document:</td>
<td>documentPageCount</td>
<td>Integer</td>
<td>Any value</td>
</tr>
<tr>
<td>Number of pages per document to read bar codes:</td>
<td>pagesPerDoc2ReadBarcodes</td>
<td>Integer</td>
<td>Any value</td>
</tr>
<tr>
<td>Maximum Number of Pages Per Document:</td>
<td>maxPageCountPerDoc</td>
<td>Integer</td>
<td>Any value</td>
</tr>
<tr>
<td>Optimistic bar code detection</td>
<td>optimizeBarcodeDetection</td>
<td>Boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Separator Page:</td>
<td>coverPages</td>
<td>List&lt;SeparatorDefinition&gt;</td>
<td>Any value</td>
</tr>
<tr>
<td>If more than one value is found for a bar code within a document</td>
<td>multiBarcodeValuesOption</td>
<td>Integer</td>
<td>0 - Use the first value found 1 - Always overwrite the bar code value 2 - Clear the bar code value</td>
</tr>
<tr>
<td>Property Name (on the UI)</td>
<td>Property Name (in the Class)</td>
<td>Data Type</td>
<td>Valid Options</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------------------------</td>
<td>-----------</td>
<td>---------------</td>
</tr>
<tr>
<td>Source Document Attachments:</td>
<td>sourceDocAttachments</td>
<td>Integer</td>
<td>0 - Include all attachments to created documents 1 - Include attachments with matching Document Profile attachment types 2 - Do not include attachments</td>
</tr>
<tr>
<td>Dynamic Document Profile Option:</td>
<td>dynamicDocType</td>
<td>Integer</td>
<td>0 - Do not determine dynamically 1 - Determine dynamically using bar code 2 - Determine dynamically using separator page</td>
</tr>
<tr>
<td>Default Document Profile:</td>
<td>defaultDocTypeID</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Document Profile and Bar Code Value Mappings</td>
<td>docTypeMappings</td>
<td>List&lt;Definition&gt;</td>
<td>Any value</td>
</tr>
<tr>
<td>Fields</td>
<td>jobFields</td>
<td>List&lt;RecognitionJobField&gt;</td>
<td>Any value</td>
</tr>
<tr>
<td>DB Lookup Using:</td>
<td>dblookupUsing</td>
<td>Integer</td>
<td>0 - None 1 - Bar code 2 - Field value</td>
</tr>
<tr>
<td>Field Name</td>
<td>dblookupIndexDefID</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Database Lookup Profile</td>
<td>dblookupProfile</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Database Search Field:</td>
<td>dblookupSearchField</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>When more than one record is found:</td>
<td>dblookupMultipleRecordAction</td>
<td>Integer</td>
<td>0 - Use first record 1 - Do not link</td>
</tr>
<tr>
<td>When no record is found:</td>
<td>dblookupNoMatchAction</td>
<td>Integer</td>
<td>0 - Allow document commit 1 - Prevent document commit</td>
</tr>
<tr>
<td>To rename the batch, enter a prefix:</td>
<td>renamePrefix</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>To send email notification, enter email address:</td>
<td>renameEmail</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>To change batch status, select a status:</td>
<td>renameStatus</td>
<td>String</td>
<td>Any value</td>
</tr>
</tbody>
</table>
### A.3.2 BarcodeDefinition Class Properties

The following table lists the BarcodeDefinition properties that can be set (use class name: oracle.odc.recognition.RecognitionJob).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>barcodeName</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Validation Rules</td>
<td>validationRule</td>
<td>Integer</td>
<td>0 - None 1 - Length 2 - Mask 3 - Regular expression 4 - Choice list</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>validationLength</td>
<td>Integer</td>
<td>Any value</td>
</tr>
<tr>
<td></td>
<td>validationMask</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td></td>
<td>validationRegularExpression</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td></td>
<td>pickListSourceID</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td></td>
<td>pickListID</td>
<td>String</td>
<td>Any value</td>
</tr>
</tbody>
</table>

### A.3.3 DocumentDefinition Class Properties

The following table lists the DocumentDefinition properties that can be set (use class name: oracle.odc.recognition.RecognitionJob).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>To change batch priority, enter a priority between 1 to 10:</td>
<td>renamePriority</td>
<td>Integer</td>
<td>Number between 1 to 10</td>
</tr>
<tr>
<td>Batch Processor</td>
<td>processorID</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Batch Processor Job:</td>
<td>processorJobID</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>To rename the batch, enter a prefix:</td>
<td>failureRenamePrefix</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>To send email notification, enter email address:</td>
<td>failureRenameEmail</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>To change batch status, select a status:</td>
<td>failureRenameStatus</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>To change batch priority, enter a priority between 1 to 10:</td>
<td>failureRenamePriority</td>
<td>Integer</td>
<td>Number between 1 to 10</td>
</tr>
<tr>
<td>Batch Processor</td>
<td>failureProcessorID</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Batch Processor Job:</td>
<td>failureProcessorJobID</td>
<td>String</td>
<td>Any value</td>
</tr>
</tbody>
</table>
The following table lists the RecognitionJobField properties that can be set (use class name: oracle.odc.recognition.RecognitionJob).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto Populate</td>
<td>autoPopulate</td>
<td>Integer</td>
<td>0 - None</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 - Bar code</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 - Batch name</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3 - Default</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4 - Index date</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5 - Scan date</td>
</tr>
<tr>
<td>Barcode</td>
<td>populateValue</td>
<td>String</td>
<td>Any value</td>
</tr>
</tbody>
</table>

The following table lists the SeparatorDefinition properties that can be set (use class name: oracle.odc.recognition.RecognitionJob).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule Name</td>
<td>name</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Delete this separator page upon commit</td>
<td>deleteUponCommit</td>
<td>Boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Operator</td>
<td>operator</td>
<td>Integer</td>
<td>0 - OR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 - AND</td>
</tr>
<tr>
<td>Document Profile</td>
<td>docTypeID</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Attachment Type</td>
<td>attachmentTypeID</td>
<td>String</td>
<td>Any value</td>
</tr>
</tbody>
</table>

The following table lists all the properties that can be set for a Commit Profile object (class name: oracle.odc.entity.CommitProfileEntity).

This section contains the following topics:
- General Commit Profile Properties
- Text File Commit Profile Specific Properties
- Webcenter Content Commit Profile Specific Properties
- ContentAttributeMappingInfo Class Properties
- CaptureToContentFieldMapping Class Properties
Commit Profile Object Properties

- ContentFieldNameInfo Class Properties
- Webcenter Imaging Commit Profile Specific Properties
- PDF Searchable Document Output Specific Properties
- DOCS Commit Driver Properties

### A.4.1 General Commit Profile Properties

The following table lists the General Commit Profile properties that can be set (use class name: oracle.odc.entity.CommitProfileEntity).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commit Profile Name</td>
<td>profileName</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Document Output Format</td>
<td>exportDriverID</td>
<td>String</td>
<td>oracle.odc.export.PDFSearchable/oracle.odc.export.PDFImage/ oracle.odc.export.TIFFMultiImage</td>
</tr>
<tr>
<td>Commit Driver</td>
<td>commitDriverID</td>
<td>String</td>
<td>oracle.odc.commit.TextFileCommitDriver(TextFile)/ oracle.odc.commit.cs.ContentCommitDriver(WebCenter Content)/ oracle.odc.commit.ipm.IPMCommitDriver(WebCenter Content Imaging)</td>
</tr>
<tr>
<td>Online</td>
<td>active</td>
<td>Boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Edit Execution Order</td>
<td>executionOrder</td>
<td>Integer</td>
<td>Any value</td>
</tr>
<tr>
<td>Error Handling Policy</td>
<td>errorHandlingPolicy</td>
<td>Integer</td>
<td>0 - Continue to the next document</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 - Cancel to next commit profile</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2 - Cancel commit</td>
</tr>
<tr>
<td>Restrict Commit to Document Profiles</td>
<td>documentTypes</td>
<td>List&lt;Docu-</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>mentTypeEntity&gt;</td>
<td></td>
</tr>
<tr>
<td>Default Locale</td>
<td>defaultLocaleString</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Default Date Format</td>
<td>defaultDateFormat</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Encoding</td>
<td>encoding</td>
<td>String</td>
<td>Any value</td>
</tr>
</tbody>
</table>

**Note:** To update the Default Locale field associated with a commit profile, pass a value that includes all the three parts of a locale, that is, language, country, and variant. Use a vertical bar (|) to separate the parts of the locale. The following example sets the locale to Chinese (China): defaultLocaleString = zh|CN|.
## A.4.2 Text File Commit Profile Specific Properties

The following table lists the Text File Commit Profile properties that can be set (use class name: oracle.odc.entity.CommitProfileEntity).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Do not create Commit Text File</td>
<td>doNotCreateTextFile</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Store in subfolders</td>
<td>useTextSubFolder</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Text File Folder: Subfolder Options</td>
<td>textSubFolderOption</td>
<td>int</td>
<td>0 - Year, 1 - Year and month, 2 - Year, month, and date</td>
</tr>
<tr>
<td>Commit Text File Folder</td>
<td>textFileFolder</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>File Prefix</td>
<td>textFilePrefix</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>File Extension</td>
<td>textFileExtension</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Create a copy for each page</td>
<td>duplicateLinkedItems</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Create a folder per committed batch</td>
<td>oneFolderPerBatch</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Store in subfolders</td>
<td>useDocSubFolder</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Document Folder: Subfolder Options</td>
<td>docSubFolderOption</td>
<td>int</td>
<td>0 - Year, 1 - Year and month, 2 - Year, month, and date, 3 - Metadata field(s)</td>
</tr>
<tr>
<td>Document Folder</td>
<td>docFolder</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Selected Metadata Fields</td>
<td>subFolderIndexes</td>
<td>List&lt;String&gt;</td>
<td>Any value</td>
</tr>
<tr>
<td>SubFolder Path</td>
<td>subFolderPath</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Field Delimiter</td>
<td>delimiterOption</td>
<td>int</td>
<td>0 - Comma, 1 - Semicolon, 2 - Other</td>
</tr>
<tr>
<td>Text Qualifier</td>
<td>textQualifier</td>
<td>int</td>
<td>0 - None, 1 - Double quote, 2 - Single quote</td>
</tr>
<tr>
<td>Other Character</td>
<td>formatDelimiter</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Text Qualifier</td>
<td>formatFields</td>
<td>List&lt;String&gt;</td>
<td>Any value</td>
</tr>
<tr>
<td>Name document file based on metadata field values</td>
<td>nameDoc</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>If File Name Consists of Invalid Characters</td>
<td>docNameInvalidCharOption</td>
<td>int</td>
<td>0 - Remove invalid characters, 1 - Cancel document commit</td>
</tr>
</tbody>
</table>
A.4.3 Webcenter Content Commit Profile Specific Properties

The following table lists the Webcenter Content Commit Profile properties that can be set (use class name: oracle.odc.entity.CommitProfileEntity).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server URL</td>
<td>serverURL</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Name document file based on metadata values</td>
<td>documentFileNaming</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Document File Name</td>
<td>documentFilename</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>If file name consists of invalid characters:</td>
<td>removeDocFilenameInvalidChars</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Document Title</td>
<td>documentTitle</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Default Account</td>
<td>documentAccount</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Default Security Group</td>
<td>documentSecurityGroup</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Default Type</td>
<td>documentType</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Assign Values Dynamically</td>
<td>dynamicAssignment</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Assign Value</td>
<td>dynamicMappingOption</td>
<td>int</td>
<td>0 - By field mappings 1 - By metadata field</td>
</tr>
</tbody>
</table>

| Metadata Field | dynamicMappingIndexField | String | Any value |
| Choice List Source | dynamicMappingPicklistSourceId | String | Any value |
| Choice List | dynamicMappingPicklistId | String | Any value |
| Account | accountMappingField | String | Any value |
| Type | contentTypeMappingField | String | Any value |
| Security Group | securityGroupMappingField | String | Any value |
| Content Attribute Mapping | attributeMappingList | List<String> | Any value |
| Bypass Workflows | bypassWorkflow | boolean | Any value |
| Alternative Check-In Service | alternateCheckInService | String | Any value |
A.4.4 ContentAttributeMappingInfo Class Properties

The following table lists the ContentAttributeMappingInfo Class properties that can be set (use class name: oracle.odc.entity.CommitProfileEntity).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select Capture metadata fields to assign to Content Server fields</td>
<td>fieldMappingList</td>
<td>List&lt;CaptureToContentFieldMapping&gt;</td>
<td>Any value</td>
</tr>
<tr>
<td>Custom Fields</td>
<td>customMetadataList</td>
<td>List&lt;ContentFieldNameInfo&gt;</td>
<td>Any value</td>
</tr>
<tr>
<td>Exclude Attachments</td>
<td>excludeAttachments</td>
<td>boolean</td>
<td>Any value</td>
</tr>
</tbody>
</table>

A.4.5 CaptureToContentFieldMapping Class Properties

The following table lists the CaptureToContentFieldMapping Class properties that can be set (use class name: oracle.odc.entity.CommitProfileEntity).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field Value</td>
<td>fieldValue</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Type</td>
<td>docType</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Security Group</td>
<td>docSecurityGroup</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Account</td>
<td>docAccount</td>
<td>String</td>
<td>Any value</td>
</tr>
</tbody>
</table>

A.4.6 ContentFieldNameInfo Class Properties

The following table lists the ContentFieldNameInfo Class properties that can be set (use class name: oracle.odc.entity.CommitProfileEntity).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Custom Field Name</td>
<td>name</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Custom Field Caption</td>
<td>caption</td>
<td>String</td>
<td>Any value</td>
</tr>
</tbody>
</table>

A.4.7 Webcenter Imaging Commit Profile Specific Properties

The following table lists the Webcenter Imaging Commit Profile properties that can be set (use class name: oracle.odc.entity.CommitProfileEntity).
### Commit Profile Object Properties

The following table lists the PDF Searchable Document Output properties that can be set (use class name: oracle.odc.entity.CommitProfileEntity).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
</table>
| Commit Methods            | commitMethod                 | int       | 0 - Direct commit  
1 - Input agent commit  |
| Search and append to matching Imaging documents | append | boolean | Any value |
| Imaging WebService URL    | url                          | String    | Any value |
| Security Policy           | policy                       | int       | 0 - Basic security  
1 - Token security         |
| Capture Output Directory  | inputDir                     | String    | Any value |
| Input Agent Directory     | inputAgentDir                | String    | Any value |
| Search Definition         | searchName                   | String    | Any value |

### A.4.8 PDF Searchable Document Output Specific Properties

The following table lists the PDF Searchable Document Output properties that can be set (use class name: oracle.odc.entity.CommitProfileEntity).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
</table>
| Create Text File          | textFormat                   | Integer   | 0 - Do not create text file  
1 - Create text file format |
| Languages:                | language                     | String    | comma separated language IDs [0-122]  
For example, “0, 2, 3, 18” |
| Text File Code Page       | codePage                     | String    | Any value |
| Professional Dictionaries | profDictionary               | String    | Comma separated professional dictionary name. For example, “English Medical Dictionary, German Legal Dictionary” |
| Color Image Quality       | colorImageQuality            | int       | 0 - Minimum: minimum size  
1 - Good: medium size  
2 - Best: large size |
| Compatibility             | compatibility                | String    | PDF14/PDF15/PDF1  
6/PDF17/PDFA1B/PDFA2B/PDFA2U |
A.4.9 DOCS Commit Driver Properties

The following table lists the DOCS Commit Driver properties that can be set (use class name: oracle.odc.entity.CommitProfileEntity).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create Linear PDF for Efficient Web Viewing</td>
<td>linearized</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Preserve Original Image Orientation</td>
<td>keepImgOrientation</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Single Language Detection per Page</td>
<td>singleLangPerPage</td>
<td>boolean</td>
<td>Any value</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server URL</td>
<td>serverURL</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Name</td>
<td>parentFolderName</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>ID</td>
<td>parentFolderID</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Sub-Folder Fields:</td>
<td>subFolderFields</td>
<td>List&lt;String&gt;</td>
<td>Any value</td>
</tr>
<tr>
<td>Create Sub-Folders using Field Values</td>
<td>createSubFolders</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>If folder name consists of invalid characters:</td>
<td>folderInvalidCharOption</td>
<td>int</td>
<td>0 - Remove invalid characters, 1 - Cancel document commit</td>
</tr>
<tr>
<td>Use original file name for non-image files</td>
<td>useOriginalFileName</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Name document file based on metadata field values</td>
<td>nameDoc</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Fields to Include in Document File Name:</td>
<td>docFileFields</td>
<td>List&lt;String&gt;</td>
<td>Any value</td>
</tr>
<tr>
<td>Field Delimiter</td>
<td>docFileDelimiter</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>If file name consists of invalid characters:</td>
<td>docFileInvalidCharOption</td>
<td>int</td>
<td>0 - Remove invalid characters, 1 - Cancel document commit</td>
</tr>
</tbody>
</table>

A.5 Document Conversion Job Object Properties

The following table lists all the properties that can be set for a Document Conversion Job object (class name: oracle.odc.docconverter.DocConverterJob).

<table>
<thead>
<tr>
<th>Property Name (on the UI)</th>
<th>Property Name (in the Class)</th>
<th>Data Type</th>
<th>Valid Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>jobName</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Property Name (on the UI)</td>
<td>Property Name (in the Class)</td>
<td>Data Type</td>
<td>Valid Options</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>--------------------------------</td>
<td>-----------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Online:</td>
<td>online</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>JPEG Image Quality</td>
<td>jpegQuality</td>
<td>Integer</td>
<td>1&lt;jpegQuality&lt;=99</td>
</tr>
<tr>
<td>Documents to Convert:</td>
<td>docFilter</td>
<td>Boolean</td>
<td>true: Selected non-image documents false: All non-image documents</td>
</tr>
<tr>
<td>Output Image Format:</td>
<td>outputImageFormat</td>
<td>Integer</td>
<td>0 - TIFF: black and white 1 - JPEG: color</td>
</tr>
<tr>
<td>File Name Filter:</td>
<td>filterpattern</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Batch Merge Option:</td>
<td>documentMergeType</td>
<td>Integer</td>
<td>0 - Do not merge documents 1 - Merge the first document with all other documents 2 - Merge the last document with all other documents 3 - Merge all documents</td>
</tr>
<tr>
<td>Source Document Page Placement:</td>
<td>mergeToBeginning</td>
<td>boolean</td>
<td>true: Start of the target document false: End of the target document</td>
</tr>
<tr>
<td>Apply source document's metadata values</td>
<td>applySourceMetadata</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Allow target document's metadata values to be overwritten</td>
<td>overwriteDestinationMetadata</td>
<td>boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>Description:</td>
<td>description</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Batch Processor:</td>
<td>processorID</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Batch Processor Job:</td>
<td>processorJobID</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>To rename the batch, enter a prefix:</td>
<td>renamePrefix</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>To send email notification, enter email address:</td>
<td>renameEmail</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>To change batch status, select a status:</td>
<td>renameStatus</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>To change batch priority, enter a priority between 1 to 10:</td>
<td>renamePriority</td>
<td>Integer</td>
<td>[1-10]</td>
</tr>
<tr>
<td>Batch Processor:</td>
<td>errProcessorID</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Batch Processor Job:</td>
<td>errProcessorJobID</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Property Name (on the UI)</td>
<td>Property Name (in the Class)</td>
<td>Data Type</td>
<td>Valid Options</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------</td>
<td>-----------</td>
<td>---------------</td>
</tr>
<tr>
<td>To rename the batch, enter a prefix:</td>
<td>errRenamePrefix</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>To send email notification, enter email address:</td>
<td>errRenameEmail</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>To change batch status, select a status:</td>
<td>errRenameStatus</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>To change batch priority, enter a priority between 1 to 10:</td>
<td>errRenamePriority</td>
<td>Integer</td>
<td>[1-10]</td>
</tr>
<tr>
<td>DPI:</td>
<td>dpi</td>
<td>Integer</td>
<td>100 / 150 / 200 / 240 / 300 / 400 / 600</td>
</tr>
<tr>
<td>File Name Filter(s):</td>
<td>extConvFilterPattern</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>External Conversion Program:</td>
<td>extConvProgram</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Command Line Parameters:</td>
<td>extConvCommandLine</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Process Monitoring Method:</td>
<td>extConvMonitoringMethod</td>
<td>Integer</td>
<td>0 - Duration time out, 1 - Output file inactivity time out</td>
</tr>
<tr>
<td>Timeout (minutes):</td>
<td>extConvTimeout</td>
<td>Integer</td>
<td>[1-1000]</td>
</tr>
<tr>
<td>Do not convert:</td>
<td>doNotConvertDocs</td>
<td>Boolean</td>
<td>Any value</td>
</tr>
<tr>
<td>File Name Filter:</td>
<td>attachmentFilterPattern</td>
<td>String</td>
<td>Any value</td>
</tr>
<tr>
<td>Source Attachments:</td>
<td>documentAttachmentMergeType</td>
<td>Integer</td>
<td>0 - Do not include attachments, 1 - Include all attachments to the merged documents, 2 - Include attachments with the matching Document Profile Attachment Types</td>
</tr>
<tr>
<td>Restrict to Document Profiles:</td>
<td>documentTypes</td>
<td>List&lt;DOCTYPE_ENTITY&gt;</td>
<td>Any value</td>
</tr>
</tbody>
</table>