Content Server

Version: 6.3 Patch 10

Property Files Reference

Document Revision Date: Jun. 15, 2011

FatWire® SOFTWARE

FATWIRE CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. In no event shall FatWire be liable for any loss of profits, loss of business, loss of use of data, interruption of business, or for indirect, special, incidental, or consequential damages of any kind, even if FatWire has been advised of the possibility of such damages arising from this publication. FatWire may revise this publication from time to time without notice. Some states or jurisdictions do not allow disclaimer of express or implied warranties in certain transactions; therefore, this statement may not apply to you.

This product may be covered under one or more of the following U.S. patents: 4477698, 4540855, 4720853, 4742538, 4742539, 4782510, 4797911, 4894857, 5070525, RE36416, 5309505, 5511112, 5581602, 5594791, 5675637, 5708780, 5715314, 5724424, 5812776, 5828731, 5909492, 5924090, 5963635, 6012071, 6049785, 6055522, 6118763, 6195649, 6199051, 6205437, 6212634, 6279112 and 6314089. Additional patents pending.

FatWire, Content Server, Content Server Bridge Enterprise, Content Server Bridge XML, Content Server COM Interfaces, Content Server Desktop, Content Server Direct, Content Server Direct Advantage, Content Server DocLink, Content Server Engage, Content Server InSite Editor, Content Server Satellite, and Transact are trademarks or registered trademarks of FatWire, Inc. in the United States and other countries.

iPlanet, *Java*, *J2EE*, *Solaris*, *Sun*, and other Sun products referenced herein are trademarks or registered trademarks of Sun Microsystems, Inc. *AIX*, *IBM, WebSphere,* and other IBM products referenced herein are trademarks or registered trademarks of IBM Corporation. *WebLogic* is a registered trademark of BEA Systems, Inc. *Microsoft*, *Windows* and other Microsoft products referenced herein are trademarks or registered trademarks of Microsoft Corporation. *UNIX* is a registered trademark of The Open Group. Any other trademarks and product names used herein may be the trademarks of their respective owners.

This product includes software developed by the Apache Software Foundation (http://www.apache.org/) and software developed by Sun Microsystems, Inc. This product contains encryption technology from Phaos Technology Corporation.

You may not download or otherwise export or reexport this Program, its Documentation, or any underlying information or technology except in full compliance with all United States and other applicable laws and regulations, including without limitations the United States Export Administration Act, the Trading with the Enemy Act, the International Emergency Economic Powers Act and any regulations thereunder. Any transfer of technical data outside the United States by any means, including the Internet, is an export control requirement under U.S. law. In particular, but without limitation, none of the Program, its Documentation, or underlying information of technology may be downloaded or otherwise exported or reexported (i) into (or to a national or resident, wherever located, of) Cuba, Libya, North Korea, Iran, Iraq, Sudan, Syria, or any other country to which the U.S. prohibits exports of goods or technical data; or (ii) to anyone on the U.S. Treasury Department’s Specially Designated Nationals List or the Table of Denial Orders issued by the Department of Commerce. By downloading or using the Program or its Documentation, you are agreeing to the foregoing and you are representing and warranting that you are not located in, under the control of, or a national or resident of any such country or on any such list or table. In addition, if the Program or Documentation is identified as Domestic Only or Not-for-Export (for example, on the box, media, in the installation process, during the download process, or in the Documentation), then except for export to Canada for use in Canada by Canadian citizens, the Program, Documentation, and any underlying information or technology may not be exported outside the United States or to any foreign entity or “foreign person” as defined by U.S. Government regulations, including without limitation, anyone who is not a citizen, national, or lawful permanent resident of the United States. By using this Program and Documentation, you are agreeing to the foregoing and you are representing and warranting that you are not a “foreign person” or under the control of a “foreign person.”

Content Server Property Files Reference
Document Revision Date: Jun. 15, 2011
Product Version: 6.3 Patch 10

FatWire Technical Support
www.fatwire.com/Support

FatWire Headquarters
FatWire Corporation
330 Old Country Road
Suite 303
Mineola, NY 11501
www.fatwire.com

Table of Contents

About This Reference 7
Who Should Use This Reference 7
How This Reference Is Organized 7
Related Publications 7

Introduction 9
Using the Property Editor 10
Starting the Property Editor 10
Setting Properties 10
Adding Properties 11
Deleting Properties 13
Property Files by Product 14

Part 1. Content Server

Content Server Core Property Files 17
batch.ini 18
Configuration Tab 19
Debug Tab 20
Results Tab 20
Security Tab 20
User Defined Tab 20
commons-logging.properties 21
Factory Tab 22
Loggers Tab 23
Traditional Log Tab 27
User Defined Tab 28

CSPortletRequest.properties 29
- User Defined Tab 29

dir.ini 30
- Attribute Names Tab 31
- Compatibility Tab 33
- Global Data Tab 34
- Interface Implementations Tab 35
- JNDI SPI Env. Tab 37
- Naming Syntax Tab 39
- Schema Defaults Tab 41
- Search Controls Tab 43
- User Defined Tab 43

futuretense.ini 44
- App Server Tab 45
- Authentication Tab 46
- Basic Tab 49
- Blob Server Tab 52
- Cluster Tab 53
- Compatibility Tab 54
- Content Tables Tab 58
- Database Tab 59
- Debug Tab 65
- Email Tab 66
- Export/Mirror Tab 68
- JSP Tab 70
- Misc Tab 72
- Page Cache Tab 74
- ResultSet Cache Tab 76
- Satellite Server Tab 80
- Search Tab 82
- User Defined Tab 86

jsprefresh.ini (WebLogic Only) 90

logging.ini 91
- Global Data Tab 92
- Message Resources Tab 93
- User Defined Tab 93

omii.ini 94
- User Defined Tab 95

omproduct.ini 101
- User Defined Tab 102

satellite.properties 107
- Caching Tab 108
- Configuration Tab 110
- Remote Host Tab 111
- Sessions Tab 112

Compatibility Tab 114
User Defined Tab 115
ServletRequest.properties 116
Request Encoding Tab 117
Request Threshold Tab 117
URI Assembler Tab 118
User Defined Tab 120
WL6.ini 121
User Defined Tab 121
xmles.ini 122
General Properties Tab 122
Parsing Tab 122
Inbound Handlers Tab 123
User Defined Tab 123

CS-Direct Property Files 125
futuretense_xcel.ini 126
Asset Default Tab 127
Authorization Tab 130
Debug Tab 136
Directories Tab 137
Element Override Tab 139
KeyView Tab 140
Preference Tab 141
Publishing Tab 143
xcelerate Tab 148
User Management Tab 151
User Defined Tab 153

CS-Direct Advantage Property Files 155
assetframework.ini 156
User Defined Tab 156
catalog.ini 157
Catalog Tab 158
User Defined Tab 158
gator.ini 159
Gator Tab 160
User Defined Tab 163
visitor.ini 164
Visitor Data Tab 165
User Defined Tab 166

Part 2. Content Server Applications

Engage Property Files .169

Satellite Server Property Files .169

Part 3. Third-Party Libraries

HTTP Client Access .173
Apache Commons HttpClient 173
Integration with Content Server 173
Implementation 175
HTTPClient Parameters and Content Server Properties 176

Index .185

Index of Properties .189

About This Reference

This guide is a reference to Content Server's property files and the properties they contain. The properties are used to specify the operating parameters for Content Server and its applications.

Who Should Use This Reference

This reference is written for Content Server installation engineers, developers, and administrators of Content Server systems.

How This Reference Is Organized

This reference organizes property files by Content Server product.

The introductory section explains the importance of using the Property Editor to modify property files, and provides instructions on using the Property Editor.

Part 1, "Content Server" covers the property files of the Content Server core, its modules CS-Direct and CS-Direct Advantage, and its page caching application Satellite Server.

Part 2, "Content Server Applications" covers the property files of the Content Server applications Engage and remotely installed Satellite Server.

Part 3, "Third-Party Libraries" explains how Content Server integrates with third-party libraries.

At the end of this reference are two indexes: a standard index and a properties index.

Related Publications

The FatWire library includes publications written for Content Server users, administrators, and developers. The publications are provided as product manuals with your Content Server installation. They are also posted by version number at the following url:

```
http://e-docs.fatwire.com/CS
```

Other publications, such as case studies and white papers, provide information about Content Server's feature set and its business applications. To obtain these publications, contact `sales@fatwire.com`.

Introduction

Many of your configuration tasks for Content Server and its applications require you to set or modify values for properties that are specified in property files. These files have either the `.ini` or `.properties` suffix.

A set of property files, along with the "Property Editor" utility, is provided with Content Server. At least one property file is associated with each of the Content Server applications.

This introduction describes the Property Editor and summarizes the property files that are presented in the rest of this reference.

This introduction contains the following sections:

- Using the Property Editor
- Property Files by Product

Using the Property Editor

The Property Editor is used to set or modify the values of properties in property files.

FatWire recommends that you always use the Property Editor to set values in the property files, because using other editors can create problems. For example, the JumpStart Kit renames property files; launching the Property Editor ensures that the correct file will be displayed for modification. The Property Editor is also capable of encrypting password fields; modifying the fields outside the Property Editor can corrupt the property values. In addition, the Property Editor ensures that the files are correctly formatted; specifically, the correct property separator characters are used.

Furthermore, the Property Editor organizes properties by function on tabs, and provides descriptive information about the properties, as well as default or possible values.

Starting the Property Editor

To start the Property Editor

Execute the following scripts at the MS DOS prompt or in a UNIX shell:

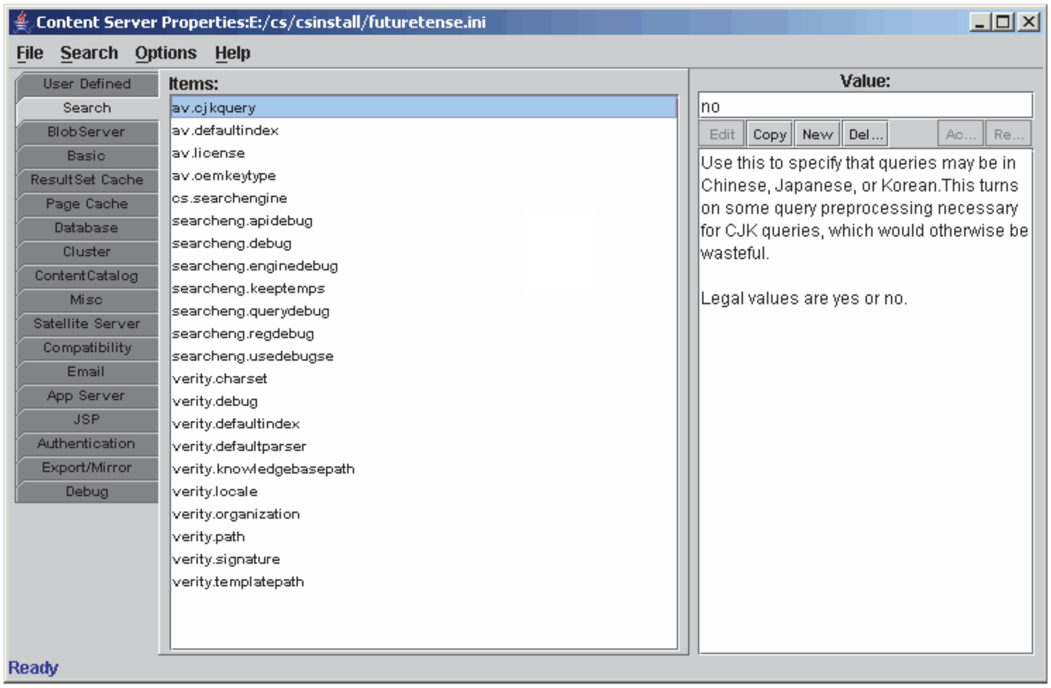
- Windows: `propeditor.bat`, which is usually located in *`installdir/`*
- Solaris: `propeditor.sh`, which is usually located in `$HOME/`*`installDir`*

Setting Properties

To set properties for the Content Server products

1. Start the Property Editor, as previously described in "Starting the Property Editor."
2. Select **File** > **Open**.
3. Browse to the property file that you need to modify and then select it.

   The Property Editor opens the file. For example, this is what the `futuretense.ini` file looks like when it is first opened in the Property Editor:



av.cjkquery
av.defaultindex
av.license
av.oemkeytype
cs.searchengine
searcheng.apidebug
searcheng.debug
searcheng.enginedebug
searcheng.keeptemps
searcheng.querydebug
searcheng.regdebug
searcheng.usedebugse
verity.charset
verity.debug
verity.defaultindex
verity.defaultparser
verity.knowledgebasepath
verity.locale
verity.organization
verity.path
verity.signature
verity.templatepath

no

Use this to specify that queries may be in Chinese, Japanese, or Korean.This turns on some query preprocessing necessary for CJK queries, which would otherwise be wasteful.

Legal values are yes or no.

The tabs on the left represent functional groups of properties.

The **Items** pane lists the properties in the selected tab (functional group).

The **Value** pane lists the current value for the selected item, a brief description of the item, and the acceptable values for it.

4. Select the tab that represents the functional group that contains the property that you want to configure.

   The Property Editor displays the properties from that tab in the **Items** pane.

5. Select the property from the **Items** pane.

   The Property Editor displays the value that is currently set for that property value and a provides a brief description of the property in the **Values** pane.

6. In the **Values** pane, enter the value for the property in the text field at the top.
7. Click **Accept**.
8. Repeat steps 4 through 7 for each property that you want to configure.
9. When you are finished, select **File** > **Save**.
10. Select **File** > **Close**.
11. Stop and restart the application server so the new values can take effect.

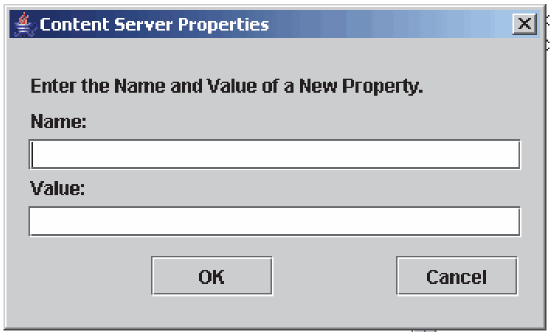
Adding Properties

Some configuration tasks require you to add new, custom properties for your system. For example, when you implement resultset caching, you can create up to three properties for any Content Server database table that you want to set caching values for. (For information about resultset caching, see the *Content Server Developer's Guide*.)

To add a property to the Content Server database

1. Start the Property Editor, previously described in "Starting the Property Editor" on page 10.
2. Select **File** > **Open**.
3. Browse to the property file in which you need to add a property.
4. (Optional) Select the **User Defined** tab. (This step is optional because any property that you create is displayed on the **User Defined** tab the next time you open this property file in the Property Editor no matter which tab you select.)
5. In the **Values** pane, just under the text entry field, click **New**.

   The Property Editor displays the **Content Server Properties** dialog box:


Content Server Properties
Enter the Name and Value of a New Property.
Name:
Value:
OK
Cancel

6. Click in the **Name** field and enter the name of the new property.
7. Click in the **Value** field and enter the value for the new property.
8. Click **OK**.

   The new property appears in the **Items** pane and the value appears in the **Values** pane.

> **Note:** If you did not select the **User Defined** tab in step 4, the new property might appear on the tab that was selected when you clicked **New** in step 5. This is temporary. The next time that you open this `ini` file in the Property Editor, the property is displayed on the **User Defined** tab.

9. Select **File** > **Save**.

Deleting Properties

It is unlikely that you will ever have to delete a property. However, if you do have to delete a property, refer to the steps below.

To delete a property

> **Caution.** Never delete a required property.

1. Start the Property Editor. (See "Starting the Property Editor," above.)
2. Select **File** > **Open**.
3. Browse to the property file that you need to modify by deleting a property and select it.
4. Select the tab that represents the functional group that holds the property that you want to delete.
5. Take note of the current value for this property just in case you need to restore it for any reason.
6. In the **Values** pane, just under the text entry field, click **Delete**.

   The Property Editor displays a confirmation message.
7. Click **Yes**.
8. The property is deleted from the property file.
9. Select **File** > **Save**.

Property Files by Product

This section lists all the Content Server property files, organized by the product with which they are associated.

Product	Modules	Property Files
Content Server	Content Server Core, which includes Satellite Server (co-resident), CS-Bridge XML, and the user manager plug-ins	batch.ini commons-logging.properties CSPortletRequest.properties dir.ini futuretense.ini jsprefresh.ini (WebLogic Only) logging.ini omii.ini omproduct.ini satellite.properties ServletRequest.properties WL6.ini xmles.ini
	CS-Direct	futuretense_xcel.ini
	CS-Direct Advantage	assetframework.ini catalog.ini gator.ini visitor.ini
Engage		ms.ini
Satellite Server (remote server version)		`resin.conf` `satellite.properties`

Part 1
Content Server

This part lists the property files that are used by the Content Server core product and its modules, CS-Direct and CS-Direct Advantage.

This part contains the following sections:

- Content Server Core Property Files
- CS-Direct Property Files
- CS-Direct Advantage Property Files

Content Server Core Property Files

Property files installed by Content Server are the following:

- batch.ini
- commons-logging.properties
- CSPortletRequest.properties
- dir.ini
- futuretense.ini
- jsprefresh.ini (WebLogic Only)
- logging.ini
- omii.ini
- omproduct.ini
- satellite.properties
- ServletRequest.properties
- WL6.ini
- xmles.ini

Content Server includes Satellite Server (co-resident), CS-Bridge XML, the user manager plug-ins, and the two modules CS-Direct and CS-Direct Advantage.

batch.ini

The batch.ini file provides configuration information for batch processes, which are used in the background by a Content Server system for various reasons (publishing, for example).

This section presents the properties in the batch.ini file, by the tab where the properties are organized to form a functional group.

Tabs in the batch.ini file are the following, in alphabetical order:

- Configuration Tab
- Debug Tab
- Results Tab
- Security Tab
- User Defined Tab

batch.ini: Configuration Tab

The **Configuration** tab holds properties that configure settings for the threads that are used by the batch processes.

Properties in `batch.ini`: **Configuration** Tab

Property	Description
`thread.count`	Specifies the number of dispatcher threads to allocate and manage in the pool. Default value: `32`
`thread.growcache`	Specifies whether additional dispatcher threads (in excess of the number specified by `thread.count`) can be allocated to the pool if they are needed. Possible values: `true` or `false` Default value: `false`
`thread.idle`	Applies only when `thread.growcache` is set to `true`. Specifies the number of seconds a dispatcher thread can remain idle before it is released by the pool. Default value: `10`
`thread.wait`	Applies only when `thread.growcache` is set to `false`. Specifies the number of seconds that a batch process waits for a free dispatcher thread before it reports an error because it cannot complete its task. Default value: `15`

batch.ini: Debug Tab

Properties in `batch.ini`: **Debug** Tab

Property	Description
`debug`	Specifies whether debugging for batch processes is enabled or disabled. If you set this value to `true`, messages about the status of batch processes are written to the `futuretense.txt` file. Default value: `false`

batch.ini: Results Tab

Properties in `batch.ini`: **Results** Tab

Property	Description
`request.folder`	Specifies the location of the file that stores information about the results of batch processes. For example, the CS-Direct publishing system uses this directory to hold the publishing log files. Default value: `./dispatcher/`

batch.ini: Security Tab

Properties in `batch.ini`: **Security** Tab

Property	Description
`security.class`	Specifies the name of the class file that is used for security checks. The default is provided for reference only: `com.openmarket.Batch.DefaultSecurity` Do **not** change the value of this property.

batch.ini: User Defined Tab

Properties in `batch.ini`: **User Defined** Tab

Property	Description
Note: This tab, by default, holds no properties.	

commons-logging.properties

Content Server employs Jakarta Commons Logging to write messages to its log files. By default, two loggers are provided with Content Server: TraditionalLog and StandardLog. A large variety of other loggers are available in the public domain. You can also write your own if the features sought are not present in any available loggers.

- TraditionalLog writes all log messages to the `futuretense.txt` file for each server. TraditionalLog supports log file rolling, custom formatting, date-stamping, and a variety of other features. Optionally, TraditionalLog supports (optionally) browser-based log retrieval by filtering log messages based on the client's IP address.
- StandardLog also supports custom formatting, date-stamping, and so on, but it sends messages to the standard output and standard error streams; in other words, it sends messages to the java console instead of `futuretense.txt`. While StandardLog does not support browser-based log retrieval, it offers another useful feature: it allows developers running a Jump Start Kit or running Content Server in an IDE debugger to see all of the log messages in the most obvious place. Using StandardLog is a good practice for all template developers, because it alerts you to errors by default, as opposed to sending an error to the `futuretense.txt` file where it requires some effort to be viewed. StandardLog is similar to Apache's default SimpleLog.

To configure Content Server to use StandardLog or TraditionalLog, open `commons-logging.properties` and set the value of the `org.apache.commons.logging.Log` property. The class names of the loggers are as follows:

TraditionalLog	`com.fatwire.cs.core.logging.TraditionalLog`
StandardLog	`com.fatwire.cs.core.logging.StandardLog`

Because Content Server uses JCL, however, any variety of loggers may be used. For example, WebLogic uses Log4J as its logger. Users simply need to configure the appropriate categories in the Log4J configuration file to use it. Consult JCL and Log4J documentation for details about advanced log configuration.

The `commons-logging.properties` file lists the logging properties that can be modified for Content Server.

Tabs in the file are the following, in alphabetical order:

- Factory Tab
- Loggers Tab
- Traditional Log Tab
- User Defined Tab

commons-logging.properties: Factory Tab

Properties in `commons-logging.properties`: **Factory** Tab

Property	Description
`org.apache.commons.logging.Log`	Specifies the class name implementing the `org.apache.commons.logging.Log` interface that the default commons-logging log factory will use as the logger. Default value: `com.fatwire.cs.core.loggingTraditionalLog` The default value names a Content Server enhanced logger that supports additional features not necessarily supported by other loggers, including browser-based retrieval of the log file, rolling logs, and timestamps. **Note:** Before modifying the value, consult the commons-logging documentation.

commons-logging.properties: Loggers Tab

Properties in `commons-logging.properties`: **Loggers** Tab

Property	**Description**
`com.fatwire.logging.cs`	This property specifies the log severity for the generic Content Server logger. Messages written to this logger relate to the core Content Server functionality, or they have not been assigned a more appropriate logger. Possible values: `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, and `FATAL` Default value: `INFO`
`com.fatwire.logging.cs.auth`	This property specifies the log severity for user authentication and user authorization messages. Messages written to this logger relate to user login, user ACLs and user roles. Possible values: `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, and `FATAL` Default value: `INFO`
`com.fatwire.logging.cs.blobserver`	This property specifies the log severity for BlobServer debug messages. Messages written to this logger relate to the addition, retrieval, and modification of data in CS page caches. Possible values: `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, and `FATAL` Default value: `INFO`
`com.fatwire.logging.cs.cache.page`	This property specifies the log severity for page cache debug messages. Messages written to this logger relate to the addition, retrieval, and modification of data in CS page caches. Possible values: `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, and `FATAL` Default value: `INFO`
`com.fatwire.logging.cs.cache.resultset`	This property specifies the log severity for result set cache debug messages. Messages written to this logger relate to the addition, retrieval, and modification of data in CS result set caches. Possible values: `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, and `FATAL` Default value: `INFO`

Properties in `commons-logging.properties`: **Loggers** Tab *(continued)*

Property	**Description**
`com.fatwire.logging.cs.db`	This property specifies the log severity for database access messages. Messages written to this logger relate to database access, queries and statement execution. It can be very helpful to use this logger to debug database queries that are not behaving as expected. Possible values: `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, and `FATAL` Default value: `INFO`
`com.fatwire.logging.cs.event`	This property specifies the log severity for CS event engine messages. Messages written to this logger relate to the invocation of events by CS. Possible values: `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, and `FATAL` Default value: `INFO`
`com.fatwire.logging.cs.export`	This property specifies the log severity for CS page export engine messages. Messages written to this logger relate to the invocation of events by CS. Possible values: `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, and `FATAL` Default value: `INFO`
`com.fatwire.logging.cs.install`	This property specifies the log severity for the CS installer's messages. Messages are only written to this log during installation. Possible values: `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, and `FATAL` Default value: `INFO`
`com.fatwire.logging.cs.jsp`	This property specifies the log severity for CS jsp element debug information. Messages written to this logger relate to the invocation of JSP elements by CS. Possible values: `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, and `FATAL` Default value: `INFO`

Properties in `commons-logging.properties`: **Loggers** Tab *(continued)*

Property	Description
`com.fatwire.logging.cs.request`	This property specifies the log severity for request processing messages. Possible values: `TRACE, DEBUG, INFO, WARN, ERROR,` and `FATAL` Default value: `INFO`
`com.fatwire.logging.cs.satellite`	This is the generic logger for Satellite Server. Messages logged with this logger pertain to system configuration.
`com.fatwire.logging.cs.satellite.cache`	This logger for Satellite Server deals with information relating to the parsing of objects that can be cached, as well as their lifecycle in the cache.
`com.fatwire.logging.cs.satellite.host`	This logger for Satellite Server is dedicated to logging information that relates to communication between Satellite Server and its host Content Server.
`com.fatwire.logging.cs.satellite.request`	This logger for Satellite Server is dedicated to logging information that relates to processing requests from the client.
`com.fatwire.logging.cs.session`	This property specifies the log severity for the Content Server session logger. Messages written to this logger relate to session creation, modification, and access. Possible values: `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR, FATAL` Default value: `INFO`
`com.fatwire.logging.cs.sync`	This property specifies the log severity for CS cluster synchronization messages. Messages written to this logger relate to cluster sync behavior of CS. Possible values: `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR,` and `FATAL` Default value: `INFO`

Properties in `commons-logging.properties`: **Loggers** Tab *(continued)*

Property	Description
`com.fatwire.logging.cs.time`	This property specifies the log severity for CS time and performance debug information. Messages written to this logger include information that can be used to tune the performance of pages and of CS in general. Possible values: `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, and `FATAL` Default value: `INFO`
`com.fatwire.logging.cs.visitor.object`	This is the logger for tracking the visitor object lifecycle.
`com.fatwire.logging.cs.visitor.ruleset`	This is the logger for rule set compilation.
`com.fatwire.logging.cs.xcelerate.advantage.recommendation`	This is the logger for recommendation processing.
`com.fatwire.logging.cs.xcelerate.approval`	This is the logger for asset approval processing.
`com.fatwire.logging.cs.xml`	This property specifies the log severity for CS XML element debug information. Messages written to this logger relate to the parsing and evaluation of XML elements. It should be noted that enabling this debug flag will cause CS to use a validating XML parser instead of the default non-validating parser. This can have effects on rendered pages. Consult the documentation for details. Possible values: `TRACE`, `DEBUG`, `INFO`, `WARN`, `ERROR`, and `FATAL` Default value: `INFO`

commons-logging.properties: Traditional Log Tab

The table below lists properties that are specific to TraditionalLog, the default logger specified in the property `org.apache.commons.logging.Log`. TraditionalLog supports numerous features that other loggers might not support. In your Spark installation, you may choose to use a different logger with your own set of properties.

Properties in `commons-logging.properties`: **Traditional Log** Tab

Property	Description
`logging.file`	Specifies the full path to the log file. It is required and should be set to a different value for each VM to avoid conflicts. **Note:** This value must be specified or logging will fail.
`logging.format`	Specifies the `java.text.SimpleDateFormat` that will be used in timestamps that are written to the log file. Consult the API documentation for `java.text.SimpleDateFormat` for syntax information. If not specified, the default SimpleDateFormat will be used.
`logging.maxlogsize`	Specifies the maximum size of the log file in bytes. Once the log file grows to the size specified, it will either be rolled or deleted. Set it to `-1` to allow the log to grow indefinitely. Default value: `10MB`
`logging.per-client-log`	This property specifies whether or not a log file will be kept for each distinct IP address requesting data from Content Server. This property allows the use of CatalogManager to retrieve the log file from a web browser. Legal values: `true` or `false`
`logging.roll`	This property specifies how many logs will be kept when rolling the log file. Previous versions will be renamed and once all available versions are used, the oldest log will be deleted. Set to 0 to disable rolling of the log file. Set it to a positive integer to specify the number of logs to keep. Default value: `0`

Properties in `commons-logging.properties`: **Traditional Log** Tab *(continued)*

Property	Description
`logging.timestamp`	This property specifies whether or not the timestamp will be written to the log file for each logging entry. Possible values: `true` or `false` Default value: `true`

commons-logging.properties: User Defined Tab

Properties in `commons-logging.properties`: **User Defined** Tab

Property	Description
Note: This tab, by default, holds no properties.	

CSPortletRequest.properties

The `CSPortletRequest.properties` file holds portal configuration properties that are set by the Content Server installation program.

This section presents the properties in the `CSPortletRequest.properties` file by the tabs where the properties are organized to form a functional group.

The following tab is displayed for the `CSPortletRequest.properties` file:

- User Defined Tab

> **Caution.** Do **not** change the values of any properties on this tab.

CSPortletRequest.properties: User Defined Tab

Properties in `CSPortletRequest.properties`: **User Defined** Tab

Property	Value
`cs.contenttype.UTF-8`	`text/html; charset=UTF-8`
`cs.charset`	`_charset_`
`cs.contenttype`	`text/html; charset=UTF-8`
`cs.disksize`	`102400`

dir.ini

The `dir.ini` file is the main configuration file for Content Server's directory services API. The API enables your Content Server system to connect to directory servers containing authentication information, user information, and so on. Content Server provides three directory services options (implemented through the directory services API) for managing user information (one of the options is configured for your Content Server system):

- The Content Server directory services plug-in, which uses the native Content Server user management tables (`SystemUsers` and `SystemUserAttrs`).
- The LDAP plug-in. When you use this option, user names and attributes are stored in your directory server rather than in the Content Server database.
- The NT 4.0 plug-in, which uses the Content Server user management tables, but authenticates users through NT authentication.

The `dir.ini` file contains the properties that configure the directory services options described above.

This section presents the properties in the `dir.ini` file, by the tab where the properties are organized to form a functional group.

Tabs in the `dir.ini` file are the following, in alphabetical order:

- Attribute Names Tab
- Compatibility Tab
- Global Data Tab
- Interface Implementations Tab
- JNDI SPI Env. Tab
- Naming Syntax Tab
- Schema Defaults Tab
- Search Controls Tab
- User Defined Tab

Note: Although the `dir.ini` file is the main configuration file for the directory services API, there are additional user manager/directory services properties in another property file, `futuretense.ini`. See also the section "Authentication Tab" on page 46, which describes the properties on the `authentication` tab of `futuretense.ini`.

dir.ini: Attribute Names Tab

The **Attribute Names** tab holds attribute-mapping properties. You use these properties to specify how a user attribute used by Content Server is identified in the directory server.

Properties in `dir.ini`: **Attribute Names** Tab

Property	Description
`cn`	Specifies the name of the attribute in the directory server that serves as the group name attribute. Possible values: • Content Server and NT: `cn` • LDAP, iPlanet: `cn` • LDAP, Active Directory: `cn`
`loginattribute`	Specifies the name of the attribute in the directory server that serves as the user login attribute. Default value: `uid`
`password`	Specifies the name of the attribute in the directory server that serves as the password attribute. Possible values: • Content Server and NT: `password` • LDAP, iPlanet: `userPassword` • LDAP, Active Directory: `password`
`uniquemember`	Specifies the name of the attribute in the directory server that serves as the group assignment attribute. Possible values: • Content Server and NT: `uniquemember` • LDAP, iPlanet: `uniquemember` • LDAP, Active Directory: `member`

Properties in `dir.ini`: **Attribute Names** Tab *(continued)*

Property	Description
`username`	Specifies the name of the attribute in the directory server that serves as the user name attribute. Possible values: • Content Server and NT: `username` • LDAP, iPlanet: `uid` • LDAP, Active Directory: `sAMAccount`
`memberof`	Specifies the name of the user attribute that contains information about the user's groups. If a user's membership in a group is specified by the `uniquemember` attribute on the group, leave this property blank. If the group membership of a user is specified by an attribute of the user, specify the name of that attribute here. Possible values: • LDAP, WebLogic embedded LDAP: `wlsmemberof` • LDAP, SunONE Identity Server: `memberof`

dir.ini: Compatibility Tab

The **Compatibility** tab holds properties that determine how any strings that are extracted from the directory server and stored in the Content Server database are treated.

Properties in `dir.ini`: **Compatibility** Tab

Property	Description
`cleandns`	Specifies how the strings for distinguished names are stored in the Content Server database. If set to true, the Directory Services API extracts distinguished names from the directory server, and then removes extra spaces from them and then changes all the upper-case letters to lower-case letters before storing the strings in the Content Server database. Possible values: `true` | `false` Default value: `false` **Note:** Do not set this value to `true` if you are upgrading from an earlier version of Content Server. If you do, you must manually change any existing dns strings that are stored in the Content Server tables. Also, if you set it to `true`, you must also verify that the `syntax.ignorecase` property is also set to `true`.

dir.ini: Global Data Tab

The **Global Data** tab holds properties that determine global values for all users.

Properties in `dir.ini`: **Global Data** Tab

Property	Description
`baseDN`	Specifies the distinguished name for the root to use by default for searches and for prepending to the names for attribute values that require a DN type. Default value: blank Do **not** change the value of this property. Because the authentication module sets the currentUser session variable to a fully qualified name, CS-Direct assumes that all names returned from search are fully qualified.
`groupparent`	Specifies the entry to use as the parent of all Content Server entries of type `Group`. Possible values: • Content Server and NT: `ou=groups` • LDAP, iPlanet: `ou=groups,dc=`*`companyname`*`,dc=com` • LDAP, Active Directory: `cn=groups,dc=`*`companyname`*`,dc=com`
`peopleparent`	Specifies the entry to use as the parent of all Content Server entries of type `User`. Possible values: • Content Server and NT: `ou=people` • LDAP, iPlanet: `cn=people,dc=`*`companyname`*`,dc=com` • LDAP, Active Directory: `cn=users,dc=`*`companyname`*`,dc=com`

dir.ini: Interface Implementations Tab

The **Interface Implementations** tab holds two properties that determine which user manager module your Content Server system is using. The values of the rest of the properties on the tab should never be modified.

Properties in `dir.ini`: **Interface Implementations** Tab

Property	Description
`className.Attribute`	Specifies the name of the concrete class to implement the interface Attribute. Do **not** change the value of this property.
`className.Attributes`	Specifies the name of the concrete class to implement the interface Attributes. Do **not** change the value of this property.
`className.IDir`	With the `className.IName` property, specifies which user manager module your system is using. Possible values: • Content Server: `com.openmarket.directory.cs.CSDir` • LDAP: `com.openmarket.directory.jndi.JNDIDir` Do **not** change the value of this property after the installation.
`className.IFactory`	Specifies the name of the concrete class to implement the interface Ifactory. Do **not** change the value of this property.
`className.IName`	With the `className.IDir` property, specifies which user manager module your system is using. Possible values: • Content Server: `com.openmarket.directory.cs.CSName` • LDAP: `com.openmarket.directory.jndi.NameWrapper` Do **not** change the value of this property after the installation.

Properties in `dir.ini`: **Interface Implementations** Tab *(continued)*

Property	Description
`className.IUserDir`	Specifies the name of the concrete class to implement interface `IUserDir`. Possible values: • Content Server: `com.openmarket.directory.cs.CSDir` • LDAP: `com.openmarket.directory.jndi.LDAPUserDir`
`className.JNDIName`	Specifies the name of the concrete class to implement the interface JNDIName. Do **not** change the value of this property.

dir.ini: JNDI SPI Env. Tab

The properties on the **JNDI SPI Env** tab are used only if your Content Server system is configured to use the LDAP user manager module.

Properties in `dir.ini`: **JNDI SPI Env** Tab

Property	Description
`java.naming.factory.initial`	Specifies the initial factory class to use. The value of this property should be the fully qualified class name of the factory class that will create an initial context. Do **not** change the value of this property.
`java.naming.security.authentication`	Specifies the security level to use. Its value is one of the following strings: `none`, `simple`, `strong`. If this property is unspecified, the security level is determined by the service provider. Default value: `simple`
`jndi.baseURL`	Specifies the server name and port number of the directory server. The value uses the following format: `ldap://`*`hostname`*`:`*`port`*
`jndi.connectAsUser`	Specifies whether Content Server needs a designated user account to query the directory server for user attribute information. If set to `true`, `jndi.connectAsUser` specifies a Content Server login to the LDAP server. This means that Content Server queries the directory server for information as the user who is logged in to the CS system and is making the inquiry. For example, when an administrator examines user information in the Content Server interface, Content Server makes the inquiry as that user (admin, for example.) If set to `false`, `jndi.connectAsUser` specifies a direct login to the LDAP server. This means there must be a valid username/password combination specified for the `jndi.login` and `jndi.password` properties; Content Server uses that user account to make inquiries.
`jndi.custom`	System property. Do **not** enter a value for this property. Specifies other spi-specific variables to pass into the `javax.naming.context`. Syntax follows x-www-form-urlencoded format.

Properties in `dir.ini`: **JNDI SPI Env** Tab *(continued)*

Property	Description
`jndi.login`	Applies only when `jndi.connectAsUser` is set to `false`. Specifies the fully qualified, fully distinguished name of the user account that Content Server uses to query the directory server.
`jndi.password`	Applies only when `jndi.connectAsUser` is set to `false`. Specifies the password of the user account that Content Server uses to query the directory server. This value is encrypted.
`jndi.poolConnections`	Applies only when `jndi.connectAsUser` is set to `false`. If this property is set to `true`, the system will create a pool of `DirContent` connections. If set to `false`, `jndi.poolsize` is ignored.
`jndi.poolsize`	Indicates the size of the pool. Default value: `20` This property is ignored if `jndi.poolConnections` is set to `false`.
`syntax.custom`	Specifies classJNDIName-specific variables to pass into the classIName constructor. Syntax follows x-www-form-urlencoded format.

dir.ini: Naming Syntax Tab

The **Naming Syntax** tab holds properties that determine how strings for user attributes and their values are interpreted.

Properties in `dir.ini`: **Naming Syntax** Tab

Property	Description
`syntax.beginquote`	Specifies the string that delimits the beginning of a quoted string. Default value: `'`
`syntax.beginquote2`	Specifies an alternative to the value specified for the `syntax.beginquote` property.

`syntax.direction`	Specifies the direction that the components in a designated name are read in.
	Possible values:
	`left_to_right`
	`right_to_left`
	`flat`
	Default value: `left_to_right`

`syntax.endquote`	Specifies the string that delimits the end of a quoted string. Default value: `'`
`syntax.endquote2`	Specifies an alternative to the value specified for `syntax.endquote`.
`syntax.escape`	Specifies the escape string for overriding separators, escapes, and quotes. If you are using special characters, such as ‘`,`’, ‘`+`’, ‘`-`’, ‘`;`’, etc., in DN, you cannot use \ as an escape string. Default value: \

Properties in `dir.ini`: **Naming Syntax** Tab *(continued)*

Property	Description
`syntax.ignorecase`	Specifies whether strings are case-sensitive or not. Set to `false` if the uppercase and the lowercase version of a letter character should be considered as different characters. (That is, "admin" and "Admin" should be interpreted as different words.) Set to `true` if you want the uppercase and the lowercase version of a letter character to be considered as the same character. (That is "admin" and "Admin" should be interpreted as the same string.) Default value: `true` **Note**: if you need to set the `cleandns` property on the **Compatibility** tab to `true`, you must also set this property's value to `true`.
`syntax.separator`	Specifies the separator character used between atomic name components. This property is required unless `syntax.direction` is set to a value of `flat`. Default value: `,`
`syntax.separatorava`	Specifies the separator character used to separate multiple attribute/value pairs. Typically the comma character (,) is used. Default value: `,`
`syntax.separatortypeval`	Specifies the separator character used to separate an attribute from its value. For example, the equals symbol (=) is used. Default value: **=**
`syntax.trimblanks`	Specifies whether spaces and whitespace characters are significant or should be ignored (trimmed) when evaluating a string. Set to `true` if spaces should be ignored. Set to `false` if spaces should be considered when evaluating a string. Default value: `false`

dir.ini: Schema Defaults Tab

The **Schema Defaults** tab holds properties that identify the following entities to Content Server:

- The directory server attributes for which users must have values in order to be valid users
- Attribute values that are assigned to users by default (if any).

Properties in `dir.ini`: **Schema Defaults** Tab

Property	Description
`defaultGroupAttrs`	Specifies the attribute name/value pairs that are set for every descendent of the entry specified by the `groupparent` property. CS-Direct uses this information to create the default groups that it needs during the installation, which means that this property must be set **before** you install CS-Direct. Values must be entered in the x-www-form-urlencoded format.
`defaultPeopleAttrs`	Specifies the attribute name/value pairs that are set for every descendent of the entry specified by the `peopleparent` property. CS-Direct uses this information to create the default users that it needs during the installation, which means that this property must be set **before** you install CS-Direct. Values must be entered in the x-www-form-urlencoded format.
`defaultReaderACLs`	The ACL list to be assigned to DefaultReader by Login Module. Default value: `Browser`
`objectclassGroup`	Specifies the name of the base object that signifies a Content Server group. The `DIR.GROUPMEMBERSHIPS` tag uses the value set for this property to differentiate group entries from user or other entries. Possible values: • Content Server: `groupofuniquenames` • LDAP, iPlanet: `groupofuniquenames` • LDAP, Active Directory: `group`

Properties in `dir.ini`: **Schema Defaults** Tab *(continued)*

Property	**Description**
`objectclassPerson`	Specifies the name of the base object that signifies a Content Server user (person). The `DIR.LISTUSERS` tag uses the value set for this property to differentiate user entries from group or other entries Value for Content Server or LDAP: `person`
`requiredGroupAttrs`	Specifies the attributes that every descendent of the entry specified by the `groupParent` property must have values for. Values must be entered in the x-www-form-urlencoded format.
`requiredPeopleAttrs`	Specifies the attributes that every descendent of the entry specified by the `peopleParent` property must have values for. Values must be entered in the x-www-form-urlencoded format.

dir.ini: Search Controls Tab

The **Search Controls** tab holds properties that constrain the queries that the user manager plug-in makes to the directory server.

Properties in `dir.ini`: **Search Controls** Tab

Property	Description
`search.returnLimit`	Specifies the maximum number of entries to return. To obtain all the entries that satisfy the search criteria, set the value to `0`.
`search.scope`	Specifies to what depth in the hierarchy a search reaches: search just the specified or current node, or search the nodes under that node. Default value: `2` (which means search all nodes under the stated node)
`search.timeoutVal`	Specifies the number of seconds to wait for results before returning an error. A value of `0` means to wait indefinitely (that is, wait until the network timeout limit ends the wait).

dir.ini: User Defined Tab

Properties in `dir.ini:` **User Defined** Tab

Property	Description
Note: This tab, by default, has no properties.	

futuretense.ini

The futuretense.ini file is the main property file for Content Server. This section presents the properties in the futuretense.ini file, by the tab where the properties are organized to form a functional group.

Tabs in the futuretense.ini file are the following, in alphabetical order:

- App Server Tab
- Authentication Tab
- Basic Tab
- Blob Server Tab
- Cluster Tab
- Compatibility Tab
- Content Tables Tab
- Database Tab
- Debug Tab
- Email Tab
- Export/Mirror Tab
- JSP Tab
- Misc Tab
- Page Cache Tab
- ResultSet Cache Tab
- Satellite Server Tab
- Search Tab
- User Defined Tab

futuretense.ini: App Server Tab

The **App Server** tab holds the `futuretense.ini` properties that supply information to Content Server about the application server.

Properties in `futuretense.ini`: **App Server** Tab

Property	Description
`cs.eventhost`	The host string for running the event engine on application servers. Enter the host and port number, as in the following example: `http://localhost:80`
`ft.cgipath`	The web server CGI directory where Content Server objects are installed. Used in the constructions of URLs and form actions. Be sure the value ends with a backslash (/). Possible values: `/NASApp/CS/` - when using the Sun ONE application server (iAS); the path is defined to be the path used when installing Sun ONE. `/servlet/` - in http servlet application environments (such as WebLogic).

futuretense.ini: Authentication Tab

The **Authentication** tab holds user authentication properties that are configured during installation based on the user manager plug-in in use on your Content Server system. Some of these properties apply no matter which user management module you are using while others apply only if you are using NT authentication.

See also, the main configuration file for the user manager plug-ins is the `dir.ini` file. See "dir.ini" on page 30.

Properties in `futuretense.ini`: **Authentication** Tab

Property	Description
`cs.manageACL`	Specifies the class that replaces the default Content Server ACL name-to-privilege mask function. Do **not** change the value of this property.
`cs.manageproperty`	Specifies the name of the appropriate property file that configures the NT authentication plug-in or the LDAP plug-in, depending on whether you are using either of these user manager modules. • For NT, set the value to: `futuretense.ini` • For LDAP, set the value to: `dir.ini`
`cs.manageUser`	Specifies which user manager plug-in to use with this CS system. • If you are using the default Content Server plug-in, the value is blank. • If you are using LDAP, the value is: `com.openmarket.directory.jndi.auth.JNDILogin` • If you are using NT authentication, the value is: `com.FutureTense.NTUserGroups.ValidateLogin.NTUserGroupsLogin` This property was set when your CS system was installed. Do **not** change it after installation.
`cs.manageUserAccess`	Specifies the class that replaces the default Content Server user-to-privilege-by-resource lookup functionality. Do **not** change the value of this property.

Properties in `futuretense.ini`: **Authentication** Tab *(continued)*

Property	Description
`cs.manageUserSystem`	Applies only when your CS system is using NT authentication (that is, `cs.manageUser` points to the NT plug-in). Specifies a comma-separated list of NT domain names that Content Server uses to authenticate users. Authentication is done in the order specified by the list of domains. A user is declared a valid user if his username/password combination is found in any of those domains. Specify the local system with a period (.) character. If the `cs.manageUser` points to the NT plug-in, but there are no domain names set for this property, Content Server attempts authentication on the local NT domain only. This value was set during installation.
`ntlogin.DefaultACL`	NT user manager plug-in only. A comma-separated list of any ACLs that should be assigned to all users by default. By default, this value is blank.
`ntlogin.DefaultReaderACL`	NT user manager plug-in only. The ACL list to be assigned to the account that is used as the default reader account. By default, this value is blank.
`ntlogin.DefaultReaderID`	NT user manager plug-in only. The user name to be used as the default reader account. By default, it is set to `DefaultReader`. This value can be blank.
`ntlogin.DefaultReaderPW`	NT user manager plug-in only. The password for the user name that the NT authentication module uses as the default reader account. Required if there is a value for `ntlogin.DefaultReaderID`. By default, it is set to `SomeReader`. The value is encrypted.

Properties in `futuretense.ini`: **Authentication** Tab *(continued)*

Property	Description
`ntlogin.LogFile`	NT user manager plug-in only. The complete path to the file where debug information from the NT authentication module should be written. (Used only when `ntlogin.Logging` is set to `true`.)
`ntlogin.Logging`	NT user manager plug-in only. Enables or disables debugging for the NT authentication module. Possible values: `true` | `false`
`singlesignon`	If set to `true`, enables single sign-on for an authentication plugin that supports it.

futuretense.ini: Basic Tab

The **Basic** tab holds the `futuretense.ini` properties that control such things as security settings, session timeouts, and Global Unique Identifiers that the Content Server servlets use.

Properties in `futuretense.ini`: **Basic** Tab

Property	Description
`bs.security`	Specifies whether the BlobServer servlet checks security before allowing database access and image retrieval. If security is on, images cannot be cached in memory. If you enable BlobServer security, the BlobServer servlet serves the data only if the `csblobid` parameter exists in the URL and its value matches a session variable of the same name, as in the following example: `<IMG SRC="BlobServer? blobtable=MovieImages&#38; blobcol=urlimage&#38;blobkey=id&#38; csblobid=SessionVariables.blobid&#38; blobwhere=25"/>` Possible values: `true` | `false`.
`cc.security`	Specifies whether Content Server checks security before allowing database access. This property should always be set to `true` except in special cases. Possible values: `true` | `false`.
`cs.barEqualsSlash`	Specifies whether an Internet Explorer browser should interpret the bar (|) character as a forward slash (/) when it is included in a page name. Possible values: `true` | `false` For example, when set to `true`, Internet Explorer interprets `pagename=folder|subfolder|page` as the same page as `pagename=folder/subfolder/page` Default value: `false`
`cs.session`	Specifies whether Content Server starts and maintains a browser session for each user. Possible values: `true` | `false`. Cannot be set to `false` when the `cc.security` property is set to `true`.

Properties in `futuretense.ini`: **Basic** Tab *(continued)*

Property	Description
`cs.timeout`	Specifies the number of seconds a connection can remain idle before the application server logs out this connection, which ends a browser session. Idle time is the time between Content Server http requests. Default value: `300` (5 minutes) **Note**: When the approval system approves assets, it is not a background process. Therefore, if you use the **Approve Multiple Assets** feature, be sure to set this property to a value that is greater than the amount of time it takes to approve a batch of asset so that the browser session does not time out. You will have to experiment with this setting, but you can start by setting it to 1800 seconds (30 minutes).
`cs.uniqueidpoolsize`	Specifies the number of unique and cluster-safe ID numbers that are cached at one time. (Content Server generates unique IDs for every row in any database table.) Default value: `100`
`cs.wrapper`	Specifies whether the Content Server HTML wrapper pages should (can) be used. Default value: `true` Set this value to `false` on a CS system in which the application server does not have HTTP access to the web server, or, if you have removed the directory that holds the wrapper pages for security reasons. See also the *Content Server Administrator's Guide*.
`ft.version`	Specifies the version number of the Content Server application. Do **not** modify this value.
`secure.CatalogManager`	Specifies whether the DefaultReader user can access the CatalogManager servlet. Possible values: `true` or `false` During installation, this property is set to `false`. Be sure that this value is changed to `true` after the installation. For more information, see the *Content Server Administrator's Guide*.

Properties in `futuretense.ini`: **Basic** Tab *(continued)*

Property	Description
`secure.DebugServer`	Controls whether `DefaultReader` can connect to the specified servlet when security is on. If set to `true`, `DefaultReader` cannot connect. Legal values: `true` or `false`
`secure.TreeManager`	Specifies whether the DefaultReader user can access the TreeManager servlet. Default value: `true` For more information, see the *Content Server Administrator's Guide*.

futuretense.ini: Blob Server Tab

The Blobs/Eval tab holds properties that configure the BlobServer and EvalServer servlets:

- BlobServer serves blobs. It gathers a blob from a table and performs all the necessary security checks. When BlobServer serves a blob, it caches it in both the Content Server and Satellite Server.

  BlobServer properties have the prefix “`bs`”.

- EvalServer is a little-used servlet that still remains in the Content Server for backwards compatibility. In general, you should leave all EvalServer properties set to their defaults.

  EvalServer properties have the prefix “`es`”.

Properties in `futuretense.ini`: **Blobs/Eval** Tab

Property	Description
`bs.bCacheSize`	Specifies the default number of blobs that can be cached (to memory). Default value: `100`
`bs.bCacheTimeout`	Specifies the number of seconds that a blob is kept in the memory cache since it was last requested before it is flushed. Note that the memory cache is cleared whenever the BlobServer servlet is restarted. Default value: `300` (5 minutes)

futuretense.ini: Cluster Tab

The **Cluster** tab holds the properties that Content Server uses to communicate with all the servers in a cluster when a Content Server system is installed in a cluster.

Properties in `futuretense.ini`: **Cluster** Tab

Property	Description
`cc.cacheNoSync`	Specifies whether the system allows the database query transaction data cached by Content Server to persist even if a cluster member updates the table. Default value: `false` You can set table-specific values for individual tables by adding a property in the following format: `cc.<`*`sometable`*`>CSync=<true or false>`
`ft.sync`	An identifier or value that defines the synchronizer key for clustered servers; for example, the DSN that cluster members use for their shared database. Each cluster member must have the same `ft.sync` value. It is valid to leave this field blank; the field should be left blank during the installation. **Note:** You should not turn off `ft.sync` on systems with multiple Java virtual machines. Instead, you can set `cc.cacheNoSync=false`, or you can set a specific table as follows: `cc.<sometable>CSync=false`. For example, `cc.ElementCatalogCSync=false` turns off ft.sync for the `ElementCatalog` table.
`ft.usedisksync`	Specifies a shared file system folder to synchronize data across a cluster. Set this property to a valid folder when synchronization is turned on with the `ft.sync` property. For example, set it to a directory where you have read/write access.

futuretense.ini: Compatibility Tab

The **Compatibility** tab holds properties that set values necessary for backward compatibility with earlier versions of Content Server.

Properties in `futuretense.ini`: **Compatibility** Tab

Property	Descriptions
`cs.cookievariables`	Specifies if cookie variables should be created for servlet request information. For Content Server V5.5 compatibility specify `true`, otherwise specify `false`. Generating cookie variables can confuse page criteria for page requests and only works for outer pages.
`cs.dataindatabase`	Specifies whether large data should be saved in the database or in the file system as an upload column.
`cs.httpvariables`	Specifies whether the Content Server variables that contain HTTP header information are created on each page request, as was necessary in pre-4.0 versions of the product. Starting with 4.0, Content Server provides built-in variables that perform the same function. Default value: `false` For best performance, leave this value set to `false`. If developers need to retrieve an http variable for a site page, they can retrieve the value of the ones they need by using the built-in variables.

Properties in `futuretense.ini`: **Compatibility** Tab *(continued)*

Property	Descriptions
`cs.pgcachefolder`	For backwards compatibility. In previous versions, when cache information is specified in the `cacheinfo` column for a `SiteCatalog` page entry, it pointed to this property. This property specified the default directory location where Content Server pages would be cached. Starting in version 5.0, Content Server pages are cached in the database, not to a directory. The value in the `cacheinfo` column now starts with a value of `true` or `false`, which determines whether the page is cached or not. For backwards compatibility, if the value in the `cacheinfo` column of a SiteCatalog page entry follows the old syntax, the CacheManager caches Content Server pages using the old methodology. Therefore, if the `SiteCatalog` page entries on your system have not yet been updated to use the new syntax (which invokes the new caching behavior), this property must be set to a valid directory.
`cs.satellitehosts`	Specifies the host names of the servers that are hosting Satellite servlets that the CacheManager on this server (the one that hosts Content Server) needs to communicate with. Enter a comma-separated list of host names. The value for each host must include the path to the Content Server servlets. Use the following format: `http://hostname:port/servlet/` You can use https or special ports, if necessary. If required by your configuration, be sure to specify a fully-qualified domain name. The Satellite servlet that resides on this server is listed by default.

Properties in `futuretense.ini`: **Compatibility** Tab *(continued)*

Property	Descriptions
`cs.satellitepassword`	Specifies the passwords for the user accounts specified by the `cs.satelliteusrs` property. Note that the password for the Satellite servlet on this server is listed by default. The value of this property is encrypted as a single string. Therefore, when you edit the value of this property, you must enter all the passwords for all the Satellite servlet hosts, including the comma delimiter. Enter a comma-separated list of passwords in the order that matches the order in which you enter the corresponding users for the `cs.satelliteusers` property. Be sure that the order of this list also matches the order of the list of host names provided for the `cs.satellitehosts` property.
`cs.satelliteusers`	Specifies the user names for the Satellite Server hosts. Note that the user name for the Satellite servlet on this server is listed by default. Enter a comma-separated list of user names in the order that matches the list of passwords that you specified for the `cs.satellitepassword` property.
`cs.selfmodify`	Specifies a comma-separated list of attributes that a user is allowed to modify. Currently, only `password` is supported. For complete backward compatibility, set the property to blank to ensure that the user has write privileges to the `SystemUsers` table.
`cs.xmlHeaderAutoStream`	Tells Content Server whether it should stream the XML header as defined in the property `cs.xmlHeader` as the first content of the response, automatically, or not. If this property is set to `true`, then CS will automatically insert the header into the response. If it is set to `false`, then CS will not do anything. If the response is not a SOAP response, this property is ignored entirely and the header is never inserted. Default value: `false` For backward-compatibility with CS 6.1 and earlier, set this property to `true`.

Properties in `futuretense.ini`: **Compatibility** Tab *(continued)*

Property	Descriptions
`ft.approot`	This property is no longer used. In some cases, it may be required for backward compatibility.
`ft.catalogmanager`	Defines the Global Unique Identifier (GUID) for the CatalogManager service. It is provided for reference only. Value: {`40DD4E30-8DE2-11D1-8599-0080C7D07E91`} Do **not** modify this value.
`ft.contentserver`	Defines the GUID for the Content Server servlet. It is provided for reference only. Value: {`29434AD0-8DE2-11D1-8599-0080C7D07E91`} Do **not** modify this value.
`ft.treemanager`	Defines the GUID for the Treemanager servlet. Do **not** modify this value.
`security.checkpagelets`	Specifies whether Content Server checks security before allowing a user to view a pagelet that is nested in an enclosing page. Note that the `cc.security` property must also be set to `true` for security to be implemented. Default value: `true` If set to `false`, the following occurs: • On systems with co-resident Satellite Server, security is not checked on any pages or pagelets, even with `cc.security` set to `true`. • On systems with stand-alone Satellite Server or with Content Server only, security is checked on the first or outermost page but security is not checked on nested pagelets.

futuretense.ini: Content Tables Tab

The **Content Tables** tab holds a property that specifies the default primary key column for all of the content tables (as opposed to object tables) in your Content Server database.

If you or your developers create any content tables to support your online sites, you can specify that a column other than the one specified by the default is the primary key column for those content tables by creating table-specific properties. Use the following format:

```
cc.<name of table>Key=<name of column>
```

For example, when CS-Direct installs the `Category` table (which is used by basic asset types), it creates a property named `cc.CategoryKey`. The `cc.CategoryKey` property and any new property that you create appears on the **User Defined** tab rather than the **Content Tables** tab.

> Note: Be sure that you do not change the key value specified for any of the CS-Direct content tables.

The following table describes the content table property:

Properties in `futuretense.ini`: **Content Tables** Tab

Property	Descriptions
`cc.contentkey`	Specifies the name of the column that serves as the primary key for content tables in the Content Server database. This is a default setting that applies to any content table that does not have a table-specific property that sets a different primary key for it. Value set during installation of the CS content applications: `id` **Caution:** Do **not** change the value of this property. If you change it, the CS content applications will not function. To specify a different primary key for an individual content table, create a table-specific key property as described in the paragraphs preceding this table.

futuretense.ini: Database Tab

The **Database** tab holds the both the general database configuration properties, properties, such as database name and user access properties, as well as vendor-specific properties, such as how the database interprets date/time values.

The database properties must be set to the same values on each of the systems in your Content Server system—development, management, and delivery—so that you can move assets and other work from one system to another.

Note: Database properties are set during the Content Server installation.

Do **not** change the values of these properties after Content Server is installed.

If you are unsure about how or why values were determined, check with your database administrator or whoever was responsible for installing Content Server.

Properties in `futuretense.ini`: **Database** Tab

Property	Description
`cc.bigint`	Specifies the SQL string for defining a 64-bit integer field. Possible values: • Oracle: `NUMBER`(20) • SQL Server: `BIGINT` • DB2: `BIGINT` Do **not** change the value of this property.
`cc.bigtext`	Specifies the SQL string for defining a large text field. Possible values: • Oracle: `CLOB` • SQL Server: `TEXT` • DB2: `LONG VARCHAR` Do **not** change the value of this property. **Note:** If you are using an Oracle database and require text attributes greater than 2000 characters, you must set `cc.bigtext` to `CLOB`. To support `CLOB`, use Oracle database 9.2.0.6 (or a higher supported version). Also use Oracle 10g drivers. `CLOB` is not supported for lower database versions and for Oracle drivers 9x (thin, type 4).

Properties in `futuretense.ini`: **Database** Tab *(continued)*

Property	Description
`cc.blob`	Specifies the SQL string for defining a BLOB (binary large object) field. Possible values: • Oracle: `BLOB` • SQL Server: `IMAGE` • DB2: `LONG VARCHAR FOR BIT DATA` Do **not** change the value of this property.
`cc.char`	Specifies the SQL string for defining a CHAR data type. Possible values: • Oracle: `CHAR` • SQL Server: `CHAR` • SQL Server, multi-lingual Unicode: `NCHAR` • DB2: `CHAR` Do **not** change the value of this property.
`cc.datepicture`	Specifies how Content Server creates a date/time literal. Value: `{ts '$date'}` Do **not** change the value of this property.
`cc.datetime`	Specifies the SQL string for defining a date/time field. Possible values: • Oracle 8: `DATE` • Oracle 9: `TIMESTAMP` • SQL Server: `DATETIME` • DB2: `TIMESTAMP` Do **not** change the value of this property.
`cc.double`	Specifies the SQL string for defining a double field. Possible values: • Oracle: `NUMBER(38,10)` • SQL Server: `NUMERIC(28,10)` • DB2: `FLOAT` Do **not** change the value of this property.

Properties in `futuretense.ini`: **Database** Tab *(continued)*

Property	Description
`cc.forcelower`	Specifies whether the column names for the tables that Content Server creates have all lowercase letters. Possible values: • Oracle: `true` • SQL Server: `false` • DB2: `true` Do **not** change the value of this property.
`cc.ignoreTblCase`	Determines whether Content Server ignores case when assessing table names. Possible values: `yes` | `no` For example, if "tablename" and "TABLENAME" would be considered different tables in your database, set this value to `no`. Possible values: • Oracle: `yes` • SQL Server: `yes` • DB2: `yes` Do **not** change the value of this property.
`cc.integer`	Specifies the SQL string for defining a 32-bit integer field. Possible values: • Oracle: `NUMBER(10)` • SQL Server: `INT` • DB2: `INTEGER` Do **not** change the value of this property.
`cc.maxvarcharsize`	Specifies the maximum size of a varchar column for your database. Possible values: • Oracle: `2000` • SQL Server: `8000` • DB2: `4000` Do **not** change the value of this property.

Properties in `futuretense.ini`: **Database** Tab *(continued)*

Property	Description
`cc.null`	Specifies the SQL string for defining a field which allows NULL values; this is nonstandard, though most databases support NULL. Possible values: • Oracle: `NULL` • SQL Server: `NULL` • DB2: `blank` Do **not** change the value of this property.
`cc.numeric`	Specifies the SQL string for defining a numeric field. Possible Values: • Oracle: `NUMBER` • SQL Server: `NUMERIC` • DB2: `NUMERIC` Do **not** change the value of this property.
`cc.primary`	Specifies the SQL string that defines a primary key. Possible values: • Oracle: `PRIMARY KEY NOT NULL` • SQL Server: `PRIMARY KEY NOT NULL` • DB2: `PRIMARY KEY NOT NULL` Do **not** change the value of this property.
`cc.queryablemaxvarcharlength`	Specifies the maximum size of a `varchar` column that can be queried based on the database driver in use. • Oracle Thin: `255` • Oracle TypeII: `2000` • SQL Server: `255` • DB2: `255`
`cc.rename`	Specifies the SQL string that renames a table in the database, as required by your database vendor. Possible values: • Oracle: `rename %1 to %2` • SQL Server: `execute sp_rename %1,%2` • DB2: `rename %1 to %2` Do **not** change the value of this property.

Properties in `futuretense.ini`: **Database** Tab *(continued)*

Property	**Description**
`cc.smallint`	Specifies the SQL string for defining a 16-bit integer field. Possible values: • Oracle: `NUMBER(5)` • SQL Server: `SMALLINT` • DB2: `SMALLINT` Do **not** change the value of this property.
`cc.stringpicture`	Specifies how Content Server creates a string literal. Possible values: • Oracle: `'$string'` • SQL Server: `'$string'` • SQL Server, multi-lingual Unicode: `N '$string'` • DB2: `'$string'` Do **not** change the value of this property.
`cc.unique`	Specifies the SQL string for defining a unique field. Possible values: • Oracle: `UNIQUE NOT NULL` • SQL Server: `UNIQUE NOT NULL` • DB2: `UNIQUE NOT NULL` Do **not** change the value of this property.
`cc.varchar`	Specifies the SQL string for defining a VARCHAR data type. Possible values: `VARCHAR`, for all supported databases except SQL Server, multi-lingual Unicode which is set to `NVARCHAR` Do **not** change the value of this property.
`cs.dbconnpicture`	Specifies the format of the database connection string used by JNDI datasources: • WebLogic 6.1: `$dsn` • SunOne: `jdbc/$dsn` • WebSphere 4.0: `$dsn` Do **not** change the value of this property.

Properties in `futuretense.ini`: **Database** Tab *(continued)*

Property	Description
`cs.dbtype`	Defines the type of database you are connecting to. Do **not** change the value of this property.
`cs.dsn`	Contains the database JNDI data source name for connecting to your database. Do **not** change the value of this property.
`cs.privpassword`	Specifies the password for the database account name used for read/write access (`cs.privuser`). The value is encrypted.
`cs.privuser`	Specifies the database account name to use for read/write access to the database. For security reasons, be sure that your system is not using the default user name/password combination. Default value: `ftuser` (set during installation)

futuretense.ini: Debug Tab

The **Debug** tab holds the properties that enable various kinds of Content Server debug logging (and one property for the DebugServer servlet). When the `ft.debug` property is set to `yes`, the ContentServer servlet writes various error and status messages to the `futuretense.txt` file, located in the Content Server installation directory.

If you enable debug logging, note the following:

- Delete or archive the `futuretense.txt` file frequently because a large log file can affect Content Server performance.
- Because enabling any of the debug logging options can affect performance, you should not enable these options on a management or delivery system that is live.
- By default, all debug log messages go into a single log file, which can make debugging more difficult. To put debug messages into a separate log file, set the **`logging.per-client-log`** property to `true`, which enables browser-based debugging. When this property is set to `true`, Content Server creates a log file for each browser IP address. Each file is stored in the same directory as the `futuretense.txt` file and is created according to the following naming convention: `futuretense.IPaddress.txt`.

  You can then use the `exportlog` argument of the `CATALOGMANAGER` tag to retrieve the log file for the IP address of the browser that you are using.

Properties in `futuretense.ini`: **Debug** Tab

Property	Description
`ft.debugport`	Specifies the port that DebugServer uses to communicate with the template debugger utility. The port number must be greater than 1024. Default value: `1025`
`ft.suppressPasswords`	Prevents any input or session variables containing the strings "`password`" or "`PASSWORD`" in their names from being logged. Default value: `true` Specify `false` to include passwords in the log.
`ft.suppressPasswordNames`	Specifies variable names that are used as passwords and should be suppressed when `ft.suppressPasswords` is set to `true`. The Cheetah installer now sets this property to: `REMOTEPASS;pubtgt:factors;factors` to suppress its mirror publish target passwords in the log.

futuretense.ini: Email Tab

The **Email** tab holds the properties that configure the Content Server e-mail system features. Note that there is an additional property on the **Preference** tab in `futuretense_xcel.ini` that enables the CS-Direct workflow e-mail system that sends notices to workflow participants when they are assigned assets through a workflow process.

Properties in `futuretense.ini`: **Email** Tab

Property	Description
`cs.emailaccount`	Specifies the user account name to be used for sending outgoing mail. This is the account name on the SMTP server. If SMTP authentication is required, you must set a value for this property.
`cs.emailauthenticator`	Specifies the class that is used as the authenticator for mail operations. Default value: `com.openmarket.framework.mail.ICSAuthenticator`
`cs.emailcharset`	Specifies the default character set that is used for the text in the subject of an e-mail message. Examples: • Latin1: `text/html; charset=ISO-8859-1` • Japanese (Shift_JIS): `text/html; charset=Shift_JIS` • UTF-8: `text/html; charset=UTF-8` If this is blank, there is no default value.
`cs.emailcontenttype`	Specifies the default character set that is used for the text in the body of an e-mail message. Examples: • Latin1: `text/html; charset=ISO-8859-1` • Japanese (Shift_JIS): `text/html; charset=Shift_JIS` • UTF-8: `text/html; charset=UTF-8` If this is blank, it reverts to the default `text/plain`
`cs.emailhost`	Defines the SMTP (e-mail server) host that is used by the ContentServer servlet to create and deliver e-mail messages. A valid value is required to send or receive mail.

Properties in `futuretense.ini`: **Email** Tab *(continued)*

Property	Description
`cs.emailpassword`	Specifies the password for the e-mail account used by Content Server (specified by `cs.emailaccount`). A valid value is required to receive mail.

`cs.emailreturnto`	Specifies the e-mail address from which mail is sent. That is, the e-mail address that appears in the From field of an e-mail message.
	Use one of the following formats:
	`user@domain.com`
	`Full Name <user@domain.com>`
	A valid value is required to send mail.

futuretense.ini: Export/Mirror Tab

The **Export/Mirror** tab holds the properties that configure the Content Server Export and Mirror APIs that are used by the CS-Direct publishing system. These properties work in conjunction with the properties located on the **Publishing** tab in the `futuretense_xcel.ini` file, which is described in the "CS-Direct Property Files" section.

When configuring the publishing operations for your Content Server systems, think of your individual systems (development, management, delivery) in the following terms:

- **Source**, which denotes the Content Server database that is the source for a publishing session. Because you can mirror assets and site configuration information from any Content Server system to any other Content Server system, the source is not necessarily the Content Server management system.
- **Target**, which denotes either the Content Server database that you are mirroring to or the file server that you are exporting to.

For more information about publishing, see the *Content Server Administrator's Guide*

Properties in `futuretense.ini`: **Export/Mirror** Tab

Property	Description
`cs.mirrorhttpversion`	Specifies the HTTP protocol version to use to communicate with the Content Server target databases. Default value: `1`
`cs.mirrorpassword`	Specifies the password for the mirror user on the target systems to which this system publishes. You set this value when you set up your CS system for publishing. For information, see the *Content Server Administrator's Guide*.
`cs.mirrorproxyserver`	Specifies the firewall server's IP address or name for the target system that this (source) system publishes to when the target and the source are separated by a firewall. You set this value when you set up your CS system for publishing. Possible values: `your_server_name` or `your_server_ip_address` For information, see the *Content Server Administrator's Guide*.

Properties in `futuretense.ini`: **Export/Mirror** Tab *(continued)*

Property	Description
`cs.mirrorproxyserverport`	Specifies the port number of the firewall server for the CS system that this system publishes to when the target system is separated from the source with a firewall. You set this value when you set up your CS system for publishing. Possible value: `port_number` For information, see the *Content Server Administrator's Guide*.
`cs.mirrorrowsperpost`	Specifies the number of table rows that can be mirrored during each HTTP POST during a mirror operation If you are mirroring data that contains URL fields, you should set this to a low number because web servers impose a limit on the size of post packets. If you are mirroring data that contains only text, you can set this to a higher number. Default value: `6` **Note:** For best performance, do not increase the value above 12. If your database is configured for UTF-8 and holds non-ASCI content, you must set this value to 4 or lower.
`cs.mirrorthreads`	Specifies the number of threads to allocate to a mirror operation. Default value: `2` **Note:** For best performance, do not increase the value above 8.
`cs.mirroruser`	Specifies the name of the mirror user on the target system that this (source) system publishes to. You set this value when you set up your system for publishing. For information, see the *Content Server Administrator's Guide.*
`cs.pgexportfolder`	Specifies the base export directory for the HTML files that are created when assets are published with the Export to Disk delivery type. • Windows NT example: `c:/FutureTense/export` • Solaris example: `/export/home/FutureTense/pgexport`

futuretense.ini: JSP Tab

The **JSP** tab holds the properties that supply information that Content Server references when serving Java Server Pages files. If your Content Server system uses WebLogic, note that there are additional, WebLogic-only, JSP properties in the `jsprefresh.ini` file. See "jsprefresh.ini (WebLogic Only)" on page 90.

Properties in `futuretense.ini`: **JSP** Tab

Property	Description
`cs.jspclear`	Configures the Content Server engine to delete any previously deployed JSP files and clear the application server's working folder (temp and class files) when the Content Server engine executes the first JSP deployed by Content Server. The working folder is defined by the application server. Possible values: `true` | `false`
`cs.jsppath`	Specifies the virtual root (zone) for executing deployed JSP pages. This property is used in conjunction with `cs.jsproot`, so the two properties must be in sync. Default value: synchronized with the WebLogic setting for `cs.jsproot` and is set at installation. **Note:** Do not change the value of this property after installation.
`cs.jsprefresh`	When the Content Server engine deploys a new or changed JSP element, the application server may require special processing to complete the deployment of the JSP. In those cases, this property specifies the name of the class which completes the deployment. In other cases, it is left blank. For WebLogic 6.1, the default value is `com.divine.wl6special`. For others, the default is blank.
`cs.jspresponsewrapper`	Specifies whether the application server requires the PrintWriter when it runs a JSP element. The Content Server installation sets this to an appropriate value based on the type of application server you are using. Do **not** change the value of this property.

Properties in `futuretense.ini`: **JSP** Tab *(continued)*

Property	Description
`cs.jsproot`	Specifies the folder where the application server expects to find JSP files. Some application servers allow this value to be modified by using property settings. WebLogic defines `weblogic.httpd.initArgs` for its JSPServlet object. The `cs.jsproot` property is used in conjunction with `cs.jsppath`, so the two must be in sync and both are set at installation. Do **not** change the value of this property after installation.
`cs.jspwork`	Specifies the directory where class files are created by the application server when executing JSP pages. This is not a required property and can be left blank, but it is normally set at installation. For more information, see `cs.jspclear`.

futuretense.ini: Misc Tab

The **Misc** tab holds miscellaneous properties such as the amount of idle time a connection can have before Content Server logs out of a connection and whether Content Server loads cache synchronization processing.

Properties in `futuretense.ini`: **Misc** Tab

Property	Description
`cs.charset`	Specifies a variable that can be included in HTML forms as a hidden variable. The variable, which gets set by the browser, specifies the text encoding of the form data that Content Server must process. Default value: `_charset_` Do **not** change the value of this property.
`cs.contenttype`	Specifies the default character set to use for HTTP headers (streaming text). Default value: `text/html; charset=UTF-8` Specify a value that is appropriate for the online site that your CS system is delivering. Examples: • Latin1: `text/html; charset=ISO-8859-1` • Japanese (Shift_JIS): `text/html; charset=Shift_JIS`
`cs.disksize`	Specifies the size limit in bytes for keeping uploaded files in memory while they are being posted. If an uploaded file is larger than the value specified, Content Server streams it to a temporary file until it is finished evaluating a page. This prevents excessive memory use and helps to prevent denial-of-service attacks. Default value: `102400`
`cs.documentation`	Specifies the URL of the CS documentation. By default, this property is set to a FatWire documentation web site. If you prefer, you can download the most recent documentation kit from that web site, install it somewhere on your network, and then set this property to point to that location rather than to the FatWire documentation web site.

Properties in `futuretense.ini`: **Misc** Tab *(continued)*

Property	**Description**
`cs.HTTP_HOST`	Specifies the HTTP host for CS systems on which the Web server does not reside on the application server machine (that is, it is a web connector installation), or in cases where an alternative web server is used to serve pages. Use one of the following formats: *`hostname:port`* *`IPaddress:port`* Default value: blank
`cs.HTTP_PROTOCOL`	Specifies the HTTP protocol for CS systems where the web server does not reside on the application server machine (that is, it is a web connector installation), or in cases where an alternative web server is used to serve pages. Possible values: `http` or `https` or blank Default value: blank, which means the protocol is assumed to be http
`cs.urlfilerollup`	Determines how changes to the files for URL columns are tracked. When this value is set to `true`, the previous versions of the related file are tracked in a name sequence. For example, if a file called `filename.txt` has been edited three times, `filename.txt` is the oldest, `filename,1.txt` is the next oldest version, and `fsilename,2.txt` is the current version. When this value is set to `false`, the file name toggles between `filename.txt` and `filename,0.txt` on alternate updates. Default value: `false` for WebLogic 7.1 and `true` for WebLogic 6.1.
`cs.xmlfolder`	Specifies the working directory for HTML filtering. That is, if elements on your system use the `XMLFILTER` tag, temporary files are written to this directory. Default value: `$HOME/FutureTense/xmltemp`
`cs.xmlHeader`	This is the XML Header streamed out for SOAP responses.

futuretense.ini: Page Cache Tab

The **Page Cache** tab holds the properties that configure Content Server's page cache settings. Content Server's page cache is monitored and maintained by the CacheManager. Content Server caching enables you to cache both complete web pages and their individual components (or pagelets).

To set up page caching on a Content Server system, you configure properties for the CacheManager and the Satellite Server servlets. In addition, there are properties for configuring BlobServer, the servlet that serves blobs and caches blobs both through Content Server and the Satellite servlet.

For more information:

- Page caching – see the page caching chapter in the *Content Server Developer's Guide*. This chapter describes how Content Server's Cache Manager, the Satellite servlets, and the BlobServer servlet interact and work together.
- Satellite Server properties – see "Satellite Server Tab" on page 80 and "satellite.properties" on page 107.
- Resultset caching properties – see "ResultSet Cache Tab" on page 76.
- BlobServer properties – see "Blob Server Tab" on page 52.

Properties in `futuretense.ini`: **Page Cache** Tab

Property	Description
`cs.alwaysusedisk`	Specifies the default cache setting for page entries in the `SiteCatalog` table that have no information in their `cacheinfo` column. If set to `yes`, then each page served from Content Server is cached to disk (the database), unless the value in that page entry's `cacheinfo` column specifies that it not be cached. Default value: `no`
`cs.freezeCache`	Specifies whether a cache maintenance event should regularly remove expired pages from the cache, or whether the expiration date of a page should be checked only when that page is requested. Set the value to `yes` if you do not want an event to regularly remove expired pages from the cache. Default value: `no`
`cs.IItemList`	Defines the file that is used for the ItemList interface that is used by the Cache Manager. Default value: `COM.FutureTense.Export.ItemList` **Note:** The default value is provided for reference only. Do **not** change the value of this property.

Properties in `futuretense.ini`: **Page Cache** Tab *(continued)*

Property	Description
`cs.nocache`	Provides you with the ability to disable all disk-based page caching (pages will still be cached in memory). Use this property to temporarily shut down page caching when you are debugging your site, but do not leave this value set to `true` on a live system. Default value: `false`
`cs.pgCacheTimeout`	Specifies the number of minutes that the CacheManager holds a page in the page cache. A value of `0` (zero) disables timeout, which means pages never expire. However, no matter what the setting for this property, CacheManager refreshes a cached page if the publishing system reports that it published any of the assets on that page. Default value: `1440`
`cs.recordBlobInventory`	Allows users to work with blob dependencies. To record blobs inventory, set the value of this property to `true`. If full backward compatibility with V5.0 is desired, set the value of this property to `false`.
`cs.requiresessioncookies`	Specifies whether session ID information can be held in cookies or whether Content Server must encode session data into the links. Set to `true` (the default) if Content Server expects session cookies to be enabled. This allows all pages to be cached and does not encode the session id into any links. A value of `false` enables URL rewriting, with a negative effect on page caching performance. Default value: `true`
`cc.SystemPageCacheTimeout`	Specifies the number of minutes a cached page is held in memory (cached pages are cached both to disk and to memory). Default value: `1440` (that is, 24 hours)
`cc.SystemPageCacheCSz`	Specifies the maximum number of pages that can be cached in memory. Pages are cached both in memory and to disk (database). This property specifies the number of pages cached to memory, not to disk. Default value: `500`

futuretense.ini: ResultSet Cache Tab

The lowest level of cache support is database query, or resultset, caching. The **ResultSet Cache** tab holds properties that configure Content Server's resultset caching. For information about resultset caching and queries on your Content Server system, see the "Data Design" section in the *Content Server Developer's Guide*.

The three main resultset caching properties are `cc.cacheResults`, `cc.cacheResultsTimeout`, and `cc.CachResultsAbs`. They specify the default number of resultsets to cache in memory, the default amount of time to keep resultsets cached in memory, and how to calculate the expiration time.

The default values specified for these properties are used to determine how to cache the resultsets of any table in the Content Server database that does not have table-specific resultset caching properties for it.

You can create three resultset caching properties for each table in the Content Server database. Use the following syntax:

```
cc.<tablename>Csz=<number of resultsets>
cc.<tablename>Timeout=<number of minutes>
cc.<tablename>Abs=<true or false>
```

You can create as many table-specific resultset caching properties as are needed to implement your resultset caching strategy for each of your Content Server systems (which means that the values for these properties are different on each system).

Most of the Content Server system tables have table-specific resultset caching properties set for them by default. These properties are displayed on the **ResultSet Cache** tab. However, when you create new table-specific resultset caching properties, they are displayed on the **User Defined** tab.

For information about page caching properties, see "Page Cache Tab" on page 74. For information about BlobServer caching properties, see "Blob Server Tab" on page 52.

The following table describes the properties that appear on the **ResultSet Cache** tab.

Properties in `futuretense.ini`: **ResultSet Cache** Tab

Property	Description
`cc.cacheResults`	Specifies the default number of resultsets to cache in memory. Note that this does not mean the number of records in a resultset, but the number of resultsets. Setting this value to `0` or `-1` disables resultset caching for all tables that do not have their own caching properties configured. **Caution:** Do **not** set this value to `0` or `-1`. If you do, the Content Server interface will fail to save assets properly. Possible values: *`n`* (number of resultsets) Default value: `500` To set a different value for a specific table, create a property for that table using the following format: `cc.<tablename>Csz=<number of resultsets>`
`cc.cacheResultsAbs`	Determines how to calculate the expiration time for the resultsets in the resultset cache. If the value is set to `true`, the expiration time for a resultset is absolute. For example, if `cc.cacheResultsTimeout` is set to 5 minutes, then 5 minutes after the resultset was cached, it is flushed form the cache. If the value is set to `false`, the expiration time for a resultset is based on its idle time. For example, if `cc.cacheResultsTimeout` is set to 5 minutes, the resultset is flushed from the cache 5 minutes after the last time it was requested rather than 5 minutes since it was originally cached. To set this value for a specific table, create a property for that table using the following format: `cc.<tablename>Abs=<true or false>`

Properties in `futuretense.ini`: **ResultSet Cache** Tab *(continued)*

Property	Description
`cc.cacheResultsTimeout`	Specifies the number of minutes to keep a resultset cached in memory. Setting this value to -1 means that there is no timeout value for tables that do not have their own caching properties configured. Possible values: *n* (in minutes), or -1 to disable for tables that use this default setting. Default value: `5` To set this value for a specific table, create a property for that table using the following format: `cc.<tablename>Timeout=<number of minutes>`
`cc.ElementCatalogCSz`	Specifies the number of resultsets to cache against the `ElementCatalog` table. For best performance this value should be set to the number of rows in the table. Default value: `1000`
`cc.ElementCatalogTimeout`	Specifies the number of minutes to keep idle resultsets for the `ElementCatalog` table in the resultset cache. Use -1 to disable timeout. Default value: `60`
`cc.SiteCatalogCSz`	Specifies the number of resultsets to cache against the `SiteCatalog` table. For best performance, this value should be set to the number of rows in the table. Default value: `1000`
`cc.SiteCatalogTimeout`	Specifies the number of minutes to keep idle resultsets for the `SiteCatalog` table in the resultset cache. Use -1 to disable timeout. Default value: `60`
`cc.SystemACLCSz`	Specifies the number of resultsets to cache against the `SystemACL` table. For best performance, this value should be proportional to the number of rows in the table. Default value: `25`

Properties in `futuretense.ini`: **ResultSet Cache** Tab *(continued)*

Property	Description
`cc.SystemACLTimeout`	Specifies the number of minutes to keep idle resultsets for the `SystemACL` table in the resultset cache. Default value: `-1` (which disables timeout for this table)
`cc.SystemInfoCSz`	Specifies the number of resultsets to cache against the `SystemInfo` table. For best performance, this value should be set to the number of rows in the table. Default value: `500`
`cc.SystemInfoTimeout`	Specifies the number of minutes to keep idle resultsets for the `SystemInfo` table in the resultset cache. Default value: `-1` (which disables timeout for this table)
`cc.SystemUsersCSz`	Specifies the number of resultsets to cache against the `SystemUsers` table. For best performance, this value should be proportional to the number of rows in the table. Default value: `100`
`cc.SystemUsersTimeout`	Specifies the number of minutes to keep idle resultsets for the `SystemUsers` table in the resultset cache. Default value: `-1` (which disables timeout for this table)
`ft.filecheck`	Specifies whether Content Server verifies the timestamp on data held in an upload field each time an item (like an element) with uploaded data is requested. On a management or delivery system, the same items are requested repeatedly and setting this value set to yes can slow the performance of the system. Set this property to `no` on management or delivery systems. You can set it to `yes` on a development system. Default value: `no`

futuretense.ini: Satellite Server Tab

The **Satellite Server** tab holds properties that describe how to communicate with any of the Satellite servlets (Satellite Server), whether they are running locally or on remote servers.

Additionally, because Satellite Server is installed by default on the server that hosts your Content Server application, each Content Server system also has a `satellite.properties` file that configures the local Satellite servlet. (Note that when Satellite Server is also running on a remote server, that server also has a `satellite.properties` file.) For information about the properties in that file, see "satellite.properties" on page 107.

The following table describes the properties on the server that hosts Content Server that support communications with all Satellite servlets.

The value for each property is a comma-separated list. The ordinal position of an item in the list is what associates the host, user name, and password for each Satellite servlet. For example, the third host named in `cs.satellitehosts` is accessed using the third user account named in `cs.satelliteusers`, giving the third password listed in `cs.satellitepassword`.

Properties in `futuretense.ini`: **Satellite Server** Tab

Property	Description
`cs.PastramiEngine`	Defines the name of the implementing class of the PastramiEngine interface. Leave this property null to disable this feature entirely. The standard value for this property is: `com.divine.pastrami.PushEngine`
`satellite.blob.cachecontrol.default`	Specifies a default value for the `cachecontrol` parameter for the `satellite.blob`, and `RENDER.SATELLITEBLOB` tags and their JSP equivalents. Default value: blank Set this property to a value that is appropriate for the majority of your blobs, and then use the `cachecontrol` parameter with the `satellite.blob` and `RENDER.SATELLITEBLOB` tags to override this value for individual blobs. Use the following format to set a value: `hours:minutes:seconds daysOfWeek/daysOfMonth/months` For more information about this format, see the description of the `expiration` property in the section "satellite.properties" on page 107.

Properties in `futuretense.ini`: **Satellite Server** Tab *(continued)*

Property	Description
`satellite.page.cachecontrol.default`	**Deprecated.** Specifies a default value for the `cachecontrol` parameter for the `satellite.page`, and `RENDER.SATELLITEPAGE` tags and their JSP equivalents. Default value: blank Set this property to a value that is appropriate for the majority of your pages and pagelets, and then use the `cachecontrol` parameter with the `satellite.page` and `RENDER.SATELLITEPAGE` tags to override this value for individual pages and pagelets. Use the following format to set a value: *`hours:minutes:seconds daysOfWeek/daysOfMonth/months`* For more information about this format, see the description of the `expiration` property in the section "satellite.properties" on page 107.

futuretense.ini: Search Tab

The **Search** tab holds properties that Content Server uses to obtain configuration information about your third-party search engine, if your Content Server system uses one. If your organization purchased one of the search engine modules, it was installed when your Content Server system was installed.

Note: The **Search** tab displays properties for all supported third-party search engine modules. In this section, the properties are organized by search engine.

Properties in `futuretense.ini`: **Search** Tab

Property	Description
`av.cjkquery`	**Deprecated.** Specifies whether AltaVista must handle queries that use Chinese, Japanese, or Korean characters. Default value: `no` Set to `yes` if your CS system is localized for Chinese, Japanese, or Korean.
`av.defaultindex`	**Deprecated.** Specifies the AltaVista search index to open if none is specified as an XML parameter.
`av.license`	**Deprecated.** Specifies an explicit license value. This property exists only if you have obtained your own license for the AltaVista search engine. It replaces `av.oemkeytype`.
`av.oemkeytype`	**Deprecated.** Specifies one of the two embedded keys: For a management system, set this value to `management` to allow up to 5 million entries. You are not authorized to use this value on a delivery site where there is public access. For a delivery system, set this value to `delivery` to allow up to 250,000 entries. If you need more entries, you must obtain a specific license from AltaVista and use the `av.license` property instead of this one.

Properties in `futuretense.ini`: **Search** Tab *(continued)*

Property	Description
`cs.searchengine`	Specifies the search engine. Set this property to `AV` (valid for Content Server upgrades) or to `verity`. Default value: `verity`
`searcheng.apidebug`	Controls whether search engine API calls should be logged and their returned values recorded. Possible values: `yes` | `no`
`searcheng.debug`	Controls whether debugging for the search engine connector implementation should be turned on or off. Possible values: `yes` | `no`
`searcheng.enginedebug`	Controls whether debugging in the third party search engine implementation is on or off. Possible values: `yes` | `no`
`searcheng.keeptemps`	Controls whether temporary files used by the search engine implementation are kept or not. Possible values: `yes` | `no`
`searcheng.querydebug`	Determines if the constructed query is written to the log before making the search. Possible values: `yes` | `no`
`searcheng.regdebug`	Controls whether failed search engine registration attempts should be logged. Possible values: `yes` | `no`
`searcheng.usedebugse`	If `debugse` is installed, controls whether all search engine interface calls/returns are logged. Possible values: `yes` | `no`
`verity.charset`	Specifies the name of the locale to use for all internal Verity engine operations. This property corresponds to a subdirectory of the common directory where the locale is defined. The property is optional.
`verity.debug`	Controls whether the Verity Search Engine should put debug messages into the log file. Possible values: `yes` | `no`
`verity.defaultindex`	The Verity search index to open if none is specified as an XML or Java parameter.

Properties in `futuretense.ini`: **Search** Tab *(continued)*

Property	Description
`verity.defaultparser`	Specifies the default parser. Possible values: • `Simple` • `FreeText` • `BoolPlus`
`verity.indexinginterval`	Specifies the time interval in milliseconds between each background mkvdk index process provided by Verity, Inc. As background indexing uses a large amount of memory and CPU, it is best to allow the front-end CS-Direct interfaces a chance to render. The bigger the interval, the faster is interface rendering, but the longer it takes for a search to be updated. This property is used only when the `verity.syncindexing` property is set to `no`. Default value: `10,000`
`verity.knowledgebasepath`	The path to the Verity topic that is used for queries. This property is optional.
`verity.locale`	Specifies the name of the locale to use for all internal Verity engine operations. This field corresponds to a subdirectory of the common directory where the locale is defined. This field is optional.
`verity.organization`	The organization specified in your Verity Information Server License Key. This property is case-sensitive and space-sensitive.
`verity.path`	The path to the Verity Information Server directory.
`verity.signature`	The signature text as specified in your Verity License Key. This property is case- and space-sensitive.
`verity.syncindexing`	Specifies whether the Verity search engine is to process indexes synchronously. If processing is not synchronous, the index process runs asynchronously in the background so that performance can be improved. In the current version, this property applies only to flex assets; basic assets always use synchronous indexing. Possible values: `yes, no` Default value: `yes`

Properties in `futuretense.ini`: **Search** Tab *(continued)*

Property	Description
`verity.templatepath`	The path of the Verity vdktemplate directory. This is a subdirectory of the FutureTense directory. For example, In Windows NT: `C:\FutureTense\vdktemplate` In Solaris: `/export/home/FutureTense/vdktemplate`

futuretense.ini: User Defined Tab

The **User Defined** tab displays custom properties that are not created by the core Content Server product but are used by Content Server and its CS content applications.

This tab displays the following kinds of properties:

- Two custom properties for the sample site, if the Content Server portal sample site is installed.
- A custom property that specifies the version number of each CS content application that is installed.
- Properties that specify the name of the primary key column for a content table (catalog). For information about content tables, see the "Data Design" section in the *Content Server Developer's Guide*. The installation program for CS-Direct creates custom properties of this type.
- Custom properties that specify resultset caching settings for individual tables in the Content Server database. The installation program for CS-Direct creates custom properties of this type.

The following table lists all of the properties that any of the Content Server products create in the `futuretense.ini` file, which means that they appear on the **User Defined** tab. Note that your system may have additional properties that are not included in this list.

Properties in `futuretense.ini`: **UserDefined** Tab

Property	Description
`analysisconnector.version`	The version number of Analytics Bridge, if it is installed. Do **not** change the value of this property.
`catalogcentre.version`	The version number of CS-Direct Advantage, if it is installed. CS-Direct Advantage was named Catalog Centre in previous versions of the product. Do **not** change the value of this property.
`cc.AssetTypeCSz`	The number of resultsets to cache against the `AssetType` table, an object table that is created by the CS-Direct installation. Default value: `50`
`cc.CategoryCSz`	The number of resultsets to cache against the Category table, an object table that the CS-Direct installation creates for asset types. Default value: `50`

Properties in `futuretense.ini`: **UserDefined** Tab *(continued)*

Property	Description
`cc.ComparatorsKey`	The primary key of the `Comparator` table, a content table that the CS-Direct installation to hold field comparator classes. Value: `name` Do **not** change the value of this property.
`cc.FiltersKey`	Value: `name`
`cc.MimeTypeKey`	The primary key of the `MimeType` table, a content table that the CS-Direct installation creates to store the mimetypes of the documents that it handles. Value: `mimetype` Do **not** change the value of this property.
`cc.PreviewgenKey`	Value: `name`
`cc.SourceKey`	The primary key of the `Source` table, a content table that the CS-Direct installation creates for asset types. Value: `source` Do not change the value of this property.
`cc.StatusCodeCSz`	The number of resultsets to cache against the `StatusCode` table, a content table that the CS-Direct installation creates for asset types. Default value: `10`
`cc.StatusCodeKey`	The primary key of the `StatusCode` table. Value: `statuscode` Do not change the value of this property.
`commerceconnector.version`	The version number of the Commerce Connector utility, if it is installed. Do **not** change the value of this property.
`contentcentre.version`	The version number of CS-Direct if it is installed. CS-Direct was named Content Centre in previous versions of the product. Do **not** change the value of this property.
`cs.dbencoding`	Value: `UTF-8`

Properties in futuretense.ini: **UserDefined** Tab *(continued)*

Property	Description
cs.requestfactory	Value: com.fatwire.cs.portals.portlet.PortletRequestFactory
ft.servletoutputstream	Value: false
image.time	Value: 5:0:0 */*/*
marketingstudio.version	The version number of Engage, if it is installed. Engage was named Marketing Studio in previous versions of the product. Do **not** change the value of this property.
page.time	Value: *:0,5,10,15,20,25,30,35,40,45,50,55:0 */*/*
soap.binaryRowsType	A CS-Direct system property used to instantiate server-side objects in response to SOAP requests made by web services. Do **not** change the value of this property.
soap.iList	A CS-Direct system property used to instantiate server-side objects in response to SOAP requests made by web services. Do **not** change the value of this property.
soap.likeConstraint	A CS-Direct system property used to instantiate server-side objects in response to SOAP requests made by web services. Do **not** change the value of this property.
soap.listRowsType	A CS-Direct system property used to instantiate server-side objects in response to SOAP requests made by web services. Do **not** change the value of this property.
soap.nestedConstraint	A CS-Direct system property used to instantiate server-side objects in response to SOAP requests made by web services. Do **not** change the value of this property.
soap.rangeConstraint	A CS-Direct system property used to instantiate server-side objects in response to SOAP requests made by web services. Do **not** change the value of this property.

Properties in `futuretense.ini`: **UserDefined** Tab *(continued)*

Property	Description
`soap.richTextConstraint`	A CS-Direct system property used to instantiate server-side objects in response to SOAP requests made by web services. Do **not** change the value of this property.
`soap.searchstate`	A CS-Direct system property used to instantiate server-side objects in response to SOAP requests made by web services. Do **not** change the value of this property.
`soap.standardConstraint`	A CS-Direct system property used to instantiate server-side objects in response to SOAP requests made by web services. Do **not** change the value of this property.
`soap.stringRowsType`	A CS-Direct system property used to instantiate server-side objects in response to SOAP requests made by web services. Do **not** change the value of this property.
`soap.stringVarsType`	Value: `com.openmarket.basic.objects.StringVarsType`
`soap.urlRowsType`	A CS-Direct system property used to instantiate server-side objects in response to SOAP requests made by web services. Do **not** change the value of this property.
`soap.URLType`	A CS-Direct system property used to instantiate server-side objects in response to SOAP requests made by web services. Do **not** change the value of this property.

jsprefresh.ini (WebLogic Only)

The `jsprefresh.ini` file holds properties that provide additional information that Content Server needs in order to serve JSP files correctly when your application server is WebLogic. See also the properties in the section "JSP Tab" on page 70.

Properties in `jsprefresh.ini` (WebLogic Only)

Property	Description
`app`	Specifies the name of the application that was created in WebLogic to refer to Content Server. This value is set during installation. It should match the value set for the `component` property.
`component`	Specifies the name of the application that was created in WebLogic to refer to Content Server. This value is set during installation. It should match the value set for the `app` property.
`domain`	Specifies the domain in which WebLogic is configured. This value is set during installation.
`password`	Specifies the password for the user account for the WebLogic Admin Server. This value is encrypted. This value is set during installation.
`url`	Specifies the URL at which you access WebLogic. This value is set during the installation of a CS system and it follows this format: `http://servername:port`
`user`	Specifies the user name of the user account for the WebLogic Admin Server. This value is set during installation.
`version`	Specifies the version number of the WebLogic server. If you are using WebLogic 6.1, set it to `6`. If you are using WebLogic 7.x, set it to `7`.

logging.ini

The logging.ini file holds properties that configure the Logging module. The Logging module is deprecated and is used only by the Directory Services API. The Logging module writes messages to the futuretense.txt file.

This section presents the properties in the logging.ini file, by the tabs where the properties are organized to form a functional group.

Tabs in the logging.ini are the following, in alphabetical order:

- Global Data Tab
- Message Resources Tab
- User Defined Tab

logging.ini: Global Data Tab

The **Global Data** tab holds one property:

Properties in `logging.ini`: **Global Data** Tab

Property	Description
`log.filterLevel`	The severity threshold that determines the amount of messages that the Logging module writes to the log. Possible values: • `info`: writes all informational, warning, error, severe, and fatal messages. • `warning`: excludes informational messages; writes warning, error, severe, and fatal messages. • `error`: excludes warning and informational messages; writes error, severe, and fatal messages. • `severe`: excludes error, warning, and informational messages; writes severe and fatal messages. • `fatal`: writes fatal messages only.

logging.ini: Message Resources Tab

The Message Resources tab holds properties that provide logical mappings for the message bundles that are located, delivered, and reported by the Logging Module when the module is invoked by various components of the applications.

Caution. Do **not** change the values of any properties on this tab.

Properties in `logging.ini`: **Message Resources** Tab

Property	Description
`log.Directory.messages`	The Java resource bundle to use for the Directory Services API. Default value: `com.openmarket.directory.DirectoryResources`. Do **not** change the value of this property.
`log.Logger.messages`	The Java resource bundle to use for the Logging module. Default value: `com.openmarket.directory.LoggerResources`. Do **not** change the value of this property
`log.transformer.messages`	The class providing message resources for the Default Transformer WebMethodsEnterprise Connector Subsystem.
`log.wmentconnector.messages`	The class providing message resources for the WebMethodsEnterprise Connector System.

logging.ini: User Defined Tab

Properties in `logging.ini`: **User Defined** Tab

Property	Description
Note: This tab, by default, holds no properties.	

omii.ini

The omii.ini file holds properties that define the installation options that are chosen for Content Server during its installation.

This section presents the properties in the omii.ini file, by the tabs where the properties are organized to form a functional group.

The following tab is displayed for the omii.ini file:

- User Defined Tab

Caution. Do **not** change the values of any properties on this tab.

omii.ini: User Defined Tab

Properties in omii.ini: **User Defined** Tab

Property	Value
$$$AppServerType	WLP8
$$$Product0	true
$$$Product1	true
$BASEDIR	c:
$HOMEDIR	C:\Documents and Settings\Administrator\
$INSTALLDEST	C:/Sparkfatwire
$INSTALLDIR	C:/csbuild/ContentServer
$LOCALHOST	fattestntsrv9
$LOGDIR	C:/Sparkfatwire/ominstallinfo
ACMSSampleSiteBF	true
ACStoragePath	C:\Sparkfatwire\Shared\\queue\\
AdditionalPathToLib	server
AnalysisConnector	true
AnalysisConnectorCurrent Version	0.0.0
bCSInstall	TRUE
bDisplayPropEditor	FALSE
bDoAllACProperties	true
bDoAllAFKProperties	true
bDoAllCatCProperties	true
bDoAllCCProperties	true
bDoAllComCProperties	true
bDoAllMSProperties	true
bDoDBInitialization	true
bDoFromAC36xTo40	true
bDoFromAC40To45	true
bDoFromAC45To51	true
bDoFromAC51To61	true
bDoFromAFK36xTo40	true
bDoFromAFK40To45	true
bDoFromAFK45To51	true
bDoFromAFK51To61	true

Properties in `omii.ini`: **User Defined** Tab *(continued)*

Property	Value
bDoFromCatC36xTo40	true
bDoFromCatC40To45	true
bDoFromCatC45To51	true
bDoFromCatC51To61	true
bDoFromCC36xTo40	true
bDoFromCC40To45	true
bDoFromCC45To51	true
bDoFromCC51To61	true
bDoFromComC36xTo40	true
bDoFromComC40To45	true
bDoFromComC45To51	true
bDoFromComC51To61	true
bDoFromMS36xTo40	true
bDoFromMS40To45	true
bDoFromMS45To51	true
bDoFromMS51To61	true
bDoUpgradeFromCC36xTo40	false
bDoUpgradeFromCC40To45	false
bDoUpgradeFromCC45To51	false
bDoUpgradeFromCC51To61	false
bJSPResponseWrapper	false
bLDAPIntegration	TRUE
bMandatoryLegacyInstall	false
bRunningManaged	FALSE
bSunIdentityServer	false
bUrlFileRollUp	false
CartsetDir	C:\Sparkfatwire\Shared\gator\cartset
CatalogCentre	true
CatalogCentreCurrentVersion	0.0.0
CatCSampleAssets	true
CatCSampleAssetsCurrentVersion	0.0.0
CCSampleAssets	true
CCSampleAssetsBF	true

Properties in `omii.ini`: **User Defined** Tab *(continued)*

Property	Value
CCSampleAssetsBFCurrent Version	0.0.0
CCSampleAssetsCurrent Version	0.0.0
CCSampleAssetsHW	true
CCSampleAssetsHWCurrent Version	0.0.0
CCSampleSiteBF	true
CCSampleSiteHW	true
CCVersionString	0.0.0
ClassesComFolder	C:/bea/user_projects/applications/SparkApp/applications/spark/WEB-INF/classes/COM
ClassesCrysecFolder	C:/bea/user_projects/applications/SparkApp/applications/spark/WEB-INF/classes/crysec
ClassesSComFolder	C:/bea/user_projects/applications/SparkApp/applications/spark/WEB-INF/classes/com
CommerceConnector	true
CommerceConnectorCurrentVersion	0.0.0
Content Server XML Bridge	true
Content Server XML Bridge Sample	true
ContentCentre	true
ContentCentreCurrent Version	0.0.0
CSAppServerLibBin	c:/bea/weblogic81/server/lib/solaris
CSAppServerLibBinNT	c:/bea/weblogic81/server/bin
CSAppsVersion	D
CSConnectPrefix	http
CSConnectString	http://fattestntsrv9:7001/spark/
CSInstallAccountName	ContentServer
CSInstallAccountPassword	675286278df204915f228fd298c09388
CSInstallAdminDomainName	portalDomain
CSInstallAppServerPath	c:/bea/weblogic81
CSInstallAppServerRootPath	C:/bea/user_projects/applications/SparkApp

Properties in `omii.ini`: **User Defined** Tab *(continued)*

Property	Value
CSInstallAppServerType	WLP8
CSInstallDatabaseType	SQL Server
CSInstallDBDSN	sparkDataSource
CSInstallDBPrivUser	CS.Empty
CSInstallDirectory	C:/Sparkfatwire
CSInstallPlatformType	PORTALSERVER
CSInstallSharedDirectory	C:/Sparkfatwire/Shared
CSInstallSpark	false
CSInstallType	single
CSInstallWebRootPath	C:/Sparkfatwire/futuretense_cs
CSInstallWebServerPort	7001
CSInstallWebServer Address	fattestntsrv9
CSInstallWLDomainPath	C:/bea/user_projects/applications/ SparkApp
CSInstallWLWebAppName	spark
CSPortal	true
Date	041805141857
DefaultBase	C:\Sparkfatwire\Shared\Storage
Development	true
DMSampleAssets	true
DMSampleAssetsCurrent Version	0.0.0
FatWire Corporation Installation	Content Server Applications
GatorCCUrlPath	C:\Sparkfatwire\Shared\ccurl\
GatorHistoryDir	C:\Sparkfatwire\Shared\history\
GatorPublishDir	C:\Sparkfatwire\Shared\publish\
GatorSearchDir	C:\Sparkfatwire\Shared\gator\search dir\
GatorSessionData	C:\Sparkfatwire\Shared\session\
GatorVisitorGenclasspath	C:\Sparkfatwire\Shared\classes\
GatorVisitorRuleXML	C:\Sparkfatwire\Shared\ruleset\
GatorVisPath	C:\Sparkfatwire\
GrantList	SparkAdmin,GeneralAdmin,WorkflowAdm in,SiteAdmin

Properties in `omii.ini`: **User Defined** Tab *(continued)*

Property	Value
InstallACbDisplayProp Editor	false
InstallCatCbDisplayProp Editor	false
InstallCCbDisplayProp Editor	false
IsPrimaryClusterMember	true
LDAPHost	fattestntsrv9
LDAPJNDIPassword	675286278df204915f228fd298c09388
LDAPPort	7001
LinuxAddOnPathToLib	false
LockDir	C:\Sparkfatwire\Shared\lock
MarketingStudio	true
MarketingStudioCurrent Version	0.0.0
MSSampleAssetsBF	true
MSSampleAssetsBFCurrent Version	0.0.0
MSSampleSiteGE	true
ObjPubDir	C:\Sparkfatwire\Shared\objpubdir
PLATFORMVENDOR	BEA
PubKeyDir	C:\Sparkfatwire\Shared\pubkeys
ReInit	FALSE
SaveSearchDir	C:\Sparkfatwire\Shared\savedsearch
SBCompanyLaunchpad	true
sCgiPath	/spark/
SparkSampleData	true
SparkSamplePortlets	true
SparkSampleSite	true
SSUserName	SatelliteServer
SSUserPassword	675286278df204915f228fd298c09388
TempObjectsDir	C:\Sparkfatwire\Shared\tempobjects
ThumbnailDir	C:\Sparkfatwire\Shared\thumbnails
TransactVad	yes
TSStoreNumber	2
upgradelog	C:/Sparkfatwire/Shared/Upgrade/upgrade-log-Variables.Date.log

Properties in `omii.ini`: **User Defined** Tab *(continued)*

Property	Value
`VerifyAccountPassword`	`675286278df204915f228fd298c09388`
`VerifySSPassword`	`675286278df204915f228fd298c09388`
`VisitorURLPath`	`C:\Sparkfatwire\Shared\visurl\`
`WCMSampleAssets`	`true`
`WCMSampleAssetsCurrent Version`	`0.0.0`
`WeblogicVersion`	`8`
`WorkflowDir`	`C:\Sparkfatwire\Shared\workflow`
`XmlTempArea`	`C:/Sparkfatwire/xmltemp`

omproduct.ini

The omproduct.ini file holds properties that define which FatWire products, sample sites, and sample asset types are installed with Content Server.

This section presents the properties in the omproduct.ini file, by the tabs where the properties are organized to form a functional group.

The following tab is displayed for the omproduct.ini file:

- User Defined Tab

Caution. Do **not** change the values of any properties on this tab.

omproduct.ini: User Defined Tab

Properties in `omproduct.ini`: **User Defined** Tab

Property	Value
Product22	SparkSampleData
Product21	SparkSampleSite
Product20	CCSampleSiteBF
DateInstalled25	Mon Apr 18 14:43:48 EDT 2005
DateInstalled24	Mon Apr 18 14:43:48 EDT 2005
DateInstalled23	Mon Apr 18 14:43:48 EDT 2005
DateInstalled22	Mon Apr 18 14:43:48 EDT 2005
DateInstalled21	Mon Apr 18 14:43:48 EDT 2005
DateInstalled20	Mon Apr 18 14:43:48 EDT 2005
Product19	CCSampleSiteHW
Product18	MSSampleAssetsBF
Product17	DMSampleAssets
Product16	WCMSampleAssets
Product15	CatCSampleAssets
Product14	CCSampleAssetsBF
Product13	CCSampleAssetsHW
Product12	CCSampleAssets
Product11	SBCompanyLaunchpad
Product10	SparkSamplePortlets
DateInstalled19	Mon Apr 18 14:43:48 EDT 2005
DateInstalled18	Mon Apr 18 14:43:48 EDT 2005
DateInstalled17	Mon Apr 18 14:43:48 EDT 2005
DateInstalled16	Mon Apr 18 14:43:48 EDT 2005
DateInstalled15	Mon Apr 18 14:43:48 EDT 2005
DateInstalled14	Mon Apr 18 14:43:48 EDT 2005
DateInstalled13	Mon Apr 18 14:43:48 EDT 2005
DateInstalled12	Mon Apr 18 14:43:48 EDT 2005

Properties in `omproduct.ini`: **User Defined** Tab *(continued)*

Property	Value
Dependencies9	
DateInstalled11	Mon Apr 18 14:43:48 EDT 2005
Dependencies8	CatalogCentre
DateInstalled10	Mon Apr 18 14:43:48 EDT 2005
Dependencies7	ContentServer
SubComponents9	
Dependencies6	CatalogCentre
SubComponents8	
Dependencies5	ContentCentre
SubComponents7	
Dependencies4	ContentServer
SubComponents6	
Version25	D
Dependencies3	Content Server XML Bridge
SubComponents5	
Version24	D
Dependencies2	
SubComponents4	
Version9	1.0.0
Version23	D
Dependencies1	
SubComponents3	
Version8	D
Version22	D
Dependencies0	
SubComponents2	
Product9	Development
Version7	D

Properties in `omproduct.ini`: **User Defined** Tab *(continued)*

Property	Value
Dependencies25	ContentServer
Version21	D
SubComponents1	
Product8	CommerceConnector
Version6	D
Dependencies24	CCSampleAssets,CCSampleAssetsBF,CatCSampleAssets,MSSampleAssetsBF,AnalysisConnector
Version20	D
SubComponents0	Content Server XML Bridge,Content Server XML Bridge Sample,CSPortal,CSDeployDebug
Product7	AnalysisConnector
Version5	D
Dependencies23	CatCSampleAssets,CommerceConnector
Product6	MarketingStudio
Version4	D
Dependencies22	SparkSampleSite
Product5	CatalogCentre
Version3	6.2.0
Dependencies21	DMSampleAssets,WCMSampleAssets
Product4	ContentCentre
Version2	6.2.0
Dependencies20	CCSampleAssets,CCSampleAssetsBF
Product3	Content Server XML Bridge Sample
Version1	6.2.0
Product2	Content Server XML Bridge
Version0	6.2.0
Product1	CSPortal
Product0	ContentServer

Properties in `omproduct.ini`: **User Defined** Tab *(continued)*

Property	Value
SubComponents25	CCSampleAssetsHW,SparkSamplePortlets,CCSampleSiteHW,WCMSampleAssets,SBCompanyLaunchpad,MarketingStudio,SBCorporateIntranet,CCSampleAssetsBF,ACMSSampleSiteBF,SBFirstSite,CCSampleSiteBF,MSSampleAssetsBF,DMSampleAssets,SparkSampleSite,MSSampleSiteGE,ContentCentre,CatCSampleSiteGE,CatalogCentre,AnalysisConnector,Development,CCSampleAssets,MSSampleSiteBF,CatCSampleAssets,SBDeptIntranet,SparkSampleData,CommerceConnector
SubComponents24	
SubComponents23	
SubComponents22	
SubComponents21	
SubComponents20	
Version19	D
Version18	D
Version17	D
Version16	D
Dependencies19	CCSampleAssetsHW
Version15	D
Dependencies18	Development,MarketingStudio
Version14	D
Dependencies17	Development,CatalogCentre
Version13	D
Dependencies16	Development,CatalogCentre
Version12	D
Dependencies15	Development,CatalogCentre
Version11	1.0.0
Dependencies14	Development,ContentCentre
Version10	D
Dependencies13	Development,ContentCentre

Properties in `omproduct.ini`: **User Defined** Tab *(continued)*

Property	Value
Dependencies12	Development,ContentCentre
Dependencies11	Development,CatalogCentre
Dependencies10	
SubComponents19	
SubComponents18	
SubComponents17	
SubComponents16	
SubComponents15	
SubComponents14	
SubComponents13	
SubComponents12	
SubComponents11	
SubComponents10	
DateInstalled9	Mon Apr 18 14:43:48 EDT 2005
DateInstalled8	Mon Apr 18 14:43:48 EDT 2005
DateInstalled7	Mon Apr 18 14:43:48 EDT 2005
DateInstalled6	Mon Apr 18 14:43:48 EDT 2005
DateInstalled5	Mon Apr 18 14:43:48 EDT 2005
DateInstalled4	Mon Apr 18 14:43:48 EDT 2005
DateInstalled3	Mon Apr 18 14:43:48 EDT 2005
DateInstalled2	Mon Apr 18 14:43:48 EDT 2005
DateInstalled1	Mon Apr 18 14:43:48 EDT 2005
DateInstalled0	Mon Apr 18 14:43:48 EDT 2005
Product25	Content Server Applications
Product24	ACMSSampleSiteBF
Product23	MSSampleSiteGE

satellite.properties

The `satellite.properties` file is present on each Content Server system because each Content Server system has a Satellite servlet running on it. (When the Satellite servlet is running in the same virtual machine as the ContentServer servlet, it is said to be "co-resident.")

The location of `satellite.properties` is on each server that hosts a Satellite Server application. The function of `satellite.properties` is to configure the Satellite servlet that it controls.

This section presents the properties in the `satellite.properties` file, by the tab where the properties are organized to form a functional group.

Tabs in the `satellite.properties` file are the following, in alphabetical order:

- Caching Tab
- Configuration Tab
- Remote Host Tab
- Sessions Tab
- User Defined Tab

satellite.properties: Caching Tab

The **Caching** tab holds the Satellite Server cache settings.

The `file_size` property can significantly influence performance. To optimize performance, try to maximize the amount of memory caching. However, be sure that you do not exceed the host's memory capacity.

If you have lots of memory or a relatively small web site, FatWire recommends caching everything to memory by setting a large value. However, in calculating whether your entire web site can fit in memory, remember that expired web pages stay in memory until explicitly removed or until the cache cleaning thread removes them. Be sure to consider this fact when you set the value of the `cache_check_interval` property.

Properties in `satellite.properties:` **Caching** Tab

Property	Description
`cache_check_interval`	Controls the frequency of the cache cleaner thread. Expired objects are not pruned from cache when they expire. They are pruned either when they are requested and found invalid, or when a cache cleaner thread explicitly prunes them. In minutes, specify the pause between executions of the cache cleaner thread.
`cache_folder`	Specifies the location of disk based cache data. If left blank, cached data will be stored in the context's temp folder. Default value: blank
`cache_max`	Specifies the maximum number of objects to maintain in the cache. Objects are removed from the cache if the size specified is exceeded; an LRU method is used to manage cache size limits.

Properties in `satellite.properties`: **Caching** Tab *(continued)*

Property	Description
`expiration`	Specifies expiration information (in the form of a `COM.FutureTense.Util.TimePattern` string) of all cached objects that do not have this information specified elsewhere. The expiration information for an object can be specified in the `cachecontrol` attribute on the `satellite.page` (and related) tags. For pages, expiration information can also be specified in the Site Catalog's `sscacheinfo` column. For binary objects, the default value for the cachecontrol attribute is specified in the `futuretense.ini` file. Note that the outermost wrapper page of any request cannot specify an override, so this property is the only place where it can be controlled. Default value: `5:0:0 */*/*` This means that everything in the Satellite Server cache expires every day at 5:00 a.m. The format is as follows: *`hours:minutes:seconds daysOfWeek/daysOfMonth/months`* Possible values: • *`hours`*: 0 through 23, where 0 is midnight • *`minutes`*: 0 through 59 • *`seconds`*: 0 through 59 • *`days of week`*: 0 through 6, where 0 is Sunday • *`days of month`*: 1 through 31 • *`months`*: 1 through 12 Other possible values: • `never`, which means the page can expire only if the cache is full and it is the least recently used page • `immediate`, which means to never cache the page
`file_size`	Specifies the size (in kilobytes) of objects that can be cached to disk. Smaller objects are retained in memory. This value should be adjusted for system RAM, disk speed, etc.

satellite.properties: Configuration Tab

The **Configuration** tab holds the properties that configure the Satellite servlet.

Properties in `satellite.properties:` **Configuration** Tab

Property	Description
`blocktimeout`	Specifies the number of seconds a request will wait when another thread is in the process of requesting the same data from the host. Waiting helps reduce load on the host server when the cache is empty at the expense of individual user response time. Default value: `45` A value of -1 means wait until the previous thread returns. A value of 0 means never wait. This value must be tuned based on the host performance, average request size, and network latency. It is safe to use a large number or -1.
`password`	Specifies the password that the Satellite engine will require for special functions like engine restart or cache reset. Be sure to change the username and password from the defaults.
`readtimeout`	Specifies the socket read timeout in seconds, after which a read terminates with an error. A value of 0 leaves the timeout to the Java runtime environment. A value of `3` sets a 3 second wait time.
`servlet`	Specifies the URL pattern used to identify the Satellite Server servlet. Satellite Server will rewrite links and forms to use this URL pattern if pages are properly designed. Default value: `Satellite`
`username`	Specifies the username that the Satellite engine will require for special functions like engine restart or cache reset. Be sure to change the username and password from the defaults.

satellite.properties: Remote Host Tab

The **Remote Host** tab holds properties that define the communications rules between Satellite Server and Content Server. These properties are documented here in alphabetical order:

Properties in `satellite.properties:` **Remote Host** Tab

Property	Description
`bservice`	This value is the servlet path for the Blob Server servlet. It is used to tell Satellite Server where to go to resolve `satellite.blob` tags. Typical values include `/NASApp/cs/BlobServer` for iPlanet and `/servlet/BlobServer` for servlet runners.
`host`	The name of the remote host system running Content Server that the Satellite engine is caching requests for. This is required and there is no default.
`port`	The port number for communicating with the Content Server host. Default value: `80`
`protocol`	The communication protocol between the Satellite Server host and the Content Server host. (Generally `http://` or `https://`). Note that setting the protocol to `https://` will not, in itself, ensure secure communications. You will still need to get a certificate.
`service`	This value is the servlet path for the Content Server servlet. It is used to tell Satellite Server where to go to resolve `satellite.page` tags. Typical values include `/NASApp/cs/ContentServer` for iPlanet and `/servlet/ContentServer` for servlet runners.

satellite.properties: Sessions Tab

The **Sessions** tab holds properties that provide information about how the Satellite servlet should interpret a user’s browser session.

Properties in `satellite.properties`: **Sessions** Tab

Property	Description
`cookieprefix`	Satellite Server maintains a session between itself and the remote host on behalf of the client. Satellite Server needs to know the name of the session ID cookie the application server uses so that it can be properly tracked. Enter the possible session cookie name prefixes here, separated by a semicolon. If left blank, a default set will be used.
`path.to.futuretense.ini`	When the Satellite Server and Content Server servlets are running in the same web application, they should share the user's session. For this to occur, the path to the `futuretense.ini` file needs to be specified so that Satellite Server can load configuration information. If Satellite Server is running in a different web application than Content Server, this property must be left blank; if Satellite Server is running in the same web application as Content Server, set this to the fully-qualified path to the `futuretense.ini` file.
`sessionid.cookie.prefix`	Users can now specify the prefix that is prepended to the session id cookie. The session id cookie is the session id cookie for the host (i.e., Content Server). Satellite Server needs to pass the session id cookie to the client in order to maintain a session between Content Server and the client. The cookie must be renamed, so it does not conflict with the session cookie that Satellite Server itself uses. The configurable prefix allows users who know the name of the session id cookie to construct the full cookie name. This can be used in a servlet filter or other mechanism to support custom functionality.

Properties in `satellite.properties`: **Sessions** Tab *(continued)*

Property	Description
`sharesession`	Specifies whether the ContentServer servlet and the Satellite servlet share the user session. If Satellite Server is running remotely, set this to `false`; if Satellite Server and ContentServer are co-resident, set this property to `true`. If this property is not set appropriately, user-specific information may be inconsistent between pages.

satellite.properties: Compatibility Tab

Properties in `satellite.properties`: **Compatibility** Tab

Property	Description
`formaction`	The Satellite servlet converts Content Server URLs that you GET or POST to into Satellite URLs. This property specifies what string to replace in the Content Server URL to create a Satellite URL. This value is case sensitive. Effective in Satellite Server 6, use the new `satellite.form` tag for all forms.
`newformaction`	Specifies the replacement string in URLs to be GET'ed or POST'ed to the locally mapped servlet. This value is case sensitive. Effective in Satellite Server 6, use the new `satellite.form` tag for all forms.
`globally_replace_content server`	If this property is set to `true`, Satellite Server will parse through all processable pages returned from Content Server and replace all instances of the string described by the `formaction` property with the string described by the `newformaction` property. It will also replace any occurrence of `ContentServer` with the string described by the `servlet` property. Effective in Satellite Server 6, use `satellite.link` or `RENDER.GETPAGEURL` for links and `satellite.form` for forms. If this is not possible, set this property to `true`. Default value: `false`

satellite.properties: User Defined Tab

Properties in `satellite.properties:` **User Defined** Tab

Property	Description
`appserverlink`	Value: `45`
`servlet-path`	Value: `/spark/`

ServletRequest.properties

The `ServletRequest.properties` file contains properties that specify configurations for certain types of requests (for example, a portal request or a Satellite Server request).

Tabs in `ServletRequest.properties` are the following, in alphabetical order:

- Request Encoding Tab
- Request Threshold Tab
- URI Assembler Tab
- User Defined Tab

ServletRequest.properties: Request Encoding Tab

Properties in `ServletRequest.properties`: **Request Encoding** Tab

Property	Description
`cs.charset`	The name of the optional parameter that defines the character encoding of the input.
`cs.contenttype`	The default content type string used when streaming text. For example: For UTF-8: `text/html; charset=UTF-8` for UTF-8 For Latin1: `text/html; charset=iso-8859-1` Default value: `text/html`
`cs.contenttype.UTF-8`	The preferred content type string used when decoding incoming http parameters. This property is designed for use with Japanese language installations, where the user intends to override shift_jis with a special encoding. For example: In Japanese environments with encoding such as Cp943C, set the value to: `cs.contenttype.shift_jis=Cp943C` Default value: set by `cs.contenttype`

ServletRequest.properties: Request Threshold Tab

Properties in `ServletRequest.properties`: **Request Threshold** Tab

Property	Description
`cs.disksize`	Defines the maximum size of a binary blob in a page request that will be held in memory. Anything exceeding this size will be saved in a temporary file until it is needed.

ServletRequest.properties: URI Assembler Tab

Properties in `ServletRequest.properties`: **Request Encoding** Tab

Property	Description

`path.BlobServer`	The servlet context path for the Blob Server servlet.
	Typical values:
	`/cs/BlobServer`
	`/servlet/BlobServer`

`path.CacheServer`	The servlet context path for the Cache Server servlet.
	Typical values:
	`/cs/CacheServer`
	`/servlet/CacheServer`

`path.CatalogManager`	The servlet context path for the Catalog Manager servlet.
	Typical values:
	`/cs/CatalogManager`
	`/servlet/CatalogManager`

`path.ContentServer`	The servlet context path for the Content Server servlet.
	Typical values:
	`/cs/ContentServer`
	`/servlet/ContentServer`

`path.CookieServer`	The servlet context path for the Cookie Server servlet.
	Typical values:
	`/cs/CookieServer`
	`/servlet/CookieServer`

`path.DispatchManager`	The servlet context path for the Dispatch Manager servlet.
	Typical values:
	`/cs/DispatchManager`
	`/servlet/DispatchManager`

`path.PageDispatchServer`	The servlet context path for the Page Dispatch Server servlet.
	Typical values:
	`/cs/PageDispatchServer`
	`/servlet/PageDispatchServer`

Properties in `ServletRequest.properties`: **Request Encoding** Tab *(continued)*

Property	Description

`path.SatelliteServer`	The servlet context path for the Satellite Server servlet on the host that will be most often accessed. Forced Satellite Server URIs will use this path as the servlet context path.
	Typical values:
	`/cs/Satellite`
	`/servlet/Satellite`

`path.SeedDispatchServer`	The servlet context path for the Seed Dispatch Server servlet.
	Typical values:
	`/cs/SeedDispatchServer`
	`/servlet/SeedDispatchServer`

`path.SyncSeedDispatch Server`	The servlet context path for the Sync Seed Dispatch Server servlet.
	Typical values:
	`/cs/SyncSeedDispatchServer`
	`/servlet/SyncSeedDispatchServer`

`path.TreeManager`	The servlet context path for the Tree Manager servlet.
	Typical values:
	`/cs/TreeManager`
	`/servlet/TreeManager`

`uri.assembler.1.classname`	Specifies the classname for the default URI assembler to be used by this instance of Content Server. Users may override this value and specify a different assembler that conforms to the `com.fatwire.cs.core.uri.Assembler` interface. If the assembler specified by this class is unable to decode a URI, then Content Server will attempt to use the next ranked assembler to decode the URI. This process will continue until the URI is decoded.
`uri.assembler.1.shortform`	Specifies the short form name for the corresponding URI assembler. The short form is the name passed into `getURI` methods to identify which assembler to use; it is a nickname for the assembler.

Properties in `ServletRequest.properties`: **Request Encoding** Tab *(continued)*

Property	Description
`uri.assembler.2.classname`	Specifies the classname for the second URI assembler to be used by this instance of Content Server. Users may override this value and specify a different assembler that conforms to the `com.fatwire.cs.core.uri.Assembler` interface. If the assembler specified by this class is unable to decode a URI, then Content Server will attempt to use the next ranked assembler to decode the URI. This process will continue until the URI is decoded.
`uri.assembler.2.shortform`	Specifies the short form name for the corresponding URI assembler. The short form is the name passed into `getURI` methods to identify which assembler to use; it is a nickname for the assembler.
`uri.assembler.3.classname`	Specifies the classname for the third URI assembler to be used by this instance of Content Server. Users may override this value and specify another assembler that conforms to the com.fatwire.cs.core.uri.Assembler interface. If the assembler specified by this class is unable to decode a URI, then Content Server will attempt to use the next ranked assembler to decode the URI. This process will continue until the URI is decoded.
`uri.assembler.3.shortform`	Specifies the short form name for the corresponding URI assembler. The short form is the name passed into `getURI` methods to identify which assembler to use; it is a nickname for the assembler.

ServletRequest.properties: User Defined Tab

Properties in `ServletRequest.properties`: **User Defined** Tab

Property	Description
Note: This tab, by default, has no properties.	

WL6.ini

The WL6.ini file contains properties that define the installation options that are chosen for Content Server during its installation on WebLogic.

This section presents the properties in the WL6.ini file, by the tabs where the properties are organized to form a functional group.

The following tab is displayed for the WL6.ini file:

- User Defined Tab

Caution. Do **not** change the values of any properties on this tab.

WL6Props.ini: User Defined Tab

Properties in WL6Props.ini: **User Defined** Tab

Property	Value
Version	8
WebAppName	spark
WLApplicationDir	C:\bea\user_projects\applications\SparkApp\applications\spark
DomainName	portalDomain
RunningDefaultWebApp	Variables.bRunningDefaultWebApp
WLDomainDir	C:\bea\user_projects\applications\SparkApp
ContextRoot	/spark/
ServerName	Variables.CSInstallWLServerName
PoolName	Variables.CSInstallPoolName

xmles.ini

The `xmles.ini` file configures the CS-Bridge XML module that is installed as part of the core Content Server product.

This section presents the properties in the `xmles.ini` file, by the tab where the properties are organized to form a functional group.

Tabs in the `xmles.ini` file are the following, in alphabetical order:

- General Properties Tab
- Parsing Tab
- Inbound Handlers Tab
- User Defined Tab

xmles.ini: General Properties Tab

Properties in `xmles.ini`: **General Properties** Tab

Property	Description
`wc.icUploadDir`	Specifies the path to the base directory where all CS-Bridge XML documents entered in the InBound, OutBound and DTD catalogs are stored. Default value: `c:/FutureTense`.

xmles.ini: Parsing Tab

Properties in `xmles.ini`: **Parsing** Tab

Property	Description
`wc.enableCacheRet`	Specifies whether or not the DTD should be returned from the cache if the remote system is down. Default value: `true` Set to `false` if you want the default parser behavior instead.
`wc.validate`	Specifies whether an incoming document must specify a grammar. When set to `false`, the incoming document must be well-formed XML only (no grammar is required). Default value: `false`

xmles.ini: Inbound Handlers Tab

Properties in `xmles.ini`: **Inbound Handlers** Tab

Property	Description
`numHandlers`	Specifies the number of inbound handlers to be configured. This value also specifies how many `InboundHandler` properties appear on this tab (the **Inbound Handlers** tab). If you set this value to 2, there are two additional properties on the tab, one for each handler, named `InboundHander0` and `InboundHandler1`. Possible values: an integer greater than 0
`InboundHandler0, InboundHanlder1, …`	The class that implements the inbound handler. Handlers are invoked in order, starting with InboundHandler0. Default value: `com.openmarket.xmles.handlers.HTTPPostHandler.`

xmles.ini: User Defined Tab

Properties in `xmles.ini`: **User Defined** Tab

Property	Description
Note: This tab, by default, has no properties.	

CS-Direct Property Files

CS-Direct installs one property file: `futuretense_xcel.ini`.

Note: CS-Direct also inserts custom properties on to the "User Defined" tab in the `futuretense.ini` file, as described in "futuretense_xcel.ini: User Defined Tab" on page 153.

futuretense_xcel.ini

The futuretense_xcel.ini file is the only property file for CS-Direct. This section presents the properties in the futuretense_xcel.ini file, by the tab where the properties are organized to form a functional group.

Tabs in futuretense_xcel.ini are the following, in alphabetical order:

- Asset Default Tab
- Authorization Tab
- Debug Tab
- Directories Tab
- Element Override Tab
- KeyView Tab
- Preference Tab
- Publishing Tab
- User Management Tab
- User Defined Tab

futuretense_xcel.ini: Asset Default Tab

The **Asset Default** tab holds properties that are used to define certain default details about asset types, such as cache information, default ACLs, and whether eWebEditPro is present.

These properties are documented here in alphabetical order:

Properties in `futuretense_xcel.ini`: **Asset Default** Tab

Property	Description
`xcelerate.asset.shareToAllAllowed`	Specifies whether the assets can be shared to all sites. Possible values: `true/false`
`xcelerate.asset.sizeofnamefield`	Specifies the length of the **Name** field for basic and flex asset types. For 4.0 or later releases this value is set to 64. For installations that have been upgraded from releases earlier than 4.0, this value is usually set to 32.
`xcelerate.body.length`	Specifies the number of characters that are stored for the **Body** field in the `Article` table in the `urlbody` column. Data entered in the **Body** field for an article asset (a sample site asset type) is written to the `urlbody` column. Because this is a URL column, that data is actually stored as a file outside of the Content Server database. However, the first *`n`* number of characters, where *`n`* equals the value specified for this property, is also stored in the `body` column so that you can search for text in the body of an article asset with the search feature in the Content Server interface. Default value: `1000` Maximum values: `256` for Windows NT or Windows 2000; `2000` for UNIX. If this property is missing or is not set, CS-Direct uses the value from the `cc.maxvarcharsize` property in the `futuretense.ini` file instead.
`xcelerate.defaultacl`	Specifies an ACL that is automatically assigned to page entries in the `SiteCatalog` table when they are created by the creation of SiteEntry or Template assets. Default value: `blank`

Properties in futuretense_xcel.ini: **Asset Default** Tab *(continued)*

Property	Description
xcelerate.defaultbase	Specifies the default base of defdir for assets.
xcelerate.defaultcscacheinfo	Specifies the default cacheinfo that is used for Site Catalog entries created with SiteEntry and Template assets. Default value: true
xcelerate.defaultsscacheinfo	Specifies the default value that is set for the sscacheinfo column for page entries in the SiteCatalog table when they are created (for either Template or SiteEntry assets). Default value: *
xcelerate.defaultpagecriteria	Specifies the default value that is set for the pagecriteria column for page entries in the SiteCatalog table when they are created by creating a Template asset. Default values: c, cid, p, rendermode, site, context For definitions of these variables and for more information about page criteria variables in general, see the *Content Server Developer's Guide*.
xcelerate.defaultcsstatus	Specifies the default value that is set for the csstatus column for page entries in the SiteCatalog table when they are created (for either Template or SiteEntry assets). Do **not** change this property Default value: live
xcelerate.defaultpagecriteriaSiteEntry	Specifies the page criteria variables that can be set by default for SiteEntry assets. While you can add variables to this list, do **not** delete any of the default values. Default values: rendermode, site, seid, Sitepfx
xcelerate.ewebeditpro	**Deprecated**. If you have purchased the eWebEditPro HTML editor and your developers have designed asset types that use it, this property specifies the location of the ewebeditpro.js file.

Properties in `futuretense_xcel.ini`: **Asset Default** Tab *(continued)*

Property	Description
`xcelerate.MaxLinks`	Specified the number of links that could be included in a linkset, an old asset type that is no longer used.

futuretense_xcel.ini: Authorization Tab

Properties in `futuretense_xcel.ini`: **Authorization** Tab

Property	Description
`xcelerate.authorize functions`	Contains a comma-separated list of functions for which permissions can be generated on an asset. If the value is empty, all possible functions will be displayed. Additional system-defined functions can be added to the list of default functions. Default value: `inspect, preview, checkout, copy, edit, delete, rollback, share, approve, build`
`xcelerate.deny.abstain fromvoting`	Contains a comma-separated list of roles that are **not** allowed to abstain from voting when assigned assets as part of the workflow process. Default value: blank
`xcelerate.deny.approve`	Contains a comma-separated list of roles that are **not** allowed to approve assets for publishing. Default value: blank
`xcelerate.deny.authorize`	Contains a comma-separated list of roles that are **not** allowed to authorize privileges on assets. Default value: blank
`xcelerate.deny.build`	Contains a comma-separated list of roles that are **not** allowed to build Collection assets. Default value: blank
`xcelerate.deny.checkout`	Contains a comma-separated list of roles that are **not** allowed to checkout assets explicitly from the revision tracking system. Default value: blank
`xcelerate.deny.copy`	Contains a comma-separated list of roles that are **not** allowed to copy assets. Default value: blank
`xcelerate.deny.delegate`	Contains a comma-separated list of roles that are **not** allowed to delegate assigned assets to other participants in the workflow. Default value: blank

Properties in `futuretense_xcel.ini`: **Authorization** Tab *(continued)*

Property	Description
`xcelerate.deny.delete`	Contains a comma-separated list of roles that are **not** allowed to delete assets. Default value: blank
`xcelerate.deny.edit`	Contains a comma-separated list of roles that are **not** allowed to edit assets. Default value: blank
`xcelerate.deny.inspect`	Contains a comma-separated list of roles that are **not** allowed to inspect assets. Default value: blank
`xcelerate.deny.placepage`	Contains a comma-separated list of roles that are **not** allowed to place assets in the SitePlan tree. Default value: blank
`xcelerate.deny.preview`	Contains a comma-separated list of roles that are **not** allowed to preview assets with their templates. Default value: blank
`xcelerate.deny.removefromgroup`	Contains a comma-separated list of roles that are **not** allowed to remove assets from a workflow group. Default value: blank
`xcelerate.deny.removefromworkflow`	Contains a comma-separated list of roles that are **not** allowed to remove assets from workflow. Default value: blank
`xcelerate.deny.rollback`	Contains a comma-separated list of roles that are **not** allowed to roll back assets to a previous version. Default value: blank
`xcelerate.deny.setExportData`	Contains a comma-separated list of roles that are **not** allowed to set export to disk (static publishing) starting point. These users may still be allowed to approve and publish assets if some other users set the starting point. Default value: blank

Properties in `futuretense_xcel.ini`: **Authorization** Tab *(continued)*

Property	Description
`xcelerate.deny.setnestedworkflow`	Contains a comma-separated list of roles that are **not** allowed to set nested workflow. Default value: blank
`xcelerate.deny.setparticipants`	Contains a comma-separated list of roles that are **not** allowed to set participants for workflow. Default value: blank
`xcelerate.deny.setprocessdeadline`	Contains a comma-separated list of roles that are **not** allowed to set workflow process deadlines. Default value: blank
`xcelerate.deny.setstepdeadline`	Contains a comma-separated list of roles that are **not** allowed to **set a deadline** on a workflow **step**. Default value: blank
`xcelerate.deny.share`	Contains a comma-separated list of roles that are **not** allowed to **share** assets with other sites (other than the site the asset was originally created in). Default value: blank
`xcelerate.deny.showparticipants`	Contains a comma-separated list of roles that are **not** allowed to **see** the **participants** that are set for a workflow. Default value: blank
`xcelerate.deny.showstatus`	Contains a comma-separated list of roles that are **not** allowed to **see** the **status** screen for assets. The **status** option shows up in **more …** drop-down box of the inspect window. The screen shows workflow, publishing, and revision tracking information about the asset that is being inspected. If the user belongs to one of the roles that is being denied the privilege, the option to view the status screen will not show. Default value: blank
`xcelerate.deny.showversion`	Contains a comma-separated list of roles that are **not** allowed to see the list of versions for an asset. Note: User can view the list of versions for an asset that is explicitly or implicitly checked out. The inspect form will **not** display a **Show Versions** button if the user belongs to one of the roles that is denied the privilege. Default value: blank

Properties in `futuretense_xcel.ini`: **Authorization** Tab *(continued)*

Property	Description
`xcelerate.grant.abstainfromvoting`	Contains a comma-separated list of roles that are allowed to abstain from voting when assigned assets as part of the workflow process. Default value: blank
`xcelerate.grant.approve`	Contains a comma-separated list of roles that are allowed to approve assets for publishing. Default value: blank
`xcelerate.grant.authorize`	Contains a comma-separated list of roles that are allowed to authorize privileges on assets. Default values: `SparkAdmin`, `GeneralAdmin`, `WorkflowAdmin`, `SiteAdmin`
`xcelerate.grant.build`	Contains a comma-separated list of roles that are allowed to build Collection assets. Default value: blank
`xcelerate.grant.checkout`	Contains a comma-separated list of roles that are allowed to checkout assets explicitly from the revision tracking system. Default value: blank
`xcelerate.grant.copy`	Contains a comma-separated list of roles that are allowed to copy assets. Default value: blank
`xcelerate.grant.delegate`	Contains a comma-separated list of roles that are allowed to delegate assigned assets to other participants in the workflow. Default value: blank
`xcelerate.grant.delete`	Contains a comma-separated list of roles that are allowed to delete assets. Default value: blank
`xcelerate.grant.edit`	Contains a comma-separated list of roles that are allowed to edit assets. Default value: blank
`xcelerate.grant.inspect`	Contains a comma-separated list of roles that are allowed to inspect assets. Default value: blank

Properties in `futuretense_xcel.ini`: **Authorization** Tab *(continued)*

Property	Description
`xcelerate.grant.placepage`	Contains a comma-separated list of roles that are allowed to place assets in the SitePlan tree. Default value: blank
`xcelerate.grant.preview`	Contains a comma-separated list of roles that are **not** allowed to preview assets with their templates. Default value: blank
`xcelerate.grant.removefromgroup`	Contains a comma-separated list of roles that are allowed to remove assets from a workflow group. Default value: blank
`xcelerate.grant.removefromworkflow`	Contains a comma-separated list of roles that are allowed to remove assets from workflow. Default value: blank
`xcelerate.grant.rollback`	Contains a comma-separated list of roles that are allowed to roll back assets to a previous version. Default value: blank
`xcelerate.grant.setExportData`	Contains a comma-separated list of roles that are allowed to set export to disk (static publishing) starting point. Default value: blank
`xcelerate.grant.setnestedworkflow`	Contains a comma-separated list of roles that are allowed to set nested workflow. Default value: blank
`xcelerate.grant.setparticipants`	Contains a comma-separated list of roles that are allowed to set participants for workflow. Default value: blank
`xcelerate.grant.setprocessdeadline`	Contains a comma-separated list of roles that are allowed to set workflow process deadlines. Default value: blank
`xcelerate.grant.setstepdeadline`	Contains a comma-separated list of roles that are **not** allowed to set a deadline on a workflow step. Default value: blank

Properties in `futuretense_xcel.ini`: **Authorization** Tab *(continued)*

Property	Description
`xcelerate.grant.share`	Contains a comma-separated list of roles that are allowed to **share** assets with other sites (other than the site the asset was originally created in). Default value: blank
`xcelerate.grant.showparticipants`	Contains a comma-separated list of roles that are allowed to **see** the **participants** that are set for a workflow. Default value: blank
`xcelerate.grant.showstatus`	Contains a comma-separated list of roles that are allowed to **see** the **status** screen for assets. The **status** option shows up in the **more …** drop-down box of the inspect window. The screen shows workflow, publishing, and revision tracking information about the asset that is being inspected. If the user belongs to one of the roles that is being granted the privilege, the option to view the status screen will show. Default value: blank
`xcelerate.grant.showversion`	Contains a comma-separated list of roles that are allowed to see the list of versions for an asset. Note: The user can view the list of versions for an asset that is explicitly or implicitly checked out. The inspect form will display a **Show Versions** button if the user belongs to one of the roles that is granted the privilege. Default value: blank

futuretense_xcel.ini: Debug Tab

The **Debug** tab holds properties that enable the various CS-Direct debugging utilities. These properties are documented here in alphabetical order.

Properties in `futuretense_xcel.ini`: **Debug** Tab

Property	Description
`am.debug`	Specifies whether AssetMaker logs debugging information about the asset types it makes. When set to `on`, information about the creation of asset types is written to the `futuretense.txt` file. Default value: `off`
`asset.debug`	Specifies whether Desktop and the Export Assets to XML publishing method log debugging information when they manipulate assets. When set to `on`, information about the manipulation of assets is written to the `futuretense.txt` file. Default value: `off`

futuretense_xcel.ini: Directories Tab

The **Directories** tab holds properties that enable the various CS-Direct directories, such as utilities. These properties are documented here in alphabetical order.

Properties in `futuretense_xcel.ini`: **Directories** Tab

Property	Description
`xcelerate. localllanguagedir`	Specifes the directory path of the CS-Desktop client installation. This value is set by the CS-Direct installation. Default value: `C:/FutureTense/futuretense_cs/xcelerate/Client/ClientInstall/CSDesktop`
`xcelerate.lockdir`	Specifies the directory path (including the final slash) to the directory where CS-Direct stores information about the locks that lock data during database operations. If this CS system is installed on a cluster, this directory must have write permissions for and be accessible to all cluster members. This value is set by the CS-Direct installation. Default value: `c:/FutureTense/lock/`
`xcelerate.objpubdir`	Specifies the path to a directory in cluster-shared file space (including the terminating slash character) in which objects that get published are stored temporarily. This value is set by the CS-Direct installation. Default value: `C:/FutureTense/objpubdir/`
`xcelerate.pubkeydir`	Specifies the directory where the publishing system writes information about the items that have been published to the various target systems. This value is set by the CS-Direct installation. Default value: `c:/FutureTense/pubkeys/`
`xcelerate.saveSearchdir`	Specifies the defdir (default storage directory) for the `SaveSearch` table. This table has a URL column that holds the saved searches on a development or management system. This value is set by the CS-Direct installation. Default value: `c:/FutureTense/Storage/SaveSearch`

Properties in `futuretense_xcel.ini`: **Directories** Tab *(continued)*

Property	Description
`xcelerate.sePath`	Specifies the directory where search indexes are stored when you are using a search engine on your CS system. If you change this value, be sure to specify a directory that exists. (This property does not create the directory for you.) Default values: • Windows NT or Windows 2000: `c:/FutureTense/sedb` • Solaris or AIX: `/export/home/FutureTense/sedb`
`xcelerate.tempobjectsdir`	Specifies the defdir (default storage directory) for the `TempObjects` table, a CS-Direct table that stores information about objects that are uploaded or in the process of being created until they are either saved or canceled. This value is set by the CS-Direct installation. Default value: `c:/FutureTense/tempobjectsdir/`
`xcelerate.thumbnaildir`	Specifies the directory where the template asset will store thumbnail images associated with template variant thumbnails. This value is set by the CS-Direct installation. Default value: `C:/FutureTense/thumbnaildir/`
`xcelerate.workflowdir`	Specifies the name of the directory that holds files related to workflow processes. This value is set by the CS-Direct installation. Default value: `C:/FutureTense/workflowdir` If you change the value from the default, be sure that directory exists.

futuretense_xcel.ini: Element Override Tab

The **Element Override** tab holds properties that you can use to help customize the user interface. These properties are documented here in alphabetical order:

Properties in `futuretense_xcel.ini`: **Element Override** Tab

Property	Description
`xcelelem.manageuserpub`	Defines the element that CS-Direct uses to manage the roles that users fulfill for your Content Server sites. Default value: `Openmarket/Xcelerate/Actions/Security/AccessUserPublication`
`xcelelem.publishfactors`	The name of the element used to provide additional publishing control factors. May be empty. Default value: `Openmarket/Xcelerate/Actions/Publish/OverrideFactor`
`xcelelem.publishoptions`	This property allows customization of a portion of the common options area of the publishing form. It is used by the Action/Publish/PublishOptions element. If this is defined, it names an element to call which will lay out the publishing options section of all the publishing forms. Default value: `empty`
`xcelelem.setpubid`	**Deprecated** Specifies the name of the element that sets the pubid session variable when visitors first come to your site via a dynamic URL. It is run once per visitor session. Default value: `OpenMarket/Xcelerate/Actions/Publish/SetPubid`

futuretense_xcel.ini: KeyView Tab

The **KeyView** tab holds the properties that provide information to the Verity KeyView utility, a component used by the Desktop and DocLink applications.

These properties are documented here in alphabetical order:

Properties in `futuretense_xcel.ini`: **Key View** Tab

Property	Description
`keyview.apidir`	Set at installation, this value sets the path to the shared libraries for Verity KeyView. Do **not** change the value of this property.
`keyview.imgdir`	This value sets the path to the image files that Verity KeyView creates for Desktop. This path is not set by the installer. To set this path (so that you will have the ability to see these image files) do the following: **1.** Define a webroot somewhere that points to a directory in a shared filesystem that is accessible from all cluster members. (Defining this webroot is not required unless you need this feature). **2.** Set `keyview.imgurl` to be the relative url path (starting with `/`) of the webroot you defined, or the absolute url (starting with `http`) of the webroot. **3.** Set `keyview.imgdir` to the directory path that the webroot points to.
`keyview.imgurl`	This value sets the prefix that is added to the SRC attributes of IMG tags that are generated for image files by Verity KeyView for Desktop. This value is not set by the installer. If you want the ability to see the image files that Verity KeyView creates for Desktop, see the instructions provided in the description of `keyview.imgdir`.
`xcelerate.transformpath`	Set at installation, this value sets the path to the directory where Desktop temporarily stores files that are transformed by VerityKeyView. Do **not** change the value of this property.

futuretense_xcel.ini: Preference Tab

The **Preference** tab holds properties that you use to configure the search feature, the tree, and the character set used on your system.

These properties are documented here in alphabetical order.

Properties in `futuretense_xcel.ini`: **Preference** Tab

Property	Description
`xcelerate.charset`	Specifies the character set that CS-Direct uses to communicate with the server. Default value: `UTF-8` Changing this property from the default value may limit the characters that are supported.
`xcelerate.` `emailnotification`	**Deprecated** Specifies whether the CS-Direct workflow e-mail notification feature is enabled. When set to `true`, the workflow system sends e-mail messages to users when assets are assigned to them through a workflow process. Default value: `false` See also, the "Email" section in the *Content Server Administrator's Guide*.
`xcelerate.` `restrictSiteTree`	Specifies whether users other than admin users can toggle the tree on in the Content Server interface when it is configured to be toggled off by default (that is, the `xcelerate.showSiteTree` property is set to `false`). Set to `true` to enable only users with the xceladmin ACL to be able to toggle the tree back on. Default value: `false`. For more information about this feature, see the *Content Server Administrator's Guide*.
`xcelerate.seLimit`	Specifies the maximum number of query results that should be returned from any internally conducted search engine query. Default value: `10000`

Properties in futuretense_xcel.ini: **Preference** Tab *(continued)*

Property	Description
xcelerate.showSiteTree	Specifies whether the tree is displayed by default when any user logs in to the Content Server interface. Set to false if you want the tree to be toggled off by default. Default value: true For more information about this feature, see the *Content Server Administrator's Guide*.
xcelerate.treeMaxNodes	Specifies the number of items that are displayed under a node in the tree in the Content Server interface. When a node has more than this number of items, Content Server prompts the user to enter search criteria to reduce the number. Default value: 100 For more information about this feature, see the *Content Server Administrator's Guide*.
xcelerate.treeType	Specifies the kind of tree that is used in the Content Server interface. Legal values are OMTree or a value that specifies a customized replacement tree. Default value: OMTree Do **not** change this property without first consulting FatWire Professional Services or FatWire Customer Support.
xcelerate.usese	Specifies whether CS-Direct should use an installed search engine. Legal values: true or false Default value: false

futuretense_xcel.ini: Publishing Tab

The **Publishing** tab holds the properties that provide information to the CS-Direct publishing system, which also uses the properties in the `futuretense.ini` file (Export/Mirror tab). For descriptions of the properties, see "Export/Mirror Tab" on page 68.

Properties in `futuretense_xcel.ini`: **Publishing** Tab

Property	Description
`xcelerate.batchhost`	Specifies the host name and port number of the server on which the publish process will run.
`xcelerate.batchmode`	Defines the batch publishing mode. Legal values: • `single`—batch host is a dedicated IP address. • `multiple`—batch host is a cluster IP address. Default value: `single`
`xcelerate.batchpass`	Specifies the password for the batch user. Default value: `xceladmin` Be sure to change this value after you create the batch user for this CS system. For information, see the *Content Server Administrator's Guide*.
`xcelerate.batchuser`	The CS-Direct publishing system runs as a background process and you must configure a batch user account for the publishing system to use. This property specifies the user name of the batch user. Default value: `admin` Be sure to change this value after you create the batch user for this CS system. For information, see the *Content Server Administrator's Guide*.
`xcelerate.blobref`	The name of the class that manages the publish references for blobs. The default is provided here for reference only: `com.openmarket.xcelerate.publish.BlobRef`. Do **not** change the value of this property. For information about published references, see the *Content Server Administrator's Guide*.

Properties in `futuretense_xcel.ini`: **Publishing** Tab *(continued)*

Property	Description
`xcelerate.bulkapprovechunk`	Specifies the number of assets to approve at the same time, in the same batch or "chunk," when someone uses the **Approve Multiple Assets** feature in the Content Server interface. The feature approves all the assets that are selected for approval in batches and the number of assets in each batch is set by this property. Default value: `500` For information about the Approve Multiple Assets feature, see the Co*ntent Server Administrator's Guide*.
`xcelerate.donotregenerate`	Specifies whether cached pages are regenerated after a publishing session. Possible values: • blank, that is, no value—means that all the pages in the cache that were affected by the publish session are refreshed. • `unknowndeps`—means that cached pages that were generated from an element that used a `RENDER.UNKNOWNDEPS` tag are not refreshed. • `*` (asterisk)—means none of the pages in the cache are refreshed. In other words, the affected pages are refreshed only when a visitor requests the page. Default value: blank Do not change the value of this property without first consulting FatWire support personnel.
`xcelerate.exportmaxfilename`	This is the maximum length of any file name generated during export publishing. If you are running Windows NTFS, you may want to set this to a low value to make it possible to re-export, delete or rename files created by the export-to-disk publish process. NTFS has an upper limit of 255 characters for any path name.

Properties in `futuretense_xcel.ini`: **Publishing** Tab *(continued)*

Property	Description
`xcelerate.mirrorini`	If you have had the element identified by the `xcelerate.remotecall` property modified in such a way that it needs information from additional property files other than `futuretense.ini`, this property specifies the names of all the property files that are needed. Default value: `futuretense.ini` Do **not** modify this value to include additional property files without consulting FatWire Professional Services or FatWire Customer Support.
`xcelerate.pageref`	Specifies the name of the class that manages publish references for pages. The default is provided here for reference only: `com.openmarket.xcelerate.publish.PageRef.` Do **not** change the value of this property.
`xcelerate.presaveelt`	Specifies the name of the element called on the mirror target during publish after the assets' primary rows have been mirrored, but prior to the deserialization and `asset.save` of the flex and complex assets. Use `PresaveElement` for standard mirror publish.
`xcelerate.pubabortelt`	Specifies the name of the element called on the mirror target if the publish fails. Default values: `PubAbortElement` for standard mirror publish.
`xcelerate.pubcleanupelt`	Specifies the name of the element that the publishing system uses during the cleanup phase of a mirror publish operation. Default value: `PubCleanupElement` Do **not** modify this value without consulting FatWire Professional Services or FatWire Customer Support.

Properties in `futuretense_xcel.ini`: **Publishing** Tab *(continued)*

Property	Description
`xcelerate.publishallasset types`	Specifies whether to publish all asset types on mirror publish. Possible values: `true` | If set to `true` all asset types will be published. If not set to true, only asset types of the assets involved in the publish and their dependent asset types will be published. Note: If you want to set this property to `true`, you have to make sure that all asset types that exist on the source server are exist on the publish destination.
`xcelerate.publishinvalidate`	Specifies whether an asset is marked as changed on the destination system when an asset is published. If it is marked as changed, it must be approved on that system before it can be published from that system to a new destination. Possible values: `true` | `false` Default value: `true` Because having the publishing system take the time to mark the assets as changed on the destination adds time to the publishing session, typically you leave this property set to `true` on development and management systems but change it to `false` on delivery systems.
`xcelerate.pubsetupelt`	Specifies the name of the element that the publishing system uses during the setup phase of a mirror publish. Default value: `PubSetupElement` Do **not** modify this value without consulting fatwire Professional Services or fatwire Customer Support.

Properties in `futuretense_xcel.ini`: **Publishing** Tab *(continued)*

Property	Description
`xcelerate.remotecall`	Specifies the pagename that is invoked on the target system during a mirror publishing session. Default value: `Openmarket/Xcelerate/PrologActions/Publish/Mirror1/RemoteCall` Do **not** modify this element or change the value of this property without assistance from FatWire Professional Services or FatWire Customer Support.
`xcelerate.templatedefault`	The name of the template to use if a template cannot be found to render an asset type. Default value: `Openmarket/TemplateDefault`

futuretense_xcel.ini: xcelerate Tab

The **xcelerate** tab holds the properties that specify such things as default administrative settings, whether the InSite Editor is enabled, whether workflow configuration and search engine are being used, whether LDAP is being used, and so on.

These properties are documented here in alphabetical order:

Properties in `futuretense_xcel.ini`: **xcelerate** Tab

Property	Description
`xcelerate.adminacl`	Specifies the ACL that users must be assigned so they can access administrator functions (that is, any of the functions that appear on the Admin tab in the Content Server interface). Default value: `xceladmin` If you change the value of this property to a different ACL, be sure to assign that ACL to all the tables that currently have the xceladmin ACL assigned to them.
`xcelerate.adminrole`	Specifies an ACL that is set for all administrative tables during installation. Default value: `xceladmin`. Do not change the value of this property after your system is installed.
`xcelerate.base`	Specifies the top-level (base) directory of the CS-Direct elements. During installation, the installer might need to edit this value to indicate where the CS-Direct elements are in your installation. Default values: • Windows NT or Windows 2000: `c:/Fatwire/elements/OpenMarket/Xcelerate` • Solaris or AIX: `/export/home/Fatwire/elements/OpenMarket/Xcelerate` Do not change the value of this property after your system is installed.
`xcelerate.crosssiteassign`	Specifies whether users from more than one site can participate in the same workflow process. Possible values: `true` | `false` Default value: `false`
`xcelerate.defaultlang`	Specifies the default language. Default value: `en_US`
`xcelerate.defaultpreviewurlgenerator`	Set to the preview generator name to generate URLs for preview, if no site-specific one is described.

Properties in `futuretense_xcel.ini`: **xcelerate** Tab *(continued)*

Property	Description
`xcelerate.domain`	Specifies the domain name of the system, not including the server (machine) name. This property is used by applications that have been integrated with Content Server and that have a browser interface.
`xcelerate.editrole`	Specifies an ACL that is set for editorial tables during installation. Default value: `xceleditor` Do not change the value of this property after your system is installed.
`xcelerate.enableinsite`	Enables or disables the InSite Editor for this CS system. A value of `true` enables the InSite Editor. Default value: `false` Do **not** enable the InSite Editor on your Content Server delivery system.
`xcelerate.imageurl`	Specifies the webroot for all image URLs used by the applications.
`xcelerate.previewhost`	One of two properties that enable the preview host feature, this property sets the cgi path to use for the preview host. For information about this feature, see “Maintaining Separate Browser Sessions for Preview” in the *Content Server Administrator’s Guide*. If you provide a value for this property, use the following syntax: • For most application servers, including Sun ONE: `http://`*`servername:port`*`/servlet/` • For iPlanet Application Server (iAS): `http://`*`servername:port`*`/NASApps/cs/`
`xcelerate.previewservlet`	One of two properties that enable the preview host feature, this property specifies which servlet the preview host should use. For information about this feature, see “Maintaining Separate Browser Sessions for Preview” in the *Content Server Administrator’s Guide*. Possible values: `ContentServer` or `Satellite` Default value: `Satellite`
`xcelerate.previewurlpagename`	Set to the name of page to generate URLs for preview. Do **not** change the value of this property.

Properties in `futuretense_xcel.ini`: **xcelerate** Tab *(continued)*

Property	Description
`xcelerate.rolemanagerclass`	Specifies the name of the role manager class. By default, the value of this property is set to the CS-Direct role management system. Default value: `com.openmarket.xcelerate.roles.RoleManager` • If you are using a Sun ONE embedded Identity Server, set this property to the following value, exactly: `com.openmarket.xcelerate.roles.IdentityServerRoleManager` • If you are using a WebLogic embedded LDAP, set this property to the following value, exactly: `com.openmarket.xcelerate.roles.FlatLDAPRoleManager`
`xcelerate.treetabmanagerclass`	The class that implements ITreeTabManager to provide tree tab descriptions for CS-Direct. The default is provided here for reference only: `com.openmarket.xcelerate.treetab.TreeTabManager` Do not change the value of this property.
`xcelerate.usermanagerclass`	The class that implements IUserManager to provide user services for CS-Direct. Default value: `com.openmarket.xcelerate.user.UserManager` • If you want to implement LDAP attribute-mapping for site and role names, set this property to the following value, exactly: `com.openmarket.xcelerate.user.LDAPSchemaUserManager` • If you are using a WebLogic or Sun ONE Application Server embedded LDAP, set this property to the following value, exactly: `com.openmarket.xcelerate.user.FlatLDAPSchemaUserManager`
`xcelerate.workflowengineclass`	The class that implements IWorkflowEngine to provide workflow services for CS-Direct. The default is provided here for reference only: `com.openmarket.xcelerate.workflow.WorkflowEngine` Do not change the value of this property.

futuretense_xcel.ini: User Management Tab

The **User Management** tab holds the properties that specify various user attribute names, which hold different information about the user, such the list the roles that the user has for a publication, the screen name that they user uses, the list of sites the user is enabled for.

Properties in `futuretense_xcel.ini`: **User Management** Tab

Property	Description
`xcelerate.displayablenameattr`	Specifies the name of the user attribute describing the displayable name, if different from the login name.
`xcelerate.emailattr`	Specifies the name of the user attribute that is used to identify a user's e-mail address to your CS system. These attributes are kept in the `SystemUserAttr` table. Default value: `mail`
`xcelerate.localeattr`	Specifies the name of the user attribute that identifies the locale that a user specifies if you have more than one language pack installed on your CS system. Default values: • When there is one language present, the value is blank. • When there is more than one language present, the default is `locale`
`xcelerate.pubrolesattr`	Specifies the name of the user attribute that lists the roles that the user has for publication. This property is only used if `xcelerate.usermanagerclass` is set to `com.openmarket.xcelerate.user.LDAPAttrUserManager` This is combined with the publication id to obtain the attribute name containing the roles the user has for the publication. If value is not set, the UserPublication table is used, or some other LDAP User Manager Plug-in is used.

Properties in `futuretense_xcel.ini`: **User Management** Tab *(continued)*

Property	Description
`xcelerate.sitenameattr`	Specifies the naming attribute of the site entries. This property is only used if `xcelerate.usermanagerclass` is set to `com.openmarket.xcelerate.user.LDAPSchemaUserManager`. By default, this value is blank, which means that information about a user's roles is stored in the `UserPublication` table, or some other LDAP User Manager plug-in is used. If there is a value specified for this property, the `xcelerate.usermanagerclass`, and `xcelerate.sitesroot` properties must also be configured correctly.
`xcelerate.sitesattr`	Specifies the name of the user attribute describing which publications the user has roles for. This property is used only if `xcelerate.usermanagerclass` is set to `com.openmarket.xcelerate.user.LDAPAttrUserManager`. If there is a value for this property, it is combined with the value of the `pubid` column and the value for the `xclerate.pubroleattr` property to determine a user's access rights in the Content Server interface. By default, this value is blank, which means that information about a user's roles is stored in the `UserPublication` table, or some other LDAP User Manager plug-in is used.
`xcelerate.sitesroot`	Specifies the root node (dn) under which sites are located. This property is used only if `xcelerate.usermanagerclass` is set to `com.openmarket.xcelerate.user.LDAPSchemaUserManager`. By default, this value is blank, which means that information about a user's roles is stored in the `UserPublication` table, or some other LDAP User Manager plug-in is used. If there is a value specified for this property, the `xcelerate.usermanagerclass` and `xcelerate.sitenameattr` properties must also be configured correctly.

futuretense_xcel.ini: User Defined Tab

Properties in `futuretense_xcel.ini`: **User Defined** Tab

Property	Description
`xcelerate.fckeditor.basepath`	FCKEditor basepath; location of the FCKEditor files on the Content Server system. **Default value:** `/<URI>/FCKeditor/`

CS-Direct Advantage Property Files

Property files installed by CS-Direct Advantage are the following:

- assetframework.ini
- catalog.ini
- gator.ini
- visitor.ini
- visitor.ini

assetframework.ini

The assetframework.ini file holds properties that determine where files that hold information about flex asset history and publishing are stored.

This file has a single tab, named "User Defined."

assetframework.ini: User Defined Tab

Properties assetframework.ini: **User Defined** Tab

Property	Description
afk.historydata	Specifies the directory that holds history data. Default value: c:/futuretense/history/
afk.publishdata	Specifies the directory that holds publish data. Default value: c:/futuretense/publish/

catalog.ini

The catalog.ini file holds properties that CS-Direct Advantage uses to configure the shopping cart.

Tabs in catalog.ini are the following, in alphabetical order:

- Catalog Tab
- User Defined Tab

catalog.ini: Catalog Tab

Properties in `catalog.ini`: **Catalog** Tab

Property	Description
`mwb.cartsetdir`	Specifies the directory path (including the terminating slash character) where cartset data files are stored. This value is set by the installation. It points to the `/gator/cartset` directory in the installation directory. Do **not** change the value of this property.

catalog.ini: User Defined Tab

Properties in `batch.ini`: **User Defined** Tab

Property	Description
Note: This tab, by default, holds no properties.	

gator.ini

The gator.ini property file is the main property file for CS-Direct Advantage. This section presents the properties in the gator.ini file, by the tab where the properties are organized to form a functional group.

Tabs in the gator.ini file are the following, in alphabetical order:

- Gator Tab
- User Defined Tab

gator.ini: Gator Tab

Properties in `gator.ini`: **Gator** Tab

Property	Description
`cc.attrDisplayStyle`	The field to display (name or description) to describe attributes on a FlexAsset or FlexGroup ContentForm or ContentDetails screen. Possible values: `name` or `description` Default value: `name`
`cc.attributeinheritance`	A boolean that Gator uses to determine whether attributes should be inherited from parent to child. Default value: `true`
`cc.extrapath`	A boolean that Gator uses to determine if blobs should have extra path information prepended to avoid directory capacity issues on Unix. Default value: `true`
`cc.fullconstraint`	A boolean that Gator uses to determine if nested queries should include data from the outer query. Possible values: `true` | `false` Default value: `true`
`cc.money`	The sql for defining a field that will contain monetary values. Choose the default or contact your database administrator. Default value: NUMERIC(20,3) Do **not** change this value without consulting your database administrator.
`cc.querystyle`	A boolean that Gator uses to determine which basic form of assetset query to generate. Possible values: `subquery`, `join`, or `intersect` Default value: `subquery` Note that setting the value of this property to `intersect` functions only if your database can support intersection queries.
`cc.string`	The sql for defining a field that will contain string values. Default value: `SEARCHVARCHAR` Do **not** change this value without consulting your database administrator.

Properties in gator.ini: **Gator** Tab *(continued)*

Property	Description
cc.textdistinct	A boolean describing whether your database can support DISTINCT on attributes of type TEXT. Default value: no
cc.url	The sql for defining a field that will contain URLs. Default value: VARCHAR(128) Do **not** change this value without consulting your database administrator.
cc.urlattrpath	Specifies the default base path Gator uses for url attribute files. Default value: c:/futuretense/futuretense_cs/urlfiles
cc.useLegacyInputNames	Specifies the boolean that Gator uses to determine the input names of attributes on FlexAsset/Parent forms. Turn this property on if you need to support custom attribute editors that have not been updated to the new format. Default value: false
mwb.assetsetclass	Specifies the name of the class that supplies the services for assetset management. Default value: com.openmarket.gator. assetset.AssetSet Do **not** change the value of this property. The default is provided here for reference only.
mwb.cartclass	The name of the class that supplies the services for cart management. Default value: com.openmarket.catalog.cart.Cart Do **not** change the value of this property. The default is provided here for reference only.
mwb.cartsetclass	Specifies the name of the class that supplies the services for cart set management. Default value: com.openmarket.catalog. cartset.CartSet Do **not** change the value of this property. The default is provided here for reference only.

Properties in `gator.ini`: **Gator** Tab *(continued)*

Property	Description
`mwb.commercecontextclass`	Specifies the name of the class that supplies the services for commerce context. Default value: `com.openmarket.gator.commercecontext.CommerceContext` Do **not** change the value of this property. The default is provided here for reference only.
`mwb.commerceengineclass`	Specifies the name of the class that supplies the services for commerce engine management. Default value: `com.openmarket.cscommerce.txcart.TransactEngine`
`mwb.commerceuserclass`	Specifies the name of the class that supplies the services for commerce user management. Default value: `com.openmarket.catalog.cart.CommerceUser`
`mwb.conservativedependencies`	Specifies the types of dependencies between: • flex assets and flex attributes • flex assets and flex definitions If you want dependencies to be `exact` between flex assets and flex attributes, and between flex assets and flex definitions, then set this property to `true`. For information about exact and exists dependencies, see the *Content Server Administrator's Guide*. Default value: `false`
`mwb.defaultattributes`	Specifies the default attribute asset name for Gator to use when creating assetsets. Default value: blank
`mwb.externalattributes`	Specifies a boolean that Gator uses to determine if forms should allow users to define external attributes. Default value: `true`
`mwb.path`	Specifies the directory where Gator is installed. Be sure to end the directory with a forward slash (/). Default value: `c:/nas21/apps/` Do **not** change the value of this property.

Properties in `gator.ini`: **Gator** Tab *(continued)*

Property	Description
`mwb.promotioncutoff`	Specifies the confidence rating that determines whether a visitor qualifies for a promotion. Legal values: integers between `0` and `100`, inclusive Default value: `50` Do **not** change the value of this property.
`mwb.searchdir`	Specifies the directory where Gator places rich-text indexes. Be sure to end the directory with a forward slash (/). Default value: `c:/futuretense/gator/search/` Do **not** change the value of this property.
`mwb.searchstateclass`	Specifies the name of the class that supplies the services for search state management. Default value: `com.openmarket.gator.searchstate.SearchState` Do **not** change the value of this property. The default is provided here for reference only.
`mwb.segmentcutoff`	The confidence rating that determines whether to include a visitor in a segment. Legal values: integers between `0` and `100`, inclusive. Default value: `50` Do **not** change the value of this property.

gator.ini: User Defined Tab

Properties in `gator.ini`: **User Defined** Tab

Property	Description
Note: This tab, by default, has no properties.	

visitor.ini

The visitor.ini file is installed by CS-Direct Advantage; however, the properties in the file configure Engage rather than CS-Direct Advantage. The properties configure the visitor data collection and other features provided by Engage.

This section presents the properties in the visitor.ini file, by the tab where the properties are organized to form a functional group.

Tabs in the visitor.ini file are the following, in alphabetical order:

- Visitor Data Tab
- User Defined Tab

visitor.ini: Visitor Data Tab

The **Visitor Data** tab holds the main properties in the file.

Properties in `visitor.ini`: **Visitor Data** Tab

Property	Description
`vis.adminrole`	Specifies the ACL that Engage users need in order to work with the visitor attribute, history attribute, history type, and recommendation asset types. Default value: `VisitorAdmin` Do **not** change the value of this property.
`vis.compileclasspath`	Specifies the classpath against which to compile the rules. This value is set during the installation and should not be changed after that point.
`vis.editrole`	Specifies the ACL that two kinds of Engage users need: • Content providers who use the CS system to create segments and promotions. • The visitors to your online site when you are using Engage to gather information about them for segments. Default value: `Visitor` Do **not** change the value of this property.
`vis.genclasspath`	Specifies the directory (including the final slash character) where rules-engine-generated class files for visitor data are stored. This value is set during the installation and should not be changed after that point.
`vis.money`	Specifies the SQL string for defining fields that hold monetary values. Do **not** change the value of this property without consulting your database administrator.
`vis.path`	Specifies the directory that holds the `ruleset.dtd` file, which is usually the installation directory. This value is set during installation. Do **not** change the value of this property.

Properties in `visitor.ini`: **Visitor Data** Tab *(continued)*

Property	Description
`vis.rulesetxmlpath`	Specifies the defdir (default storage directory) for the XML versions of the rule sets. This value is set during installation. Do **not** change the value of this property.
`vis.sessiondata`	Specifies the defdir (default storage directory) for storing visitor session data. This value is set during installation. Do **not** change the value of this property.
`vis.update`	Specifies whether every page access will update the visitor timestamp in the visitor data. Possible values: `trues` | `false` Set to `true` if you want every page access to update the visitor timestamp in the visitor data; `false` otherwise. Default value: `true`
`vis.url`	Specifies the SQL string for defining visitor and history attributes of type URL. Default value: `VARCHAR(128)` Do **not** change the value of this property without consulting your database administrator.
`vis.urlpath`	Specifies the defdir for binary visitor and history attributes. Default value: `/futuretense/visurl/`

visitor.ini: User Defined Tab

Properties in `visitor.ini`: **User Defined** Tab

Property	Description
Note: This tab, by default, has no properties.	

Part 2
Content Server Applications

This part lists the property files that are used by the Content Server applications Engage, and Satellite Server.

This part contains the following sections:

- Engage Property Files
- Satellite Server Property Files

Engage Property Files

Engage installs one property file, named `ms.ini`, which holds only the `ms.enable` property.

ms.ini

Properties in `ms.ini`

Property	Description
`ms.enable`	Set to `true` when Engage is installed and enabled.

.

Note: Engage configuration properties are located in the CS-Direct Advantage property file named `visitor ini`, given on page 155.

Satellite Server Property Files

When you install Satellite Server as a stand-alone application on a remote server, two property files are present on that server:

- `satellite.properties`
- `resin.conf`

For information about the properties in `satellite.properties`, see "satellite.properties" on page 108.

For information about the properties in `resin.conf`, refer to your Resin documentation.

Part 3
Third-Party Libraries

This part contains information about third-party libraries and how Content Server integrates with the libraries.

This part contains the following section:

- HTTP Client Access

HTTP Client Access

This section describes the Apache Commons HttpClient library and how Content Server integrates with this library.

Apache Commons HttpClient

Content Server uses Apache Commons HttpClient as the underlying library for all HTTP access. As of version 3, HttpClient supports the parameters that are posted at:
`http://jakarta.apache.org/commons/httpclient/preference-api.html`

The parameters function as follows:

- HttpClient parameters change the runtime behavior of HttpClient components.
  For example, if you want the `Post` operation to have a timeout that differs from the default, you can call `PostMethod.getParams().setParam("timeout", 1000)` before executing it.

- HttpClient parameters can be hierarchically linked.
  In top-down order, the levels of the hierarchy are: global, client, host, and method. Values that are set for parameters at higher levels are overridden by the values of equivalent parameters at lower levels.

Despite its flexibility, HttpClient has a limitation; that is—parameters can be set only programmatically. No configuration file can be written where parameter values can be specified by users or automatically retrieved by the library. The Content Server integration, however, overcomes this limitation as explained in the next section "Integration with Content Server."

Note

This release of Content Server uses the parameters that are posted at `http://jakarta.apache.org/commons/httpclient/preference-api.html` **at the time of this writing (September 2005)**. The parameters are listed in the table "HttpClient Parameters," on page 176, along with descriptions (duplicated from the site named above). Changes to parameters and their functionality as defined by HttpClient are not automatically supported.

Integration with Content Server

Content Server abstracts HttpClient functionality by allowing Content Server users to create user-configurable property files. After creating the files, users populate them with the required HttpClient parameters (that is, parameters whose values differ from the default values), and place the property files into the classpath. Content Server loads the property files from the classpath and parses the parameters according to a predefined syntax (shown in the table "HttpClient Parameters," on page 176). The HttpAccess API retrieves the parameters and applies them at runtime.

Content Server supports a parameter hierarchy whose levels correspond directly to the levels that are defined in the HttpAccess Java API (provided in *Content Server Java*

Docs). For each level, one or more property files can be created (depending on the implementation) and populated with ***any combination*** of HttpClient parameters. The levels and property file naming conventions are given below:

> **Note**
>
> The property files must be created as text files, outside Content Server's Property Editor. Property file names are case sensitive and must be in lower case throughout.

- **HttpAccess (level 1)**

  Property File: `httpaccess.properties`

  The user specifies parameters and their values in the `httpaccess.properties` file. This file is applied to all HttpAccess instances that are created.

  Overrides: Parameter values at the HttpAccess level are overridden by the values of equivalent parameters at levels 2, 3, and 4 (described below).

- **HostConfig (level 2)**

  Property File: `<protocol>-<hostname>-<port number>.properties`

  The user specifies host-specific parameters in each property file. For example, for a host named `targetserver` accessible at port 7001, the property file would be named `http-targetserver-7001.properties` and would contain HttpClient parameters specific to that host.

  Overrides: Parameter values at the HostConfig level override the values of equivalent parameters at the HttpAccess level.

- **Request (level 3)**

  Property File: `<request type>.properties`
  where `<request type>` takes one of the following values: `post`, `get`, or `login`.

  The user specifies parameters specific to a Request. For example, `post.properties` specifies HttpClient parameters applicable to instances of `post`.

  Overrides: Parameter values at the Request level override the values of equivalent parameters at the HttpAccess and HostConfig levels.

- **Per host, per Request (level 4)**

  Property File: `<request type>-<protocol>-<host name>-<port number>.properties`
  where `<request type>` takes one of the following values: `post`, `get`, or `login`.

  Parameters in this property file function as Request level parameters. However, they apply to a specific host.

Overrides: Parameter values specified at the “Per host, per Request” level override the values of equivalent parameters at the HttpAccess, HostConfig, and Request levels *for that particular host*.

The following example illustrates how an override takes effect from the “Per Host, Per Request” level. In this example, a user defines a property file named `login-http-m2-7002.properties`, where she specifies an `http.connection.timeout` of 100 seconds. The timeout applies strictly to the host machine named `m2` and port 7002. The timeout value overrides all timeout values that might be specified for m2 at higher levels. For all other host machines, the timeout values remain unaffected.

As previously mentioned, Content Server supports all parameters defined by HttpClient in an externally configurable way. Furthermore, Content Server extends HttpClient functionality by enabling users to configure parameters externally and facilitating the specification of parameters at the fourth level (per host, per request).

In addition to all the parameters supported by HttpClient, Content Server's HttpAccess API defines a configuration property `cs.SecureProtocolSocketFactory` in `httpaccess.properties`. This property specifies the protocol socket factory to be used for SSL (Secure Socket Layer) connections. Three implementations are available at `http://jakarta.apache.org/commons/httpclient/sslguide.html`. Note that if you want to use SSL to connect to a host using self-signed certificates, you must configure the following:

```
cs.SecureProtocolSocketFactory=org.apache.commons.httpclient.
  contrib.ssl.EasySSLProtocolSocketFactory
```

Content Server does not provide this `EasySSLProtocolSocketFactory` class. You can obtain this class at `http://jakarta.apache.org/commons/httpclient/sslguide.html`. Make sure to build it differently for Sun and IBM JDKs, as the Apache implementation (at the link directly above) is Sun-specific. Alternatively you can write your own Socket factory implementation based on HttpClient documentation.

Note that there are two levels—connection manager and connection—in the HttpClient hierarchy for which parameters cannot be explicitly set, as the HttpAccess API does not directly support them. However, this does not mean users cannot configure those parameters; the parameters can be specified at a lower or higher corresponding level in the HttpAccess API.

Implementation

How does the Content Server user configure Content Server for http access? The user simply creates property files with appropriate names and places them in the classpath. The infrastructure will retrieve and use them. This seems like a good deal of work, especially given the number of parameters. However, by default, no properties or property files need to be created. All defaults will be used, and HttpClient takes the “best guess” values, which are usually the best settings for the given system. In 95% of the cases, “best guess” values are sufficient and users need not create any property files.

In the rare cases when one needs parameter values other than defaults, the Content Server infrastructure makes it possible to implement them by allowing the user to specify configuration in property files. This gives the user the full range of configuration capabilities that HttpClient itself is built upon.

HTTPClient Parameters and Content Server Properties

The table in this section describes the parameters that are supported by Apache Commons HttpClient in September 2005. Descriptions in the table are duplicated from the following site:

`http://jakarta.apache.org/commons/httpclient/preference-api.html`

Syntax and default values are defined by FatWire, as they are specific to Content Server. Where syntax is straightforward, the "Syntax" field in the table below is left blank.

Note that changes to the parameters and their functionality are not automatically supported. Information in the table below is valid until FatWire issues an update.

In addition to supporting HttpClient parameters, Content Server defines the following property:

Property: `cs.SecureProtocolSocketFactory`

Usage: applicable only to the `httpaccess.properties` file

Description: defines the class used opening SSL Socket connections

Default: empty. The system will use the JSSE-based default implementation of HttpClient. Details are available at: `http://jakarta.apache.org/commons/httpclient/sslguide.html`

HttpClient Parameters

Name	Description
`http.authentication.preemptive`	Defines whether authentication should be attempted preemptively. **Type:** Boolean **Syntax:** **Default:** <undefined>
`http.connection.stalecheck`	Determines whether stale connection check is to be used. Disabling stale connection check may result in slight performance improvement at the risk of getting an I/O error when executing a request over a connection that has been closed at the server side. **Type:** Boolean **Syntax:** **Default:** <undefined>
`http.connection.timeout`	The timeout until a connection is established. A value of zero means the timeout is not used. **Type:** Integer **Syntax:** **Default:** <undefined>

HttpClient Parameters *(continued)*

Name	Description
`http.connection-manager.class`	The default HTTP connection manager class. **Type:** Class **Syntax:** Fully qualified classname **Default:** SimpleHttpConnectionManager class
`http.connection-manager.max-per-host`	Defines the maximum number of connections allowed per host configuration. These values only apply to the number of connections from a particular instance of HttpConnectionManager. This parameter expects a value of type `Map`. The value should map instances of HostConfiguration to Integer s. The default value can be specified using `ANY_HOST_CONFIGURATION`. **Type:** Map **Syntax:** Specify `${<host>;<port>;<protocol>;<max connections>}` **Default:** <undefined>
`http.connection-manager.max-total`	Defines the maximum number of connections allowed overall. This value only applies to the number of connections from a particular instance of HttpConnectionManager. **Type:** Integer **Syntax:** **Default:** <undefined>
`http.connection-manager.timeout`	The timeout in milliseconds used when retrieving an HTTP connection from the HTTP connection manager. **Type:** Long **Syntax:** **Default:** <undefined>

HttpClient Parameters *(continued)*

Name	Description
`http.dateparser.patterns`	Date patterns used for parsing. The patterns are stored in a Collection and must be compatible with SimpleDateFormat. **Type:** Collection **Syntax:** Specify the collection with each element enclosed in `${<element>}`. e.g., `${EEE, dd-MMM-yyyy HH-mm-ss z}${EEE, dd MMM yy HH:mm:ss z}` **Default:** EEE, dd MMM yyyy HH:mm:ss zzz EEEE, dd-MMM-yy HH:mm:ss zzz EEE MMM d HH:mm:ss yyyy EEE, dd-MMM-yyyy HH:mm:ss z EEE, dd-MMM-yyyy HH-mm-ss z EEE, dd MMM yy HH:mm:ss z EEE dd-MMM-yyyy H:mm:ss z EEE dd MMM yyyy HH:mm:ss z EEE dd-MMM-yyyy HH-mm-ss z EEE dd-MMM-yy HH:mm:ss z EEE dd MMM yy HH:mm:ss z EEE,dd-MMM-yy HH:mm:ss z EEE,dd-MMM-yyyy HH:mm:ss z EEE, dd-MM-yyyy HH:mm:ss z
`http.default-headers`	The request headers to be sent per default with each request. This parameter expects a value of type Collection. The collection is expected to contain HTTP headers. **Type:** Collection **Syntax:** Specify each header in `${name=<header name>; value=<header value>}` **Default:** <undefined>
`http.method.multipart.boundary`	The multipart boundary string to use in conjunction with the MultipartRequestEntity. When not set a random value will be generated for each request. **Type:** String **Syntax:** **Default:** <undefined>

HttpClient Parameters *(continued)*

Name	Description
`http.method.response.buffer.warnlimit`	The maximum buffered response size (in bytes) that triggers no warning. Buffered responses exceeding this size will trigger a warning in the log. If not set, the limit is 1 MB. **Type:** Integer **Syntax:** **Default:** <undefined>
`http.method.retry-handler`	The method retry handler used for retrying failed methods. For details see the Exception handling guide. **Type:** HttpMethodRetryHandler **Syntax:** Fully qualified classname **Default:** default implementation
`http.protocol.allow-circular-redirects`	Defines whether circular redirects (redirects to the same location) should be allowed. The HTTP spec is not sufficiently clear whether circular redirects are permitted, therefore optionally they can be enabled. **Type:** Boolean **Syntax:** **Default:** <undefined>
`http.protocol.content-charset`	The charset to be used for encoding content body. **Type:** String **Syntax:** **Default:** ISO-8859-1
`http.protocol.cookie-policy`	The cookie policy to be used for cookie management. **Type:** String **Syntax:** **Default:** CookiePolicy.RFC_2109

HttpClient Parameters *(continued)*

Name	Description
`http.protocol.credential-charset`	The charset to be used when encoding credentials. If not defined then the value of the http.protocol. element-charset should be used. **Type:** String **Syntax:** **Default:** <undefined>
`http.protocol.element-charset`	The charset to be used for encoding/decoding HTTP protocol elements (status line and headers). **Type:** String **Syntax:** **Default:** US-ASCII
`http.protocol.expect-continue`	Activates “Expect: 100-Continue” handshake for the entity enclosing methods. The “Expect: 100-Continue” handshake allows a client that is sending a request message with a request body to determine if the origin server is willing to accept the request (based on the request headers) before the client sends the request body. The use of the “Expect: 100-continue” handshake can result in noticeable performance improvement for entity enclosing requests (such as POST and PUT) that require the target server's authentication. “Expect: 100-continue” handshake should be used with caution, as it may cause problems with HTTP servers and proxies that do not support HTTP/1.1 protocol. **Type:** Boolean **Syntax:** **Default:** <undefined>
`http.protocol.head-body-timeout`	Sets period of time in milliseconds to wait for a content body sent in response to HEAD response from a non-compliant server. If the parameter is not set or set to `-1` non-compliant response body check is disabled. **Type:** Integer **Syntax:** **Default:** <undefined>

HttpClient Parameters *(continued)*

Name	Description
`http.protocol.max-redirects`	Defines the maximum number of redirects to be followed. The limit on number of redirects is intended to prevent infinite loops. **Type:** Integer **Syntax:** **Default:** <undefined>
`http.protocol.reject-head-body`	Defines whether the content body sent in response to HEAD request should be rejected. **Type:** Boolean **Syntax:** **Default:** <undefined>
`http.protocol.reject-relative-redirect`	Defines whether relative redirects should be rejected. **Type:** Boolean **Syntax:** **Default:** <undefined>
`http.protocol.single-cookie-header`	Defines whether cookies should be put on a single response header. **Type:** Boolean **Syntax:** **Default:** <undefined>
`http.protocol.status-line-garbage-limit`	Defines the maximum number of ignorable lines before we expect a HTTP response's status code. With HTTP/1.1 persistent connections, the problem arises that broken scripts could return a wrong Content-Length (there are more bytes sent than specified). Unfortunately, in some cases, this is not possible after the bad response, but only before the next one. So, HttpClient must be able to skip those surplus lines this way. Set this to 0 to disallow any garbage/empty lines before the status line. To specify no limit, use Integer.MAX_VALUE. **Type:** Integer **Syntax:** **Default:** <undefined>

HttpClient Parameters *(continued)*

Name	Description
`http.protocol.strict-transfer-encoding`	Defines whether responses with an invalid Transfer-Encoding header should be rejected. **Type:** Boolean **Syntax:** **Default:** <undefined>
`http.protocol.unambiguous-statusline`	Defines whether HTTP methods should reject ambiguous HTTP status line. **Type:** Boolean **Syntax:** **Default:** <undefined>
`http.protocol.version`	The HTTP protocol version used per default by the HTTP methods. **Type:** HttpVersion **Syntax:** `<(int)major>.<(int)minor>`; e.g., 1.1 **Default:** HttpVersion_1_1
`http.protocol.warn-extra-input`	Defines HttpClient's behavior when a response provides more bytes than expected (specified with Content-Length header, for example). Such surplus data makes the HTTP connection unreliable for keep-alive requests, as malicious response data (faked headers etc.) can lead to undesired results on the next request using that connection. If this parameter is set to true, any detection of extra input data will generate a warning in the log. **Type:** Boolean **Syntax:** **Default:** <undefined>

HttpClient Parameters *(continued)*

Name	Description
`http.socket.linger`	The linger time (`SO_LINGER`) in seconds. This option disables/enables immediate return from a close() of a TCP Socket. Enabling this option with a non-zero Integer timeout means that a close() will block pending the transmission and acknowledgement of all data written to the peer, at which point the socket is closed gracefully. Value `0` implies that the option is disabled. Value `-1` implies that the JRE default is used. **Type:** Integer **Syntax:** **Default:** <undefined>
`http.socket.receivebuffer`	The value to set on `Socket.setReceiveBufferSize(int)`. This value is a suggestion to the kernel from the application about the size of buffers to use for the data to be received over the socket. **Type:** Integer **Syntax:** **Default:** <undefined>
`http.socket.sendbuffer`	The value to set on `Socket.setSendBufferSize(int)`. This value is a suggestion to the kernel from the application about the size of buffers to use for the data to be sent over the socket. **Type:** Integer **Syntax:** **Default:** <undefined>
`http.socket.timeout`	Sets the socket timeout (`SO_TIMEOUT`) in milliseconds to be used when executing the method. A timeout value of zero is interpreted as an infinite timeout. **Type:** Integer **Syntax:** **Default:** <undefined>

HttpClient Parameters *(continued)*

Name	Description
`http.socket.timeout`	The default socket timeout (`SO_TIMEOUT`) in milliseconds which is the timeout for waiting for data. A timeout value of zero is interpreted as an infinite timeout. This value is used when no socket timeout is set in the HTTP method parameters. **Type:** Integer **Syntax:** **Default:** <undefined>
`http.tcp.nodelay`	Determines whether Nagle's algorithm is to be used. The Nagle's algorithm tries to conserve bandwidth by minimizing the number of segments that are sent. When applications wish to decrease network latency and increase performance, they can disable Nagle's algorithm (by enabling `TCP_NODELAY`). Data will be sent earlier, at the cost of an increase in bandwidth consumption. **Type:** Boolean **Syntax:** **Default:** <undefined>
`http.useragent`	The content of the User-Agent header used by the HTTP methods. **Type:** String **Syntax:** **Default:** Official release name e.g., `Jakarta Commons-HttpClient/3.0`

Index

A

administrators
  property that sets admin ACL 148
AltaVista search engine
  properties 82
application servers
  properties 45
authentication properties 46

B

batch process properties 20
BlobServer
  cache settings 52
  properties 52
  security setting 49

C

cc.textdistinct 161
cluster
  properties 53
Content Server Direct
  properties 125
Content Server Direct Advantage
  properties 156
content tables
  properties 54, 58
CS-Direct Advantage, *See* Content Server Direct Advantage

D

database
  properties 59
deleting
  properties 13
disk cache
  properties 74

E

e-mail
  properties 66
  specifying which user attribute holds address 151
Engage
  properties 169
EvalServer
  properties 52

F

firewall server
  IP address 68
  port number 69
futuretense.txt file
  debug properties that specify what gets written to it 65
  where located 65

H

HTTP

properties 73

I
InSite Editor
property 149

J
Java Server Pages, *See* JSP
JSP
properties 70

K
KeyView properties 140

L
logging, message
properties 92

M
message logging
properties 92

P
performance
ft.filecheck property 79
preferences
properties 141
properties
adding 12
administrator 148
AltaVista 82
application server 45
authentication 46
batch 20
cluster 53
content table 54, 58
CS-Direct Advantage 156
database 59
deleting 13
e-mail 66
Engage 169
HTTP 73
InSite Editor 149
JSP 70
KeyView 140
large text fields 61
message logging 92
preferences 141
publishing 68, 143
resultset caching 76, 86
Satellite Server 80
search engine 82
security 49
setting 10
URL columns 73
Verity 85
visitor data 165
Property Editor
adding properties 12
deleting properties 13
setting properties 10
starting 10
publishing
properties 68, 143

R
resultset caching
properties 76, 86

S
Satellite Server
properties 80
search engines
properties 82
security
properties 49
sources
defined 68

T
targets
defined 68

U
URL columns
properties 73
user management

authentication properties 46

V

Verity search engine
properties 85

visitor data
properties 165

Index of Properties

$$$AppServerType 95
$$$Product0 95
$$$Product1 95
$BASEDIR 95
$HOMEDIR 95
$INSTALLDEST 95
$INSTALLDIR 95
$LOCALHOST 95
$LOGDIR 95
ACMSSampleSiteBF 95
ACStoragePath 95
AdditionalPathToLib 95
afk.historydata 156
afk.publishdata 156
am.debug 136
AnalysisConnector 95
analysisconnector.version 86
AnalysisConnectorCurrentVersion 95
app 90
appserverlink 115
asset.debug 136
av.cjkquery 82
av.defaultindex 82
av.license 82
av.oemkeytype 82
baseDN 34
bCSInstall 95
bDisplayPropEditor 95
bDoAllACProperties 95
bDoAllAFKProperties 95
bDoAllCatCProperties 95
bDoAllCCProperties 95
bDoAllComCProperties 95
bDoAllMSProperties 95
bDoDBInitialization 95
bDoFromAC36xTo40 95
bDoFromAC40To45 95
bDoFromAC45To51 95
bDoFromAC51To61 95
bDoFromAFK36xTo40 95
bDoFromAFK40To45 95
bDoFromAFK45To51 95
bDoFromAFK51To61 95
bDoFromCatC36xTo40 96
bDoFromCatC40To45 96
bDoFromCatC45To51 96
bDoFromCatC51To61 96
bDoFromCC36xTo40 96
bDoFromCC40To45 96
bDoFromCC45To51 96
bDoFromCC51To61 96
bDoFromComC36xTo40 96
bDoFromComC40To45 96
bDoFromComC45To51 96
bDoFromComC51To61 96
bDoFromMS36xTo40 96
bDoFromMS40To45 96
bDoFromMS45To51 96
bDoFromMS51To61 96
bDoUpgradeFromCC36xTo40 96
bDoUpgradeFromCC40To45 96
bDoUpgradeFromCC45To51 96
bDoUpgradeFromCC51To61 96
bJSPResponseWrapper 96
bLDAPIntegration 96
blocktimeout 110
bMandatoryLegacyInstall 96
bRunningManaged 96
bs.bCacheSize 52

bs.bCacheTimeout 52
bs.security 49
bservice 111
bSunIdentityServer 96
bUrlFileRollUp 96
cache_check_interval 108
cache_folder 108
cache_max 108
CartsetDir 96
CatalogCentre 96
catalogcentre.version 86
CatalogCentreCurrentVersion 96
CatCSampleAssets 96
CatCSampleAssetsCurrentVersion 96
cc.AssetTypeCSz 86
cc.attrDisplayStyle 160
cc.attributeinheritance 160
cc.bigint 59
cc.bigtext 59
cc.blob 60
cc.cacheNoSync 53
cc.cacheResults 77
cc.cacheResultsAbs 77
cc.cacheResultsTimeout 78
cc.CategoryCSz 86
cc.char 60
cc.ComparatorsKey 87
cc.contentkey 58
cc.datepicture 60
cc.datetime 60
cc.double 60
cc.ElementCatalogCSz 78
cc.ElementCatalogTimeout 78
cc.extrapath 160
cc.FiltersKey 87
cc.forcelower 61
cc.fullconstraint 160
cc.ignoreTblCase 61
cc.integer 61
cc.maxvarcharsize 61
cc.MimeTypeKey 87
cc.money 160
cc.null 62
cc.numeric 62
cc.PreviewgenKey 87
cc.primary 62
cc.queryablemaxvarcharlength 62
cc.querystyle 160
cc.rename 62
cc.security 49
cc.SiteCatalogCSz 78
cc.SiteCatalogTimeout 78
cc.smallint 63
cc.SourceKey 87
cc.StatusCodeCSz 87
cc.StatusCodeKey 87
cc.string 160
cc.stringpicture 63
cc.SystemACLCSz 78
cc.SystemACLTimeout 79
cc.SystemInfoCSz 79
cc.SystemInfoTimeout 79
cc.SystemPageCacheCSz 75
cc.SystemPageCacheTimeout 75
cc.SystemUsersCSz 79
cc.SystemUsersTimeout 79
cc.textdistinct 161
cc.unique 63
cc.url 161
cc.urlattrpath 161
cc.useLegacyInputNames 161
cc.varchar 63
CCSampleAssets 96
CCSampleAssetsBF 96
CCSampleAssetsBFCurrentVersion 97
CCSampleAssetsCurrentVersion 97
CCSampleAssetsHW 97
CCSampleAssetsHWCurrentVersion 97
CCSampleSiteBF 97
CCSampleSiteHW 97
CCVersionString 97
ClassesComFolder 97
ClassesCrysecFolder 97
ClassesSComFolder 97
className.Attribute 35
className.Attributes 35
className.IDir 35
className.IFactory 35
className.IName 35
className.IUserDir 36
className.JNDIName 36
cleandns 33
cn 31
com.fatwire.logging.cs 23
com.fatwire.logging.cs.auth 23
com.fatwire.logging.cs.blobserver 23
com.fatwire.logging.cs.cache.page 23
com.fatwire.logging.cs.cache.resultset 23

com.fatwire.logging.cs.db 24
com.fatwire.logging.cs.event 24
com.fatwire.logging.cs.export 24
com.fatwire.logging.cs.
install 24
com.fatwire.logging.cs.jsp 24
com.fatwire.logging.cs.
request 25
com.fatwire.logging.cs.
satellite 25
com.fatwire.logging.cs.
satellite.cache 25
com.fatwire.logging.cs.
satellite.host 25
com.fatwire.logging.cs.
satellite.request 25
com.fatwire.logging.cs.
session 25
com.fatwire.logging.cs.sync 25
com.fatwire.logging.cs.time 26
com.fatwire.logging.cs.
visitor.object 26
com.fatwire.logging.cs.
visitor.ruleset 26
com.fatwire.logging.cs.
xcelerate.advantage.
recommendation 26
com.fatwire.logging.cs.
xcelerate.approval 26
com.fatwire.logging.cs.xml 26
CommerceConnector 97
commerceconnector.version 87
CommerceConnectorCurrentVersion 97
component 90
Content Server XML Bridge 97
Content Server XML Bridge Sample 97
ContentCentre 97
contentcentre.version 87
ContentCentreCurrent
Version 97
ContextRoot 121
cookieprefix 112
cs.alwaysusedisk 74
cs.barEqualsSlash 49
cs.charset 117
cs.charset 29
cs.charset 72
cs.contenttype 117
cs.contenttype 29
cs.contenttype 72
cs.contenttype.UTF-8 117
cs.contenttype.UTF-8 29
cs.cookievariables 54
cs.dataindatabase 54
cs.dbconnpicture 63
cs.dbencoding 87
cs.dbtype 64
cs.disksize 117
cs.disksize 29
cs.disksize 72
cs.documentation 72
cs.dsn 64
cs.emailaccount 66
cs.emailauthenticator 66
cs.emailcharset 66
cs.emailcontenttype 66
cs.emailhost 66
cs.emailpassword 67
cs.emailreturnto 67
cs.eventhost 45
cs.freezeCache 74
cs.HTTP_HOST 73
cs.HTTP_PROTOCOL 73
cs.httpvariables 54
cs.IItemList 74
cs.jspclear 70
cs.jsppath 70
cs.jsprefresh 70
cs.jspresponsewrapper 70
cs.jsproot 71
cs.jspwork 71
cs.manageACL 46
cs.manageproperty 46
cs.manageUser 46
cs.manageUserAccess 46
cs.manageUserSystem 47
cs.mirrorhttpversion 68
cs.mirrorpassword 68
cs.mirrorproxyserver 68
cs.mirrorproxyserverport 69
cs.mirrorrowsperpost 69
cs.mirrorthreads 69
cs.mirroruser 69
cs.nocache 75
cs.PastramiEngine 80
cs.pgcachefolder 55
cs.pgCacheTimeout 75
cs.pgexportfolder 69
cs.privpassword 64
cs.privuser 64
cs.recordBlobInventory 75
cs.requestfactory 88
cs.requiresessioncookies 75
cs.satellitehosts 55

cs.satellitepassword 56
cs.satelliteusers 56
cs.searchengine 83
cs.selfmodify 56
cs.session 49
cs.timeout 50
cs.uniqueidpoolsize 50
cs.urlfilerollup 73
cs.wrapper 50
cs.xmlfolder 73
cs.xmlHeader 73
cs.xmlHeaderAutoStream 56
CSAppServerLibBin 97
CSAppServerLibBinNT 97
CSAppsVersion 97
CSConnectPrefix 97
CSConnectString 97
CSInstallAccountName 97
CSInstallAccountPassword 97
CSInstallAdminDomainName 97
CSInstallAppServerPath 97
CSInstallAppServerRootPath 97
CSInstallAppServerType 98
CSInstallDatabaseType 98
CSInstallDBDSN 98
CSInstallDBPrivUser 98
CSInstallDirectory 98
CSInstallPlatformType 98
CSInstallSharedDirectory 98
CSInstallSpark 98
CSInstallType 98
CSInstallWebRootPath 98
CSInstallWebServer
  Address 98
CSInstallWebServerPort 98
CSInstallWLDomainPath 98
CSInstallWLWebAppName 98
CSPortal 98
Date 98
DateInstalled0 106
DateInstalled1 106
DateInstalled10 103
DateInstalled11 103
DateInstalled12 102
DateInstalled13 102
DateInstalled14 102
DateInstalled15 102
DateInstalled16 102
DateInstalled17 102
DateInstalled18 102
DateInstalled19 102
DateInstalled2 106
DateInstalled20 102
DateInstalled21 102
DateInstalled22 102
DateInstalled23 102
DateInstalled24 102
DateInstalled25 102
DateInstalled3 106
DateInstalled4 106
DateInstalled5 106
DateInstalled6 106
DateInstalled7 106
DateInstalled8 106
DateInstalled9 106
debug 20
DefaultBase 98
defaultGroupAttrs 41
defaultPeopleAttrs 41
defaultReaderACLs 41
Dependencies0 103
Dependencies1 103
Dependencies10 106
Dependencies11 106
Dependencies12 106
Dependencies13 105
Dependencies14 105
Dependencies15 105
Dependencies16 105
Dependencies17 105
Dependencies18 105
Dependencies19 105
Dependencies2 103
Dependencies20 104
Dependencies21 104
Dependencies22 104
Dependencies23 104
Dependencies24 104
Dependencies25 104
Dependencies3 103
Dependencies4 103
Dependencies5 103
Dependencies6 103
Dependencies7 103
Dependencies8 103
Dependencies9 103
Development 98
DMSampleAssets 98
DMSampleAssetsCurrent
  Version 98
domain 90
DomainName 121
expiration 109
FatWire Corporation Installation 98

file_size 109
formaction 114
ft.approot 57
ft.catalogmanager 57
ft.cgipath 45
ft.contentserver 57
ft.debugport 65
ft.filecheck 79
ft.servletoutputstream 88
ft.suppressPasswordNames 65
ft.suppressPasswords 65
ft.sync 53
ft.treemanager 57
ft.usedisksync 53
ft.version 50
GatorCCUrlPath 98
GatorHistoryDir 98
GatorPublishDir 98
GatorSearchDir 98
GatorSessionData 98
GatorVisitorGenclasspath 98
GatorVisitorRuleXML 98
GatorVisPath 98
globally_replace_content
  server 114
GrantList 98
groupparent 34
host 111
http.authentication.
  preemptive 176
http.connection.
  stalecheck 176
http.connection.
  timeout 176
http.connection-manager.class 177
http.connection-manager.max-per-host 177
http.connection-manager.max-total 177
http.connection-manager.timeout 177
http.dateparser.
  patterns 178
http.default-headers 178
http.method.
  multipart.boundary 178
http.method.
  response.buffer.
  warnlimit 179
http.method.retry-handler 179
http.protocol.allow-circular-redirects 179
http.protocol.
  content-charset 179
http.protocol.
  cookie-policy 179
http.protocol.
  credential-charset 180
http.protocol.
  element-charset 180
http.protocol.
  expect-continue 180
http.protocol.
  head-body-timeout 180
http.protocol.max-redirects 181
http.protocol.
  reject-head-body 181
http.protocol.reject-relative-redirect 181
http.protocol.
  single-cookie-header 181
http.protocol.status-line-garbage-limit 181
http.protocol.
  strict-transfer-encoding 182
http.protocol.
  unambiguous-statusline 182
http.protocol.
  version 182
http.protocol.
  warn-extra-input 182
http.socket.linger 183
http.socket.
  receivebuffer 183
http.socket.
  sendbuffer 183
http.socket.timeout 183
http.socket.timeout 184
http.tcp.nodelay 184
http.useragent 184
image.time 88
InboundHandler0,
InboundHanlder1, … 123
InstallACbDisplayProp
  Editor 99
InstallCatCbDisplayProp
  Editor 99
InstallCCbDisplayProp
  Editor 99
IsPrimaryClusterMember 99
java.naming.factory.
  initial 37
java.naming.security.
  authentication 37
jndi.baseURL 37
jndi.connectAsUser 37
jndi.custom 37
jndi.login 38
jndi.password 38
jndi.poolConnections 38

jndi.poolsize 38
keyview.apidir 140
keyview.imgdir 140
keyview.imgurl 140
LDAPHost 99
LDAPJNDIPassword 99
LDAPPort 99
LinuxAddOnPathToLib 99
LockDir 99
log.Directory.messages 93
log.filterLevel 92
log.Logger.messages 93
log.transformer.messages 93
log.wmentconnector.messages 93
logging.file 27
logging.format 27
logging.maxlogsize 27
logging.per-client-log 27
logging.roll 27
logging.timestamp 28
loginattribute 31
MarketingStudio 99
marketingstudio.version 88
MarketingStudioCurrent
Version 99
memberof 32
ms.enable 169
MSSampleAssetsBF 99
MSSampleAssetsBFCurrent
Version 99
MSSampleSiteGE 99
mwb.assetsetclass 161
mwb.cartclass 161
mwb.cartsetclass 161
mwb.cartsetdir 158
mwb.commercecontextclass 162
mwb.commerceengineclass 162
mwb.commerceuserclass 162
mwb.conservative
dependencies 162
mwb.defaultattributes 162
mwb.externalattributes 162
mwb.path 162
mwb.promotioncutoff 163
mwb.searchdir 163
mwb.searchstateclass 163
mwb.segmentcutoff 163
newformaction 114
ntlogin.DefaultACL 47
ntlogin.DefaultReaderACL 47
ntlogin.DefaultReaderID 47
ntlogin.DefaultReaderPW 47
ntlogin.LogFile 48
ntlogin.Logging 48
numHandlers 123
objectclassGroup 41
objectclassPerson 42
ObjPubDir 99
org.apache.commons.logging.Log 22
page.time 88
password 110
password 31
password 90
path.BlobServer 118
path.CacheServer 118
path.CatalogManager 118
path.ContentServer 118
path.CookieServer 118
path.DispatchManager 118
path.PageDispatchServer 118
path.SatelliteServer 119
path.SeedDispatchServer 119
path.SyncSeedDispatch
Server 119
path.to.futuretense.ini 112
path.TreeManager 119
peopleparent 34
PLATFORMVENDOR 99
PoolName 121
port 111
Product0 104
Product1 104
Product10 102
Product11 102
Product12 102
Product13 102
Product14 102
Product15 102
Product16 102
Product17 102
Product18 102
Product19 102
Product2 104
Product20 102
Product21 102
Product22 102
Product23 106
Product24 106
Product25 106
Product3 104
Product4 104
Product5 104
Product6 104
Product7 104

Product8 104
Product9 103
protocol 111
PubKeyDir 99
readtimeout 110
ReInit 99
request.folder 20
requiredGroupAttrs 42
requiredPeopleAttrs 42
RunningDefaultWebApp 121
satellite.blob.cache
control.default 80
satellite.page.cache
control.default 81
SaveSearchDir 99
SBCompanyLaunchpad 99
sCgiPath 99
search.returnLimit 43
search.scope 43
search.timeoutVal 43
searcheng.apidebug 83
searcheng.debug 83
searcheng.enginedebug 83
searcheng.keeptemps 83
searcheng.querydebug 83
searcheng.regdebug 83
searcheng.usedebugse 83
secure.CatalogManager 50
secure.DebugServer 51
secure.TreeManager 51
security.checkpagelets 57
security.class 20
ServerName 121
service 111
servlet 110
servlet-path 115
sessionid.cookie.prefix 112
sharesession 113
singlesignon 48
soap.binaryRowsType 88
soap.iList 88
soap.likeConstraint 88
soap.listRowsType 88
soap.nestedConstraint 88
soap.rangeConstraint 88
soap.richTextConstraint 89
soap.searchstate 89
soap.standardConstraint 89
soap.stringRowsType 89
soap.stringVarsType 89
soap.urlRowsType 89
soap.URLType 89
SparkSampleData 99
SparkSamplePortlets 99
SparkSampleSite 99
SSUserName 99
SSUserPassword 99
SubComponents0 104
SubComponents1 104
SubComponents10 106
SubComponents11 106
SubComponents12 106
SubComponents13 106
SubComponents14 106
SubComponents15 106
SubComponents16 106
SubComponents17 106
SubComponents18 106
SubComponents19 106
SubComponents2 103
SubComponents20 105
SubComponents21 105
SubComponents22 105
SubComponents23 105
SubComponents24 105
SubComponents25 105
SubComponents3 103
SubComponents4 103
SubComponents5 103
SubComponents6 103
SubComponents7 103
SubComponents8 103
SubComponents9 103
syntax.beginquote 39
syntax.beginquote2 39
syntax.custom 38
syntax.direction 39
syntax.endquote 39
syntax.endquote2 39
syntax.escape 39
syntax.ignorecase 40
syntax.separator 40
syntax.separatorava 40
syntax.separatortypeval 40
syntax.trimblanks 40
TempObjectsDir 99
thread.count 19
thread.growcache 19
thread.idle 19
thread.wait 19
ThumbnailDir 99
TransactVad 99
TSStoreNumber 99
uniquemember 31

upgradelog 99
uri.assembler.1.classname 119
uri.assembler.1.shortform 119
uri.assembler.2.classname 120
uri.assembler.2.shortform 120
uri.assembler.3.classname 120
uri.assembler.3.shortform 120
url 90
user 90
username 110
username 32
VerifyAccountPassword 100
VerifySSPassword 100
verity.charset 83
verity.debug 83
verity.defaultindex 83
verity.defaultparser 84
verity.indexinginterval 84
verity.knowledgebasepath 84
verity.locale 84
verity.organization 84
verity.path 84
verity.signature 84
verity.syncindexing 84
verity.templatepath 85
Version 121
version 90
Version0 104
Version1 104
Version10 105
Version11 105
Version12 105
Version13 105
Version14 105
Version15 105
Version16 105
Version17 105
Version18 105
Version19 105
Version2 104
Version20 104
Version21 104
Version22 103
Version23 103
Version24 103
Version25 103
Version3 104
Version4 104
Version5 104
Version6 104
Version7 103
Version8 103
Version9 103
vis.adminrole 165
vis.compileclasspath 165
vis.editrole 165
vis.genclasspath 165
vis.money 165
vis.path 165
vis.rulesetxmlpath 166
vis.sessiondata 166
vis.update 166
vis.url 166
vis.urlpath 166
VisitorURLPath 100
wc.enableCacheRet 122
wc.icUploadDir 122
wc.validate 122
WCMSampleAssets 100
WCMSampleAssetsCurrent
Version 100
WebAppName 121
WeblogicVersion 100
WLApplicationDir 121
WLDomainDir 121
WorkflowDir 100
xcelelem.manageuserpub 139
xcelelem.publishfactors 139
xcelelem.publishoptions 139
xcelelem.setpubid 139
xcelerate.adminacl 148
xcelerate.adminrole 148
xcelerate.asset.shareToAllAllowed 127
xcelerate.asset.sizeofnamefield 127
xcelerate.authorize
functions 130
xcelerate.base 148
xcelerate.batchhost 143
xcelerate.batchmode 143
xcelerate.batchpass 143
xcelerate.batchuser 143
xcelerate.blobref 143
xcelerate.body.length 127
xcelerate.bulkapprovechunk 144
xcelerate.charset 141
xcelerate.crosssiteassign 148
xcelerate.defaultacl 127
xcelerate.defaultbase 128
xcelerate.defaultcscacheinfo 128
xcelerate.defaultcsstatus 128
xcelerate.defaultlang 148
xcelerate.defaultpagecriteria 128
xcelerate.defaultpagecriteriaSiteEntry 128
xcelerate.defaultpreviewurl

generator 148
xcelerate.defaultsscacheinfo 128
xcelerate.deny.abstain fromvoting 130
xcelerate.deny.approve 130
xcelerate.deny.authorize 130
xcelerate.deny.build 130
xcelerate.deny.checkout 130
xcelerate.deny.copy 130
xcelerate.deny.delegate 130
xcelerate.deny.delete 131
xcelerate.deny.edit 131
xcelerate.deny.inspect 131
xcelerate.deny.placepage 131
xcelerate.deny.preview 131
xcelerate.deny.removefromgroup 131
xcelerate.deny.removefromworkflow 131
xcelerate.deny.rollback 131
xcelerate.deny.setExportData 131
xcelerate.deny.setnestedworkflow 132
xcelerate.deny.setparticipants 132
xcelerate.deny.setprocessdeadline 132
xcelerate.deny.setstepdeadline 132
xcelerate.deny.share 132
xcelerate.deny.showparticipants 132
xcelerate.deny.showstatus 132
xcelerate.deny.showversion 132
xcelerate.displayablenameattr 151
xcelerate.domain 149
xcelerate.donotregenerate 144
xcelerate.editrole 149
xcelerate.emailattr 151
xcelerate.emailnotification 141
xcelerate.enableinsite 149
xcelerate.ewebeditpro 128
xcelerate.exportmaxfilename 144
xcelerate.fckeditor.basepath 153
xcelerate.grant.abstain fromvoting 133
xcelerate.grant.approve 133
xcelerate.grant.authorize 133
xcelerate.grant.build 133
xcelerate.grant.checkout 133
xcelerate.grant.copy 133
xcelerate.grant.delegate 133
xcelerate.grant.delete 133
xcelerate.grant.edit 133
xcelerate.grant.inspect 133
xcelerate.grant.placepage 134
xcelerate.grant.preview 134
xcelerate.grant.removefromgroup 134
xcelerate.grant.removefromworkflow 134
xcelerate.grant.rollback 134
xcelerate.grant.setExportData 134
xcelerate.grant.setnestedworkflow 134
xcelerate.grant.setparticipants 134
xcelerate.grant.setprocessdeadline 134
xcelerate.grant.setstepdeadline 134
xcelerate.grant.share 135
xcelerate.grant.showparticipants 135
xcelerate.grant.showstatus 135
xcelerate.grant.showversion 135
xcelerate.imageurl 149
xcelerate.localeattr 151
xcelerate.locallanguagedir 137
xcelerate.lockdir 137
xcelerate.MaxLinks 129
xcelerate.mirrorini 145
xcelerate.objpubdir 137
xcelerate.pageref 145
xcelerate.presaveelt 145
xcelerate.previewhost 149
xcelerate.previewservlet 149
xcelerate.previewurlpagename 149
xcelerate.pubabortelt 145
xcelerate.pubcleanupelt 145
xcelerate.pubkeydir 137
xcelerate.publishallassettypes 146
xcelerate.publishinvalidate 146
xcelerate.pubrolesattr 151
xcelerate.pubsetupelt 146
xcelerate.remotecall 147

xcelerate.
  restrictSiteTree 141
xcelerate.rolemanagerclass 150
xcelerate.saveSearchdir 137
xcelerate.seLimit 141
xcelerate.sePath 138
xcelerate.showSiteTree 142
xcelerate.sitenameattr 152
xcelerate.sitesattr 152
xcelerate.sitesroot 152
xcelerate.templatedefault 147
xcelerate.tempobjectsdir 138
xcelerate.thumbnaildir 138
xcelerate.transformpath 140
xcelerate.treeMaxNodes 142
xcelerate.treetabmanagerclass 150
xcelerate.treeType 142
xcelerate.usermanagerclass 150
xcelerate.usese 142
xcelerate.workflowdir 138
xcelerate.workflowengineclass 150
XmlTempArea 100