Oracle® Retail

Functional Artifact Generator Guide

Release 13.2

January 2011

ORACLE®

Oracle Retail Functional Artifact Generator Guide, Release 13.2

Copyright © 2011, Oracle and/or its affiliates. All rights reserved.

Primary Author: Susan McKibbon

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Value-Added Reseller (VAR) Language

Oracle Retail VAR Applications

The following restrictions and provisions only apply to the programs referred to in this section and licensed to you. You acknowledge that the programs may contain third party software (VAR applications) licensed to Oracle. Depending upon your product and its version number, the VAR applications may include:

(i) the **MicroStrategy** Components developed and licensed by MicroStrategy Services Corporation (MicroStrategy) of McLean, Virginia to Oracle and imbedded in the MicroStrategy for Oracle Retail Data Warehouse and MicroStrategy for Oracle Retail Planning & Optimization applications.

(ii) the **Wavelink** component developed and licensed by Wavelink Corporation (Wavelink) of Kirkland, Washington, to Oracle and imbedded in Oracle Retail Mobile Store Inventory Management.

(iii) the software component known as **Access Via™** licensed by Access Via of Seattle, Washington, and imbedded in Oracle Retail Signs and Oracle Retail Labels and Tags.

(iv) the software component known as **Adobe Flex™** licensed by Adobe Systems Incorporated of San Jose, California, and imbedded in Oracle Retail Promotion Planning & Optimization application.

You acknowledge and confirm that Oracle grants you use of only the object code of the VAR Applications. Oracle will not deliver source code to the VAR Applications to you. Notwithstanding any other term or condition of the agreement and this ordering document, you shall not cause or permit alteration of any VAR Applications. For purposes of this section, "alteration" refers to all alterations, translations, upgrades, enhancements, customizations or modifications of all or any portion of the VAR Applications including all reconfigurations, reassembly or reverse assembly, re-engineering or reverse engineering and recompilations or reverse compilations of the VAR Applications or any derivatives of the VAR Applications. You acknowledge that it shall be a breach of the agreement to utilize the relationship, and/or confidential information of the VAR Applications for purposes of competitive discovery.

The VAR Applications contain trade secrets of Oracle and Oracle's licensors and Customer shall not attempt, cause, or permit the alteration, decompilation, reverse engineering, disassembly or other reduction of the VAR Applications to a human perceivable form. Oracle reserves the right to replace, with functional equivalent software, any of the VAR Applications in future releases of the applicable program.

Contents

Preface ix

Audience ix
Related Documents ix
Customer Support ix
Review Patch Documentation x
Oracle Retail Documentation on the Oracle Technology Network x
Conventions x

1 Introduction

Concepts 1-1
Functional Artifact Types 1-3
RTG XML Schemas (XSD) 1-3
RTG JAXB Java Beans 1-3
RTG Objects (Oracle Objects) 1-3
Sample XML File 1-4
Technical Specifications 1-4
Supported Operating Systems 1-4
Command Line 1-4
Graphical User Interface (GUI) 1-4

2 Installation and Basic Setup

Determine Type of Installation 2-1
Installation As a Standalone Application 2-1
Installation As an Application Inside rib-home 2-2
Installation As a Web Application in Oracle WebLogic 2-3
Prerequisites 2-3
Deploy the Artifact Generator Application 2-4
Verify the Artifact Generator Web Application 2-4
Redeploy the Application 2-5

3 General Usage

Prerequisites for Payload Customization 3-1
Rules for Customization/Extension 3-1
Directories Structure 3-2
Customization 3-2
Localization 3-3
Command Line Usage Examples 3-3
Customization - Adding Optional Elements 3-3
Localization- Adding Optional Elements 3-5
Creating a New Payload XSD 3-6
GUI Usage Example 3-7

A Appendix: Screen Captures for Installation

Installation As a Web Application in Oracle WebLogic A-1
Deploy the Artifact Generator Application A-1
Verify the Artifact Generator Web Application A-11
Redeploy the Application A-16

B Appendix: References

Preface

The *Oracle Retail Functional Artifact Generator Guide* provides information about the tool as well as installation instructions.

Audience

The *Oracle Retail Functional Artifact Generator Guide* is intended for the Oracle Retail Integration application integrators and implementation staff, as well as the retailer's Information Technology personnel.

Related Documents

For more information, see the following documents in the Oracle Retail Integration Bus 13.2 documentation set:

- *Oracle Retail Integration Bus Data Model*
- *Oracle Retail Integration Bus Implementation Guide*
- *Oracle Retail Integration Bus Installation Guide*
- *Oracle Retail Integration Bus Operations Guide*
- *Oracle Retail Integration Bus Release Notes*
- *Oracle Retail Integration Bus Hospital Administration Guide*
- *Oracle Retail Functional Artifacts Guide*
- *Oracle Retail Service-Oriented Architecture Enabler Tool Guide*

Customer Support

To contact Oracle Customer Support, access My Oracle Support at the following URL:

`https://support.oracle.com`

When contacting Customer Support, please provide the following:

- Product version and program/module name
- Functional and technical description of the problem (include business impact)
- Detailed step-by-step instructions to re-create
- Exact error message received
- Screen shots of each step you take

Review Patch Documentation

When you install the application for the first time, you install either a base release (for example, 13.2) or a later patch release (for example, 13.2.2). If you are installing the base release, additional patch, and bundled hot fix releases, read the documentation for all releases that have occurred since the base release before you begin installation. Documentation for patch and bundled hot fix releases can contain critical information related to the base release, as well as information about code changes since the base release.

Oracle Retail Documentation on the Oracle Technology Network

Documentation is packaged with each Oracle Retail product release. Oracle Retail product documentation is also available on the following Web site:

http://www.oracle.com/technology/documentation/oracle_retail.html

(Data Model documents are not available through Oracle Technology Network. These documents are packaged with released code, or you can obtain them through My Oracle Support.)

Documentation should be available on this Web site within a month after a product release.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1
Introduction

The Artifact Generator is a collection of tools designed to create the various artifacts used within the Oracle Retail messaging infrastructure from an XML Schema (XSD). These XSDs are called Business Objects. They represent the functional definition and technical structure of a Retail Business Entity.

Messages (business objects) that flow between the retail applications are XML messages. Oracle Retail XML message definitions are defined statically through XML schemas (XSDs). The integration infrastructure works with multiple technologies (Java EE, PL/SQL) and so has different ways of representing the same functional XML message structure in different technologies. To make it easier to maintain the various functional artifacts, the Artifact Generator was developed.

The Artifact Generator tool is being made available to give customers the ability to add/modify data which flows from one application to the other.

This guide provides details on the installation and configuration of the tool.

Note: For more complete scenarios and best practices on usage of the tool, see Chapter 11, "Customization and Extension," in the *Oracle Retail Integration Bus Implementation Guide*. Also see the *Oracle Retail Service-Oriented Architecture Enabler Tool Guide*.

Concepts

The functional artifacts are different representations of the same message structure/definition in different technologies (Java EE, PL/SQL). Depending on the retail application's technology, appropriate artifacts are used, converting one from the other as needed.

The core concept is that the single source of truth is the XSD. The XSDs are strict and used by the Artifact Generator to produce the design time physical objects used by the application's API technology (PL/SQL or Java), as well as the runtime validations used by the various integration components.

The most common customization requirements in messaging are the addition of new elements to existing payloads, or the creation of new payloads to support custom business logic added to the base integrated applications.

Each retail message family and type combination maps to one and only one functional message definition. One functional message definition can map to one or more than one family/type combination within the same family. The RTG Integration Guide details these objects and the relationships.

Note: See the Oracle RTG Integration Guide.

The Oracle AIA approach and Enterprise Business Object (EBO) model, as well as other industry standards have defined an approach using well known tags and locations to separate the custom extension from the base. This allows the extensions to be preserved as updates to base are applied. The Oracle Retail Business Objects have been designed and constructed to accommodate customer extensions following the Oracle AIA EBO standards and guidelines.

For details and in depth examples, see:

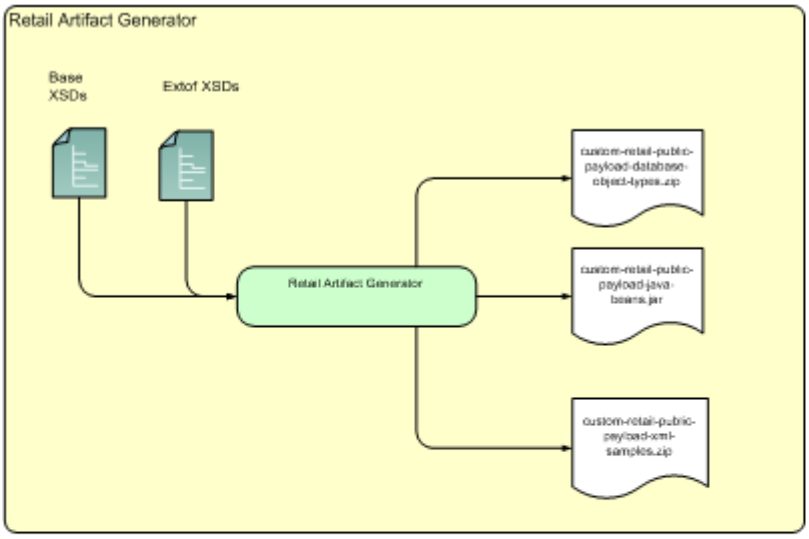
- *Oral Retail Functional Artifacts Guide*
- Oracle Application Integration Architecture - Enterprise Object Library: Enterprise Business Objects and Messages XML Naming and Design Rules

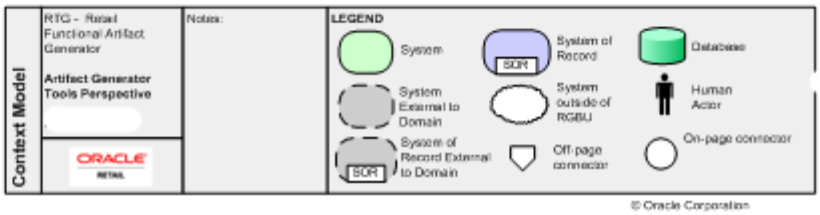
These standards and conventions define Business Object extension and customization as customer side activities. The RGBU governance process produces a Business Object that is enterprise wide. Changes or additions handled by versioning are packaged as part of a release and defined as Base Objects.

The Artifact Generator is the core tool used for customization and extension of the Business Objects used by the RIB and Web Services generated by the Retail SOA Enabler Tool (RSE).

For details and examples, see:

- *Oracle Retail SOA Enabler Tool Guide.*
- Chapter 11, "Customization and Extension," of the *Oracle Retail Integration Bus Implementation Guide*.





Functional Artifact Types

The functional artifacts are different representations of the same message structure/definition in different technologies (Java EE, PL/SQL). Depending on the Oracle Retail application's technology, RTG uses the appropriate artifacts, converting one from the other as needed. The following are the RTG functional object definitions.

RTG XML Schemas (XSD)

The functional XML message structure is a contract between the integrating retail applications and is defined by the XML schemas. All the other artifacts are generated from the XML schemas. XML schemas are the inputs required by the artifact generators.

RTG JAXB Java Beans

JAXB is a standard Java XML binding technology. It provides the mechanism to convert XML instances to Java objects (and vice versa) in a standard way. The Java EE Web service infrastructure internally uses JAXB to marshall and unmarshall the SOAP messages. For every payload XSD, the artifact generator generates the corresponding JAXB beans.

RTG Objects (Oracle Objects)

PL/SQL retail applications communicate with the integration infrastructure using Oracle Objects. These objects are user-defined database objects that define the XML message structure inside the database.

Sample XML File

The tool generates example XML files that represent instances of XML message schemas. Each element is present and has appropriate data to the full declared length.

Technical Specifications

The Oracle Retail Artifacts Generator has dependencies on Oracle Retail Application installations, as well as on the Oracle Application Servers. This section covers these requirements.

Supported Operating Systems

For the Artifacts Generator tool, there are separate requirements for the Command Line and the GUI.

Command Line

Supported On	Version Supported
Operating System	OS certified: ■ Oracle Enterprise Linux 5 update 2 (OEL 5.3) for Linux x86-64 ■ AIX 6.1 TL1

Graphical User Interface (GUI)

Supported On	Version Supported
Oracle WebLogic Server OS	OS certified with OracleWebLogic Server 11 g 10.3.3. Options are: ■ Oracle Enterprise Linux 5 update 2 (OEL 5.3) for Linux x86-64 ■ AIX 6.1 TL1 ■ Solaris 10 ■ HP-UX 11.31 Itanium
Oracle WebLogic Server	Oracle WebLogic Server 11g Release 3 (10.3.3.0)

2

Installation and Basic Setup

This chapter provides instructions for installing and deploying the Oracle Retail Artifacts Generator.

Determine Type of Installation

The Oracle Retail Artifacts Generator can be installed and used in any of the following configurations:

- Standalone application
- Application inside rib-home
- Web-application in Oracle WebLogic

Installation As a Standalone Application

1. Determine the user and location to install the Artifact Generator.
2. Verify the JAVA_HOME environment variable is set for the user. The JAVA_HOME must be set to a Java 1.5 JDK. If the user is located on the same server as the Application Server, then setting the JAVA_HOME to $ORACLE_HOME/jdk is recommended.

   ```
   > echo $JAVA_HOME
   >/home/aia1/oracle/middleware/jdk
   ```

3. Create a directory for the Artifact Generator.

   ```
   > mkdir ArtifactGeneratorStandalone
   ```

4. Download and extract the Artifact Generator to the Artifact Generator home directory.

   ```
   > cd ArtifactGeneratorStandalone
   > cp /u00/stage/RIB13.2.0/ ArtifactGenerator13.2.0ForAll13.2.0_eng_ga.tar.
   > tar -xvf  ArtifactGenerator13.2.0ForAll13.2.0_eng_ga.tar
   ```

   This will create the Artifact Generator root directory structure. For example: /user/aia1/ArtifactGenerator/retail-func-artifact-gen.

This becomes AG_HOME.

Note: AG_HOME is assumed to be the artifact-generator-home directory in the following steps.

```
> export AG_HOME=/user/home/aia1/ArtifactGenerator/retail-func-artifact-gen
```

5. Download the RIB Functional Artifacts tar to the AG_HOME/base-func-artifacts directory.

```
> cd $AG_HOME/base-func-artifacts
> cp /u00/stage/RIB13.2.0/RibFuncArtifact13.2.0ForAll13.2.0Apps_eng_ga.tar.
```

Do not untar it. This will be handled by the setup scripts.

6. Make groovy executable.

```
> cd $AG_HOME
> chmod 711 ./integration-lib/third-party/groovy/1.6.4/bin/groovy
```

7. Set the groovy environment variable. The GROOVY_HOME must be set. The Artifact Generator ships with the groovy jar files.

```
> cd $AG_HOME
> export GROOVY_HOME=`pwd`/integration-lib/third-party/groovy/1.6.4
```

8. Execute the setup script.

```
>$GROOVY_HOME/bin/groovy
com/oracle/retail/integration/artifact/generator/SetupWorkArea.groovy
```

9. Run the Artifact Generator to verify installation. All subdirectories and the various artifacts should be created without errors.

```
>$GROOVY_HOME/bin/groovy
com/oracle/retail/integration/artifact/generator/GenArtifacts.groovy
```

10. Installation is complete. See the section on usage.

Installation As an Application Inside rib-home

For the following steps, $RIB_HOME is assumed to be the rib-home directory in the rib-app-builder directory tree structure.

For example: /u00/rib/Rib1320ForAll13xxApps/rib-home

1. Verify the JAVA_HOME environment variable. The JAVA_HOME must be set to a Java 1.5 JDK. If the <RIB_HOME> workspace is located on the same server as the Application Server, then setting the JAVA_HOME to $ORACLE_HOME/jdk is recommended.

```
> echo $JAVA_HOME
> /home/aia1/oracle/middleware/jdk
```

2. Download and extract the Artifact Generator to the Artifact Generator into the rib-home/tools-home directory. There already will be a placeholder directory /retail-func-artifact-gen.

```
> cd $RIB_HOME/tools-home
> cp /u00/stage/RIB13.2.0/ ArtifactGenerator13.2.0ForAll13.2.0_eng_ga.tar.
> tar -xvf  ArtifactGenerator13.2.0ForAll13.2.0_eng_ga.tar
```

This will extract the Artifact Generator into the placeholder directory:

3. Make Groovy executable.

```
> cd $RIB_HOME/tool-home/ retail-func-artifact-gen
> chmod 711 ./integration-lib/third-party/groovy/1.6.4/bin/groovy
```

4. Set the groovy environment variable.

```
> cd $RIB_HOME/tools-home/retail-func-artifact-gen
> export GROOVY_HOME=`pwd`/integration-lib/third-party/groovy/1.6.4
```

5. Execute the setup script.

```
$GROOVY_HOME/bin/groovy
com/oracle/retail/integration/artifact/generator/SetupWorkArea.groovy
```

6. Verify that groovy in $RIB_HOME is executable.

```
> cd $RIB_HOME
> chmod 711./integration-lib/third-party/groovy/1.6.4/bin/groovy
```

7. Set the Groovy environment variable to rib-home.

```
> cd $RIB_HOME
> export GROOVY_HOME=`pwd`/integration-lib/third-party/groovy/1.6.4
```

8. Run the Artifact Generator to verify installation. All subdirectories and the various artifacts should be created without errors.

```
> cd $RIB_HOME/tools-home/retail-func-artifact-gen
> $GROOVY_HOME/bin/groovy
com/oracle/retail/integration/artifact/generator/GenArtifacts.groovy
```

9. Installation is complete. See Chapter 3, "General Usage."

Installation As a Web Application in Oracle WebLogic

The steps below describe how to deploy the Oracle Retail Artifact Generator to an Oracle WebLogic Application Server as a Web application.

> **Note:** See the section "Supported Operating Systems," in Chapter 1.

Prerequisites

- The retail-func-artifact-gen-gui.war file is located within the directory structure of the ArtifactGenerator13.2.0ForAll13.2.0_eng_ga.tar. It is recommended that the Artifact Generator be deployed from the rib-home location, although the.war file can be obtained from the stand-alone installation as well.

> **Note:** See the section, "Installation As an Application Inside rib-home," in this chapter.

- The installation and base configuration of the Oracle WebLogic Server is beyond the scope of this document. Work with the Application Server Administration team to determine the physical and logical placement of the retail-func-artifact-gen-gui component within the WebLogic Server deployment.

Note: See Oracle WebLogic Server 11g Release 3 (10.3.3) Installation Guide.

Deploy the Artifact Generator Application

Using the WebLogic Server Administration Console, complete the following steps:

Note: For instructions with illustrations (screen captures), see "Appendix: Screen Captures for Installation."

---

1. Navigate to the Deployments page:
2. Click **Install**.

Note: If the application has already been installed, see the section, "Redeploy the Application," in this chapter.

The "Locate deployment to install and prepare for deployment" page is displayed. Follow the instructions to locate the retail-func-artifact-gen-gui.war file.

3. Select **Upload Files**.
4. On the **Upload a Deployment to the admin server** page, use the Browse button to locate the retail-func-artifact-gen-gui.war file in the "Deployment Archive."
5. Select the retail-func-artifact-gen-gui.war
6. Click **Next** and move to **Choose targeting style**.
7. Select **Install this deployment as an application**.
8. Click **Next** and move to **Optional Settings**.
9. Click **Next** and move to **Review your choices and click Finish**.
10. Select **No, I will review the configuration later**.
11. Click **Finish** to deploy the application.

Verify the Artifact Generator Web Application

1. Navigate to the Deployments page.
2. Locate the **retail-func-artifact** on the Summary of Deployments page.
3. Click the name, **retail-func-artifact-gen-gui**, to move to the "Settings for the rib-func-artifact-gen-gui."
4. Select the Testing tab.

5. Click the link for the index.jsp URL in the Test Point. The URL should open to the Retail Artifact Generator Home page
6. The installation is complete. See Chapter 3, "General Usage."

Redeploy the Application

If the retail-func-artifact-gen-gui application has already been deployed, follow these steps:

1. If the retail-func-artifact-gen-gui application is running, select **Stop** and **When Work Completes** or **Force Stop Now,** depending on the environment. The recommended option always is **When Work Completes.**
2. Click **Delete.**
3. The retail-func-artifact-gen-gui should now not show on the Summary of Deployment page.
4. Return to the appropriate step in Deploy the Application.

3

General Usage

The Artifact Generator tool can be used through the command line or through the GUI, depending on the installation type selected.

The Artifact Generator implements rules for customization/extension of the Business Objects that are used to create the Functional Artifacts for use in the RTG Integration systems. The tools, regardless of the installation type, will process only XSDs that have been added or modified per these rules.

The fundamental rule is that for customization of a base XSD, the ExtOf XSDs must be modified and not the original (base) XSD. The tool scans for customization/extension implemented there, and only there. Any changes to the base XSD will be ignored.

For example, if you want to add an optional element to Currency Rate flow, add the optional element to ExtOfCurrRateDesc.xsd and not the CurrRateDesc.xsd.

There are simple examples included in the usage sections, but for more complete scenarios and best practices on usage of the tool, see Chapter 11, "Customization and Extension," in the *Oracle Retail Integration Bus Implementation Guide*. See also the *Oracle Retail Service Oriented Architecture Enabler Tool Guide*.

Prerequisites for Payload Customization

The following are prerequisites for payload customization.

- Familiarity with the Artifacts Generator tool.
- Understanding the importance of payloads and how they fit into the overall retail family of products.
- Understanding the impact of customizing a payload on other applications.

Rules for Customization/Extension

The following are rules for customization and extension.

- Always make a backup of the particular files being modified during customization.
- Customizations/Extensions of payloads must also be made accordingly to the application side.
- It is strongly recommended the only optional elements are added. The addition of mandatory elements increases complexity.

- The names of the elements in ExtOfs should not be the same as the names in the parent XSD. For example, if CurrRateDesc.xsd has an element name = **attr1**, ExtOfCurrRateDesc cannot have an element named **attr1**.
- Java keywords cannot be used in the names of elements in XSDs.

Directories Structure

This section provides examples of directories structure for customization and localization.

Customization

The installation creates a directory structure that contains all the libraries (integration-lib) and generally available (GA) input artifacts required to generate all support output types.

There are output directories for each type of artifact produced. For example:

```
./output-database-object-types
    |----------- src
                    |---- ExtOfASNInDesc.sql
    |----------- dist
                    |---- custom-retail-public-payload-database-object-types.zip
./output-jaxb-java-beans
    |----------- src
                    |----
com/oracle/retail/integration/custom/bo/extofasnindesc/v1
                             |----- ExtOfASNInDesc.java…
                                     ObjectFactory.java
                                     package-info.java
    |----------- dist
                    |---- custom-retail-public-payload-java-beans.jar
./output-xml-samples
    |----------- src
                    |---- ExtOfASNInDesc.xml
    |----------- dist
                    |---- custom-retail-public-payload-xml-samples.zip
```

Localization

The installation creates a directory structure that contains all the libraries (integration-lib) and generally available (GA) input artifacts required to generate all support output types.

There are output directories for each type of artifact produced. For example:

```
./output-database-object-types
    |----------- src
                    |---- BrASNInDesc.sql
    |----------- dist
                    |----
localization-retail-public-payload-database-object-types.zip
./output-jaxb-java-beans
    |----------- src
                    |----
com/oracle/retail/integration/localization/bobrasnindesc/v1
                            |----- BrASNInDesc.java…
                                    ObjectFactory.java
                                    package-info.java
    |----------- dist
                    |---- localization-retail-public-payload-java-beans.jar
./output-xml-samples
    |----------- src
                    |---- BrASNInDesc.xml
    |----------- dist
                    |---- localization-retail-public-payload-xml-samples.zip
```

Command Line Usage Examples

It is important to understand that the customization/extension of existing Business Objects should be performed in the ExtOfxxx XSDS of a base XSD, not the base XSD. The tool is design to enforce this best practice. It supports the preservation of customization/extension when there are new versions of the base objects released.

For example, to add an optional element to Currency Rate flow, add the optional element to ExtOfCurrRateDesc.xsd and not the CurrRateDesc.xsd.

The basics of XSDs are not covered in this document.

Note: See "Appendix: References."

Customization - Adding Optional Elements

This explains how to add an optional element (simple type or complex type) to existing message payloads.

1. Edit the desired payload XSDs in ./input-xsd/payload/xsd/retail/integration/custom/bo directory of Rib Artifact Generator tool installation. Add the optional simple or complex element to the particular message family xsd. If needed, define the type it belongs to if it doesn't exist.

```
> cd input-xsd/payload/xsd/retail/integration/custom/bo/ExtOfCurrRateDesc/v1
> vi ExtOfCurrRateDesc.xsd (make changes)
```

The following example shows the modifications to ExtOfCurrRateDesc.xsd required to add the optional element, **country**.

```
<xs:schema elementFormDefault="qualified"
targetNamespace="http://www.oracle.com/retail/integration/custom/bo/ExtOfCurrRa
teDesc/v1"
    version="1.0"

xmlns="http://www.oracle.com/retail/integration/custom/bo/ExtOfCurrRateDesc/v1"
    xmlns:retailDoc="http://www.w3.org/2001/XMLSchema"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="ExtOfCurrRateDesc">
        <xs:complexType>
            <xs:sequence>
                        <xs:element minOccurs="0" name="country"
type="varchar23">
                        </xs:element>
             </xs:sequence>
        </xs:complexType>
    </xs:element>
        <xs:simpleType name="varchar23">
                <xs:restriction base="xs:string">
                        <xs:maxLength value="3"/>
                </xs:restriction>
        </xs:simpleType>
</xs:schema>
```

2. Run the Artifact Generator to generate various functional artifacts.

```
> $GROOVY_HOME/bin/groovy
com/oracle/retail/integration/artifact/generator/GenArtifacts.groovy
```

3. All necessary artifacts are generated as follows:

   - custom-retail-public-payload-java-beans.jar is generated in retail-func-artifact-gen/output-jaxb-java-beans/dist folder
   - custom-retail-public-payload-database-object-types.zip is generated in retail-func-artifact-gen/output-database-object-types/dist folder

Localization- Adding Optional Elements

This explains how to add an optional element (simple type or complex type) to existing message payloads.

1. Edit the desired payload XSDs in ./input-xsd/payload/xsd/retail/integration/localization/bo directory of Rib Artifact Generator tool installation. Add the optional simple or complex element to the particular message family xsd. If needed, define the type it belongs to if it doesn't exist.

```
> cd input-xsd/payload/xsd/retail/integration/localization/bo/BrCurrRateDesc/v1
> vi BrCurrRateDesc.xsd (make changes)
```

The following example shows the modifications to BrCurrRateDesc.xsd required to add the optional element, "country."

```
 <xs:schema elementFormDefault="qualified"

targetNamespace="http://www.oracle.com/retail/integration/localization/bo/BrCur
rRateDesc/v1"
    version="1.0"

xmlns="http://www.oracle.com/retail/integration/localization/bo/BrCurrRateDesc/
v1"

xmlns:EOfBrCurrRateDesc="http://www.oracle.com/retail/integration/custom/bo/EOf
BrCurrRateDesc/v1"
    xmlns:retailDoc="http://www.w3.org/2001/XMLSchema"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:import

namespace="http://www.oracle.com/retail/integration/custom/bo/EOfBrCurrRateDesc
/v1"
schemaLocation="../../../../custom/bo/EOfBrCurrRateDesc/v1/EOfBrCurrRateDesc.xs
d">
        <retailDoc:annotation>
            <retailDoc:documentation>It's a referenced element. For detailed
description, please refer referenced element doc.</retailDoc:documentation>
        </retailDoc:annotation>
    </xs:import>
<xs:element name="BrCurrRateDesc">
<xs:complexType>
            <xs:sequence>
<xs:element minOccurs="0" name="country" type="varchar23">

            <xs:element maxOccurs="1" minOccurs="0"
ref="EOfBrCurrRateDesc:EOfBrCurrRateDesc">
               <retailDoc:annotation>
                 <retailDoc:documentation>Provide an extension hook to
customize CurrRateDesc
                 </retailDoc:documentation>
               </retailDoc:annotation>
             </xs:element>
            </xs:sequence>
        </xs:complexType>
</xs:element>
        <xs:simpleType name="varchar23">
```

```
                <xs:restriction base="xs:string">
                        <xs:maxLength value="3"/>
                </xs:restriction>
        </xs:simpleType>

</xs:schema>
```

2. Run the Artifact Generator to generate various functional artifacts.

```
> $GROOVY_HOME/bin/groovy
com/oracle/retail/integration/artifact/generator/GenArtifacts.groovy  -g
generateLocalization
```

3. All the necessary artifacts are generated as follows:

   localization-retail-public-payload-java-beans.jar is generated in retail-func-artifact-gen/output-jaxb-java-beans/dist folder

   localization-retail-public-payload-database-object-types.zip is generated in retail-func-artifact-gen/output-database-object-types/dist folder

Creating a New Payload XSD

The following steps must be completed to add a new XSD to a current set of payloads:

1. Create a new XSD which conforms to the MetaSchema (IntegrationMetaSchema.xsd). The artifact generator tool checks the validity of the schema before generating any artifacts. The artifact generator will fail if the XSD is not compliant with the MetaSchema.
2. Drop the new XSD in here: /input-xsd/ payload/xsd/retail/integration/custom/bo directory of the RIB Artifact Generator tool installation.

   For example:

   - For a new XSD called "FooDesc.xsd, create it under./input-xsd/ payload/xsd/retail/integration/custom/bo/custom/v1.
   - To conform to the standards, create the ExtOfCustom.xsd and place it under./input-xsd/ payload/xsd/retail/integration/custom/bo/ExtOfFooDesc/v1.
3. Run the Artifact Generator to generate various functional artifacts.

```
> $GROOVY_HOME/bin/groovy
com/oracle/retail/integration/artifact/generator/GenArtifacts.groovy
```

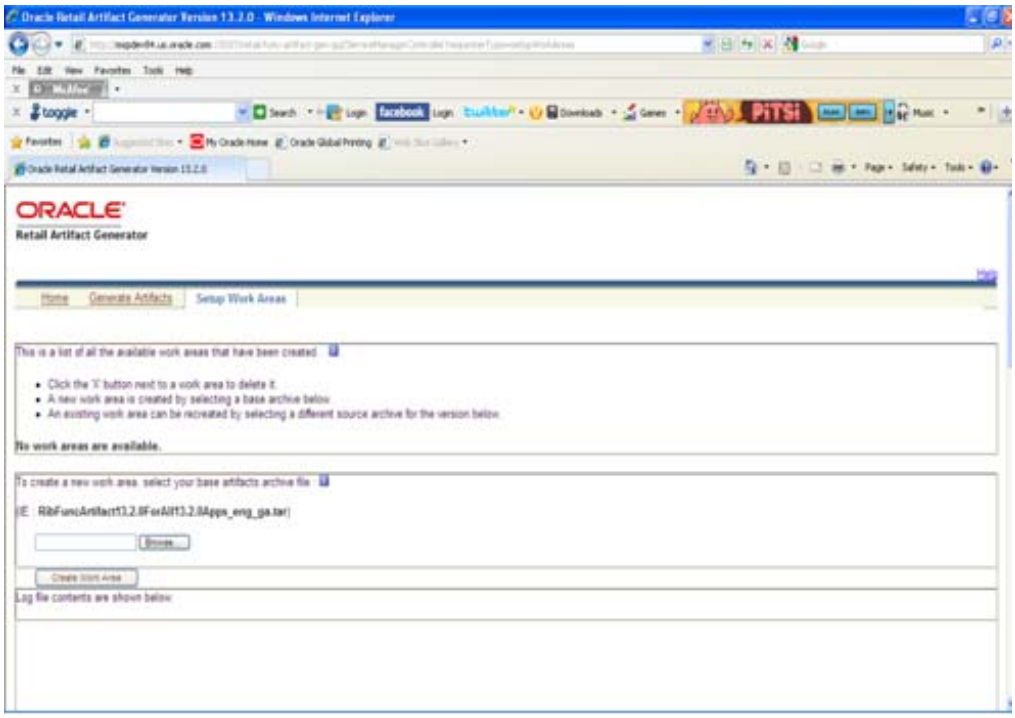
   Upon completion of this step, all necessary artifacts are generated as follows:

   - custom-retail-public-payload-java-beans.jar is generated in retail-func-artifact-gen/output-jaxb-java-beans/dist folder
   - custom-retail-public-payload-database-object-types.zip is generated in retail-func-artifact-gen/output-database-object-types/dist folder

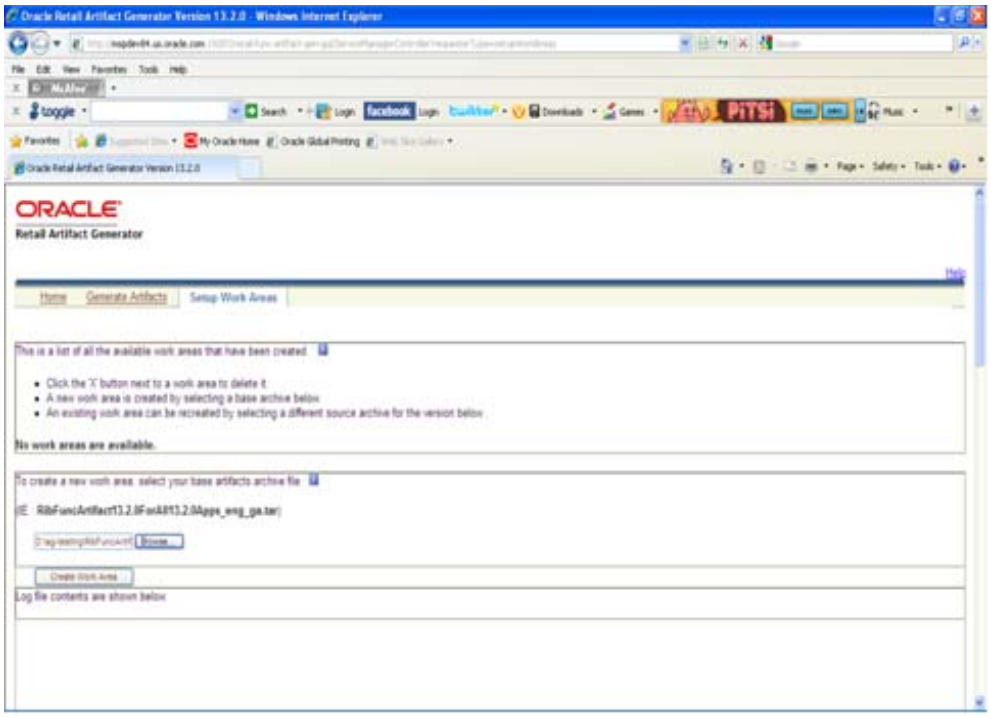
GUI Usage Example

1. Navigate to Retail Artifact Generator Home.

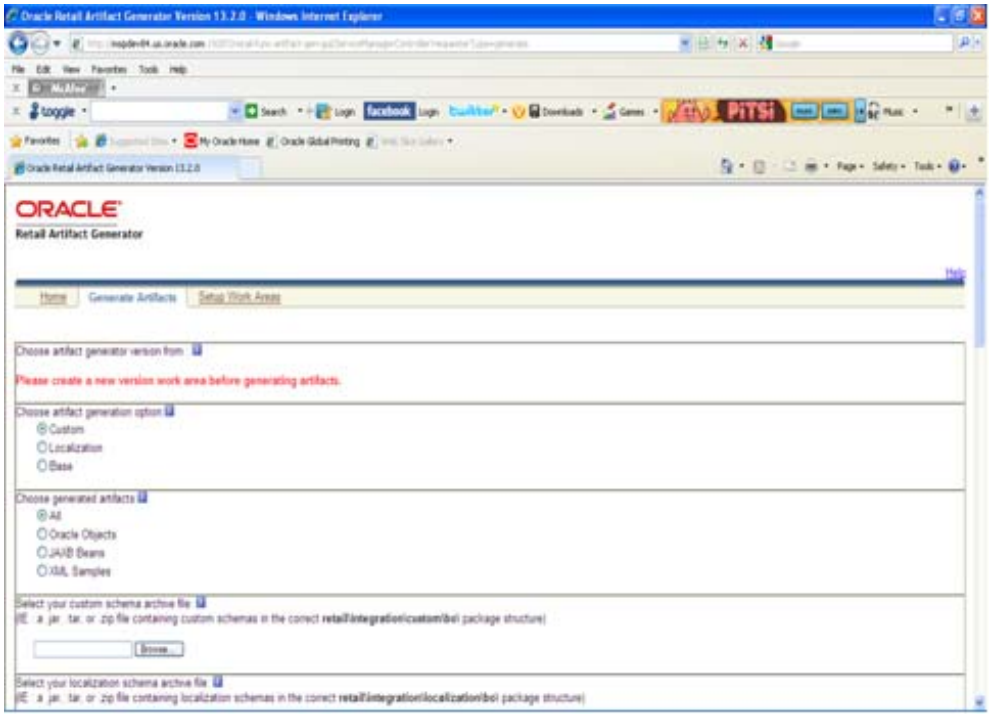
2. To create a new version workspace, select the **SetupWork Areas** tab.



3. On the SetupWork Areas page, use the browse button to locate the RibFuncArtifact13..2.0ForAll13.2.0Apps_eng_ga.tar or the RetailFuncArtifact13.2.0ForAll13.2.0Apps_eng_ga.tar. These .tar files must be in a location that is accessible by the browser.



4. Click **Create Work Area**.
5. To generate artifacts, select the **Generate Artifacts** tab.



6. Choose the artifact generator version.
7. Choose an artifact generation option (Base or Custom or Localization).

If Custom/Localization is chosen, on the Generate Artifacts page, use the browse button to locate the archive file (for example, .tar, .jar, or .zip) that contains the custom/localized schemas. This file must have the custom/localization schemas in the correct package structure, such as retail/integration/custom/bo* or retail/integration/localization/bo*.

Note: For information on the packaging structure and guidelines, see the *Oracle Retail Functional Artifacts Guide*.

Note: For information about how to create the custom schemas, see the "Command Line Usage Examples" in this chapter.

---

8. Click **Generate Artifacts** to start the generation process.

   As the process runs, the status of its progress is displayed in the log window. When the process is complete, **save file** dialog windows are displayed with options to save the resulting archive files or open them for review.

A

Appendix: Screen Captures for Installation

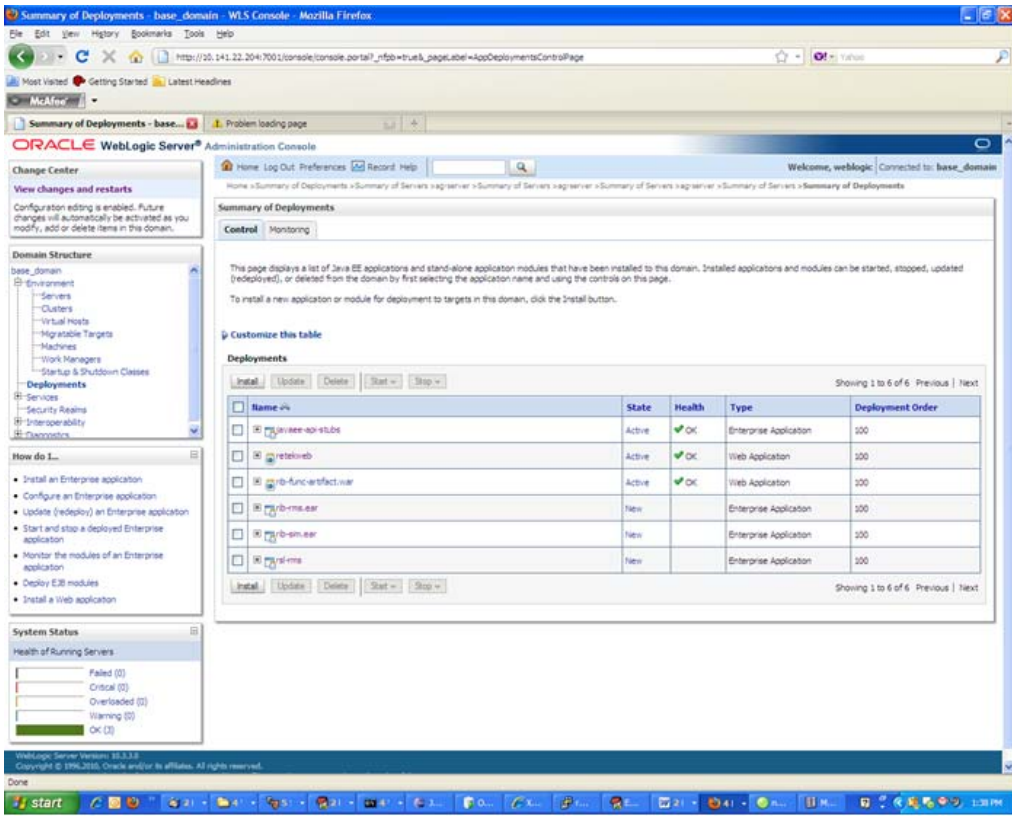
This appendix provides step-by-step instruction (with illustrations) for installing the Artifact Generator as a Web application in Oracle WebLogic.

Installation As a Web Application in Oracle WebLogic

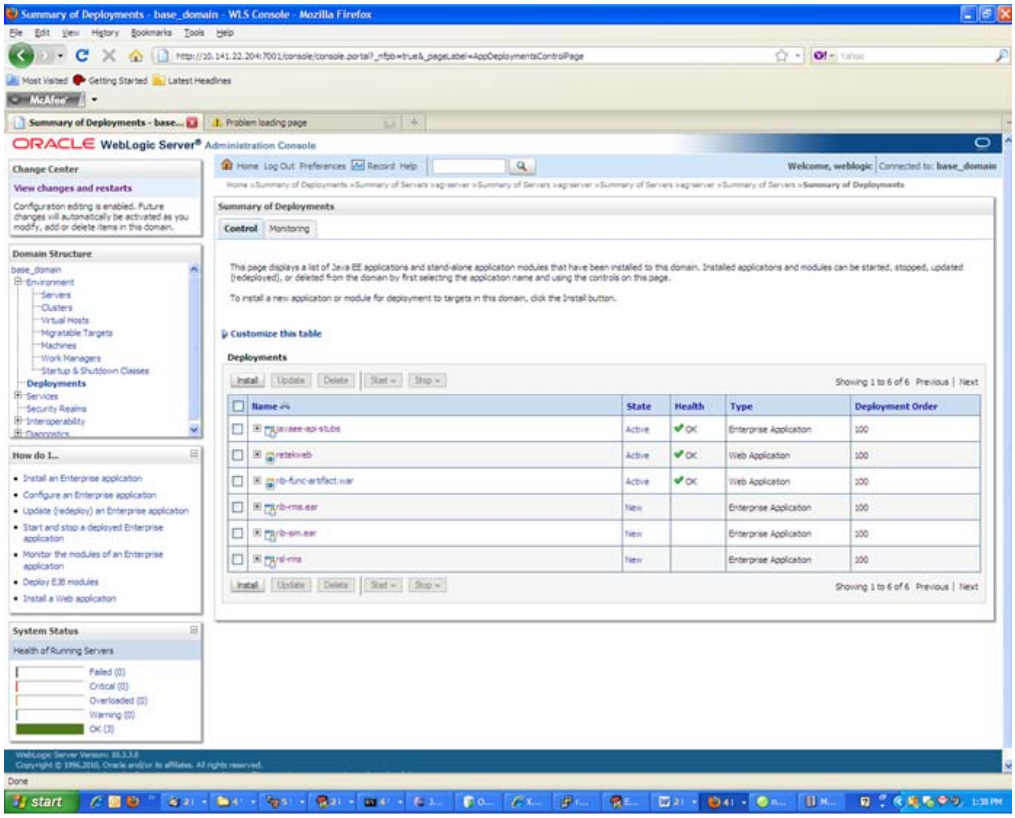
Deploy the Artifact Generator Application

Using the WebLogic Server Administration Console, complete the following steps.

1. Navigate to the Deployments page.



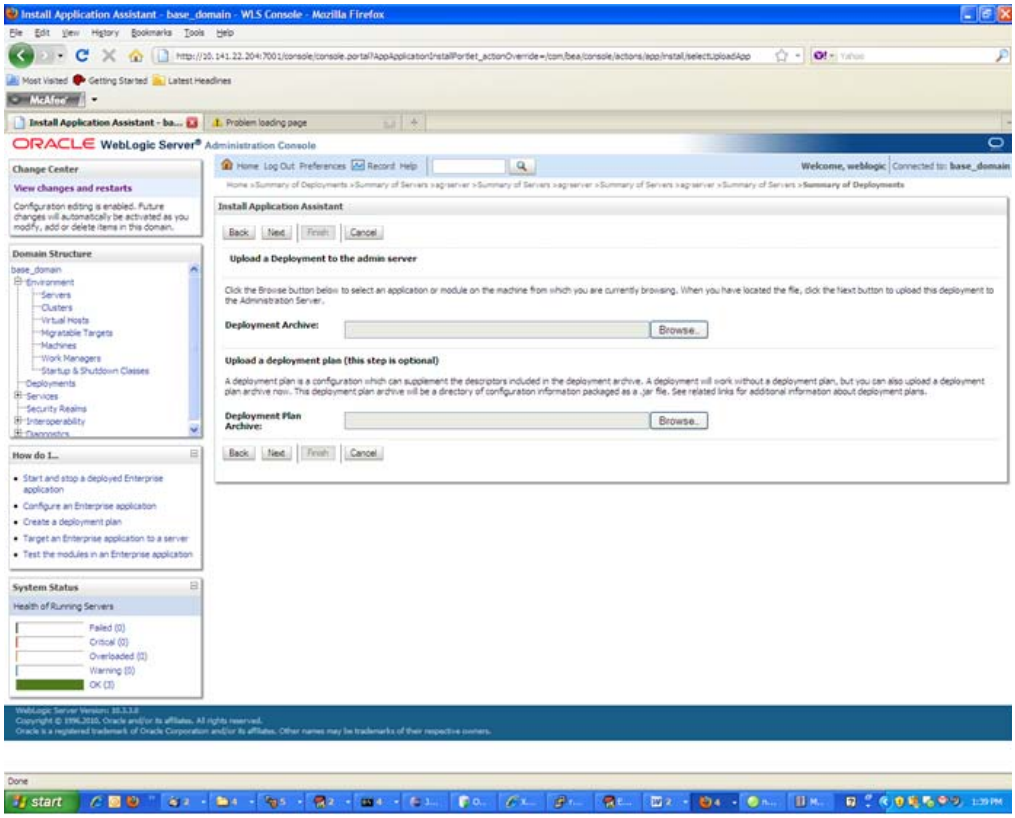
2. Click **Install**.



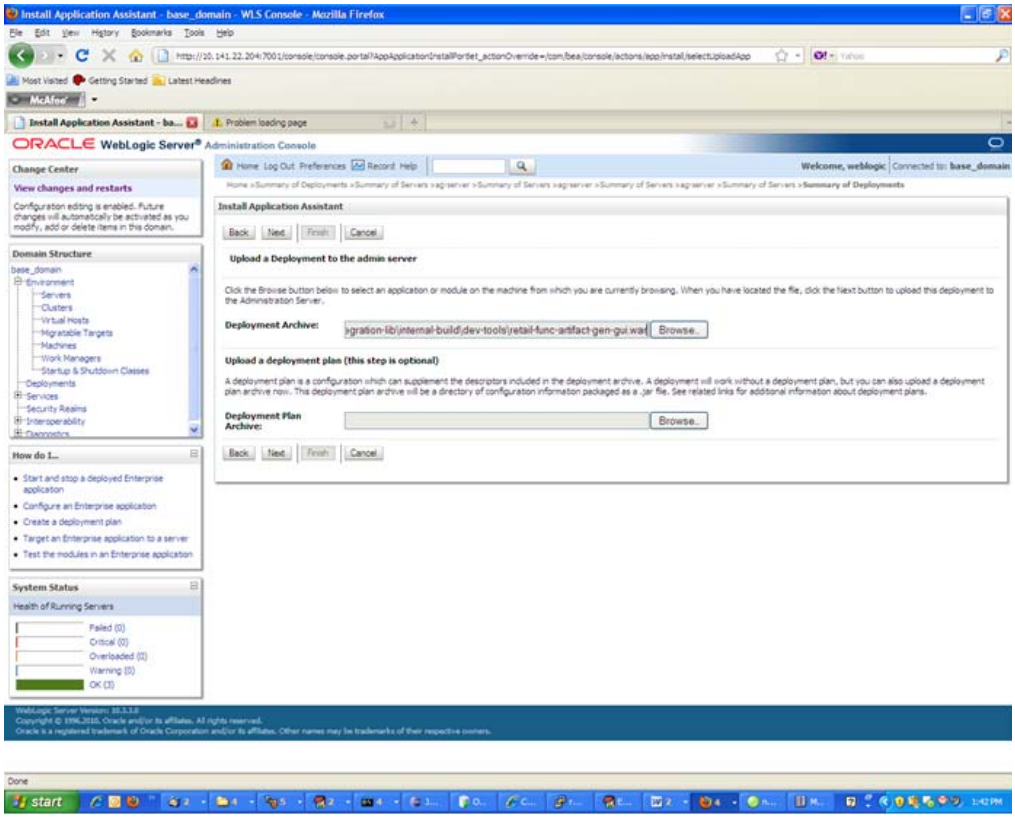
Note: If the application has already been installed, see the section, "Redeploy the Application."

The **Locate deployment to install and prepare for deployment** page is displayed. Follow the instructions to locate the retail-func-artifact-gen-gui.war file.

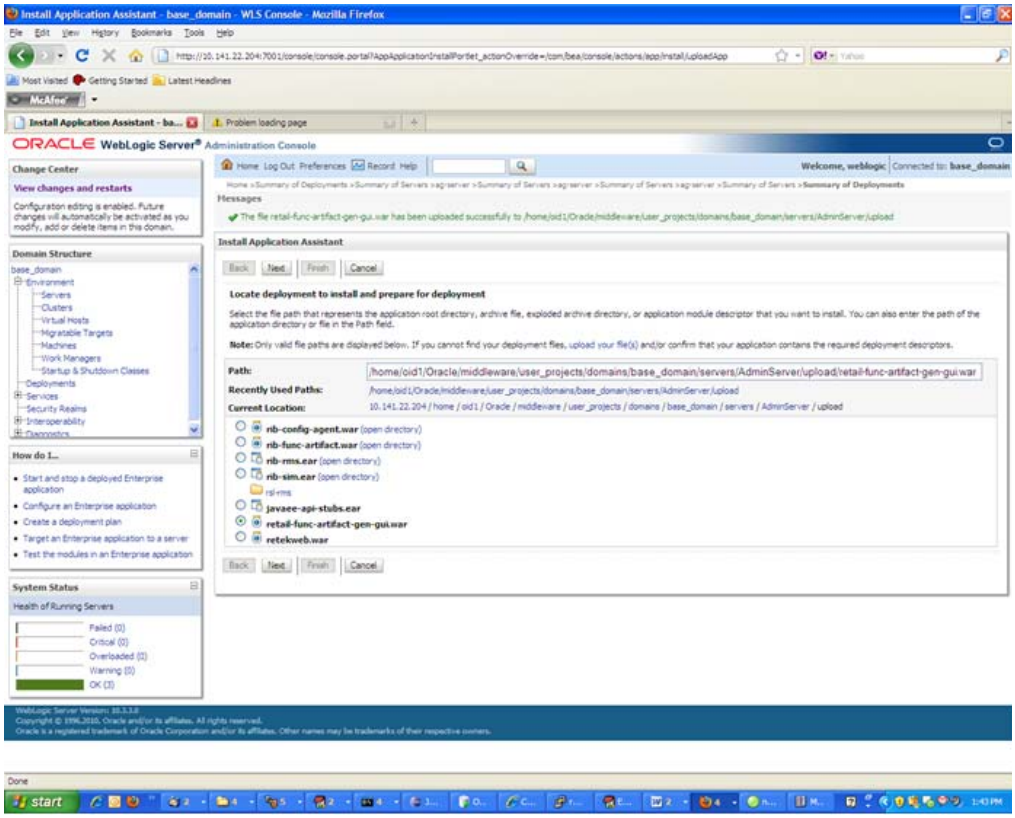
3. Select **Upload your File(s)**.



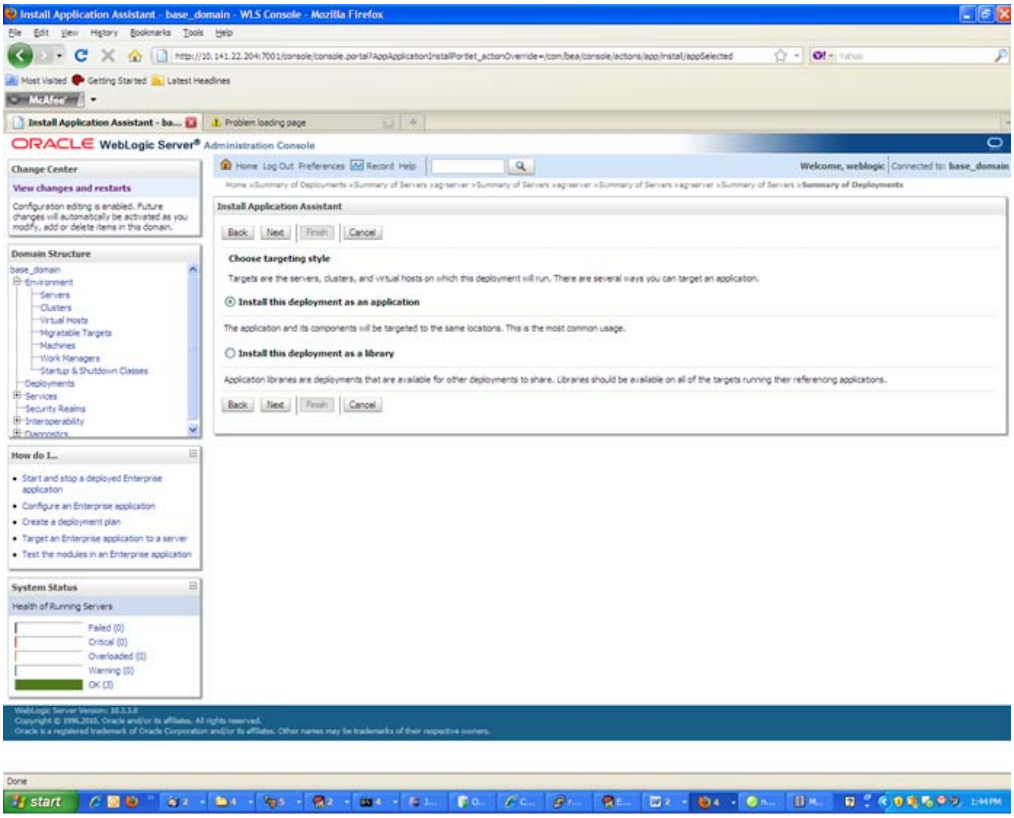
4. On the **Upload a Deployment to the admin server** page, use the Browse button to locate the retail-func-artifact-gen-gui.war file in the Deployment Archive.



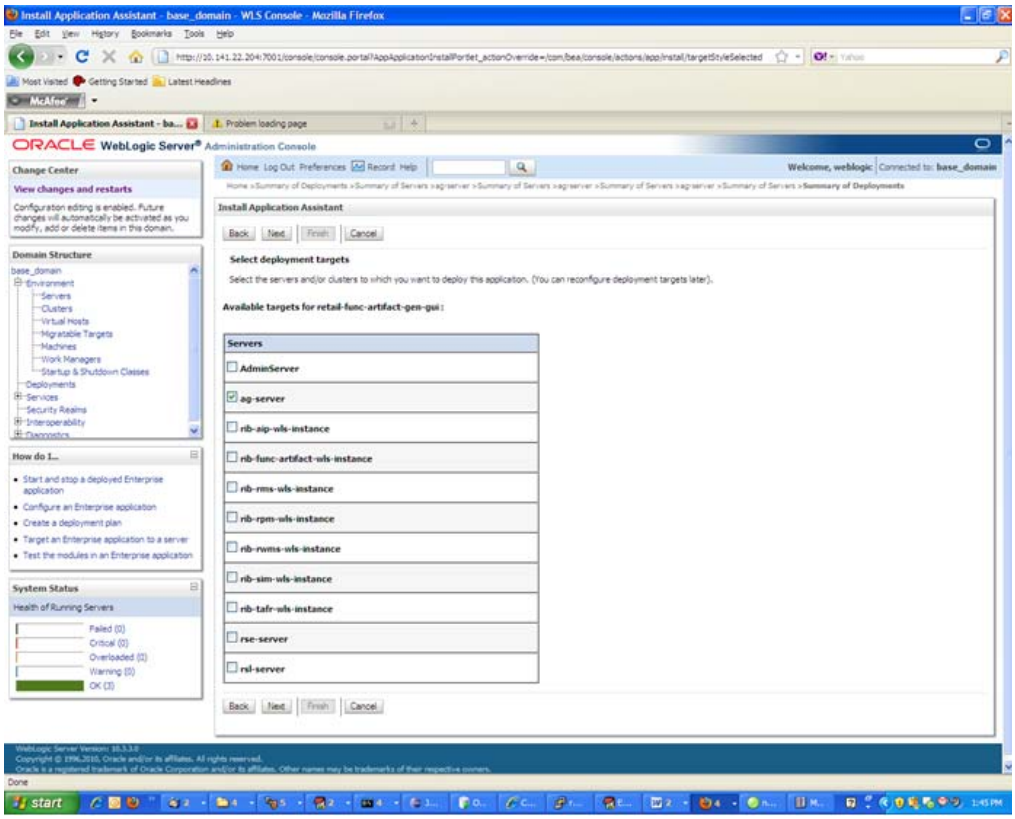
5. Select **retail-func-artifact-gen-gui.war**.



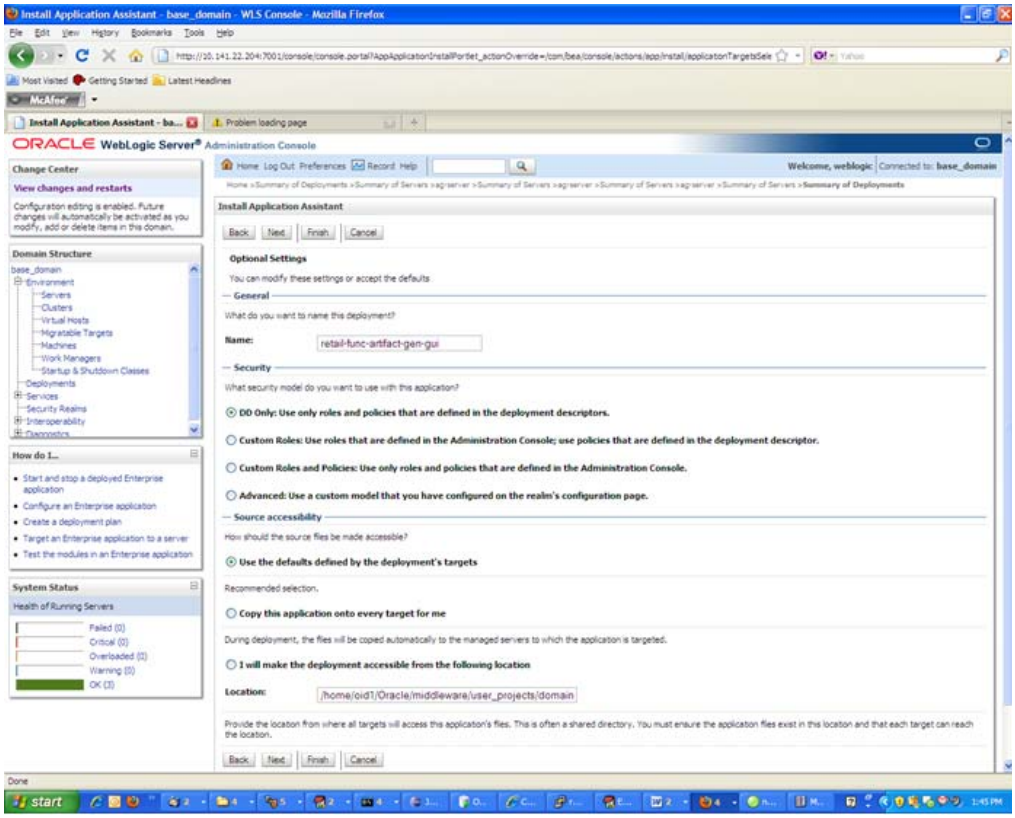
6. Click **Next** and move to **Choose targeting style**.



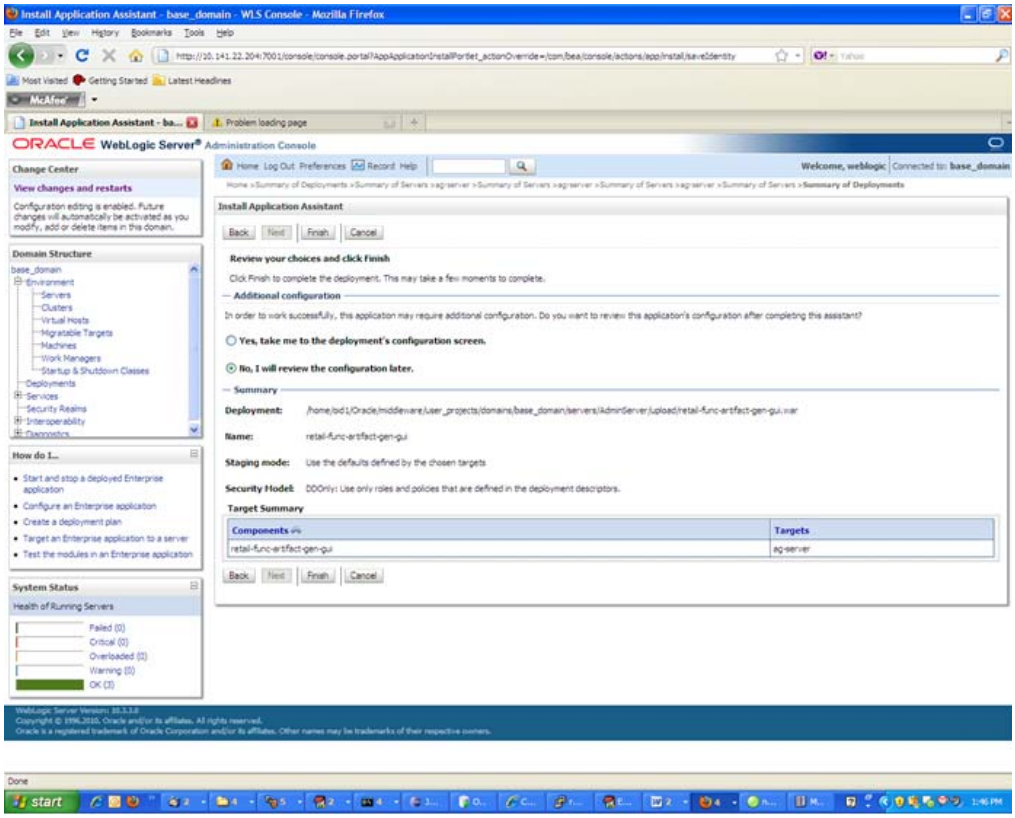
7. Select **Install this deployment as an application**.
8. Click **Next** and move to **Select deployment targets**.



9. Click the managed server to which the application will be deployed. Click **Next** and move to **Optional Settings**.

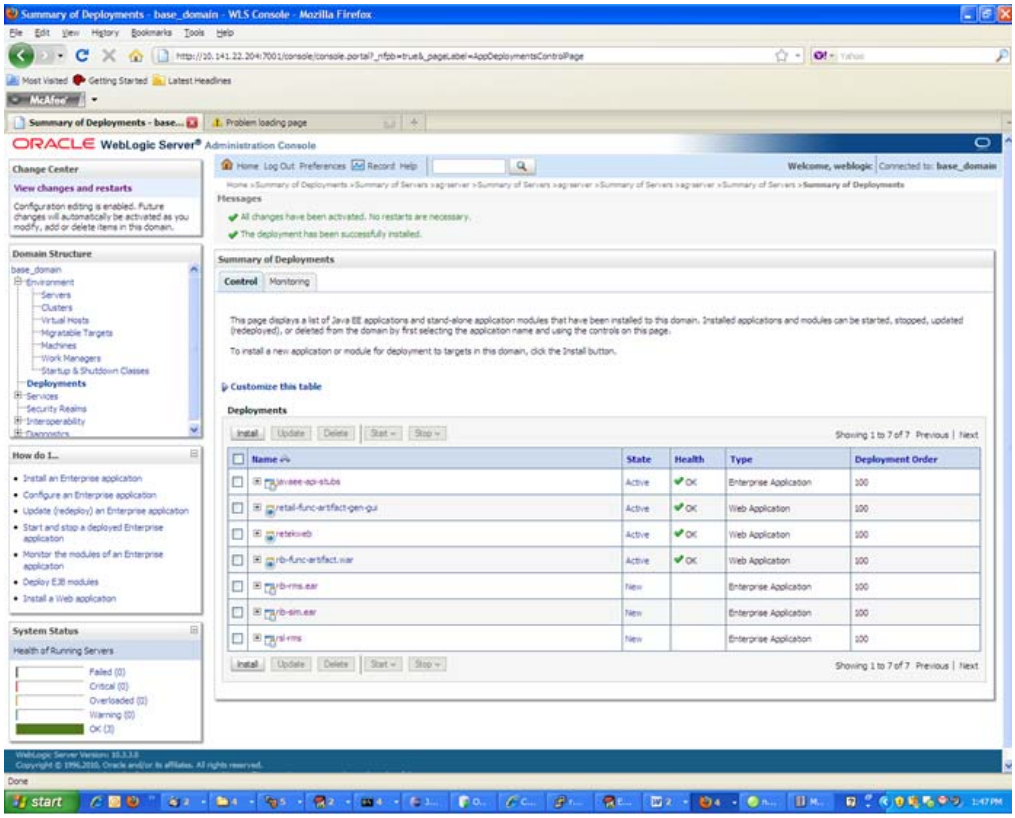


10. Select **Next** and move to **Review your choices and click Finish**.



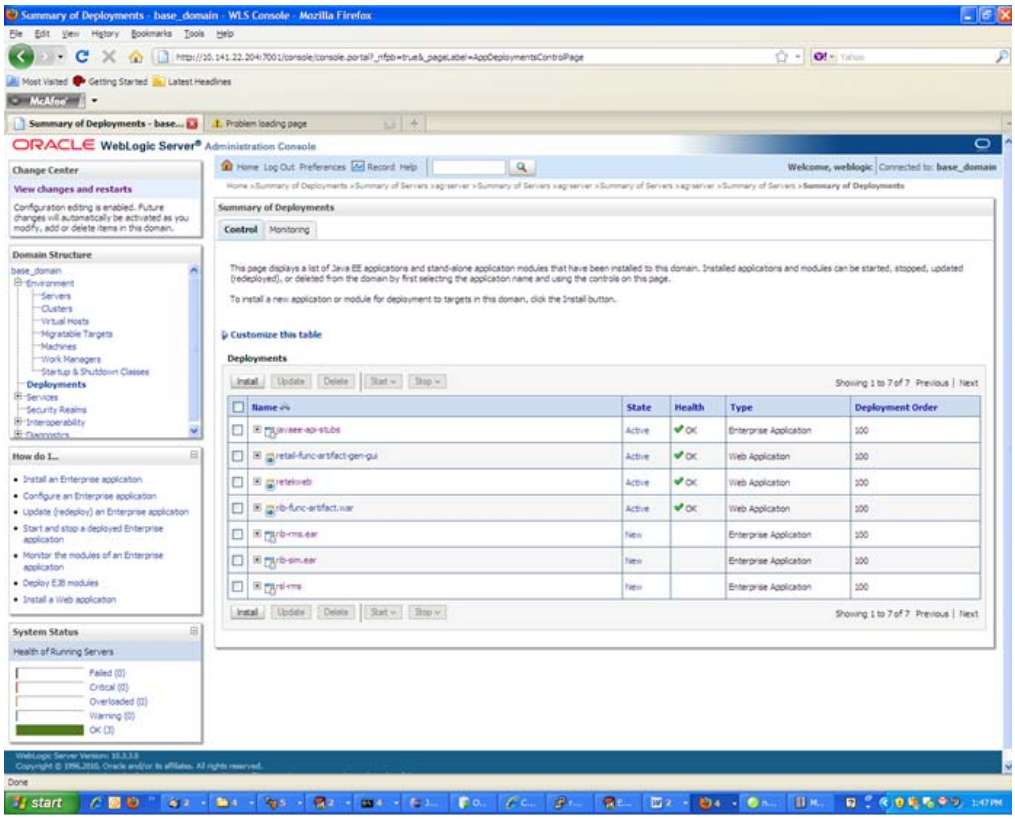
11. Select **No, I will review the configuration later**.

12. Click **Finish** to deploy the application.

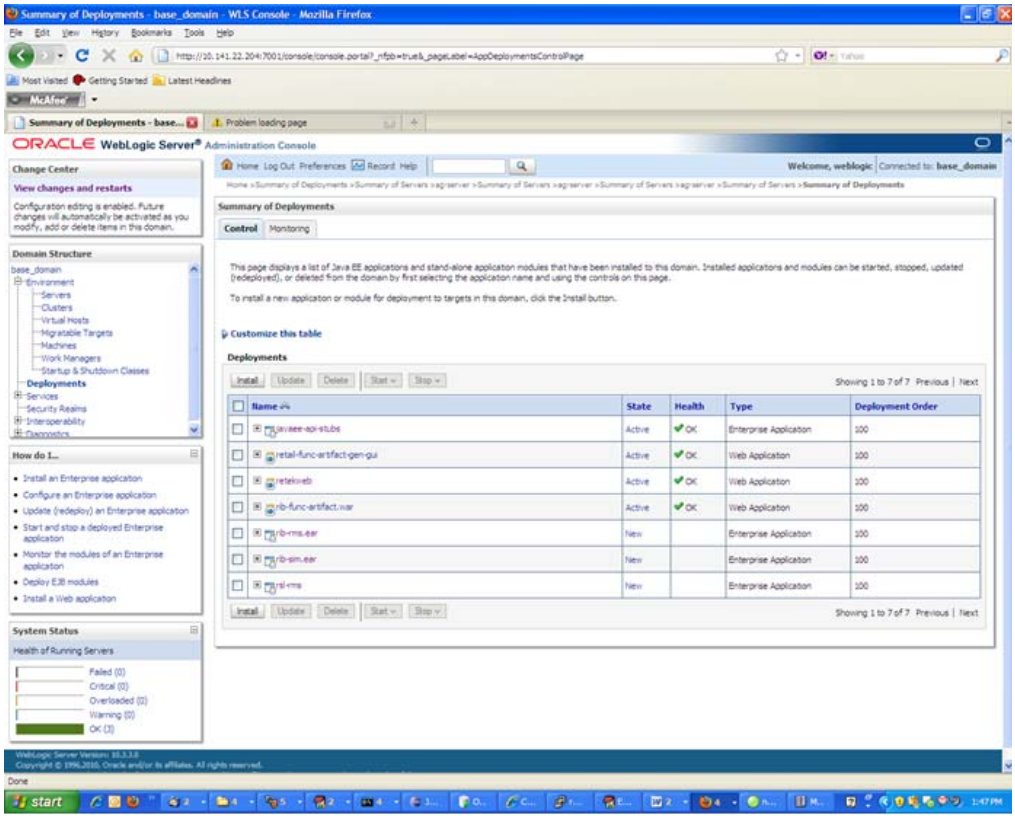


Verify the Artifact Generator Web Application

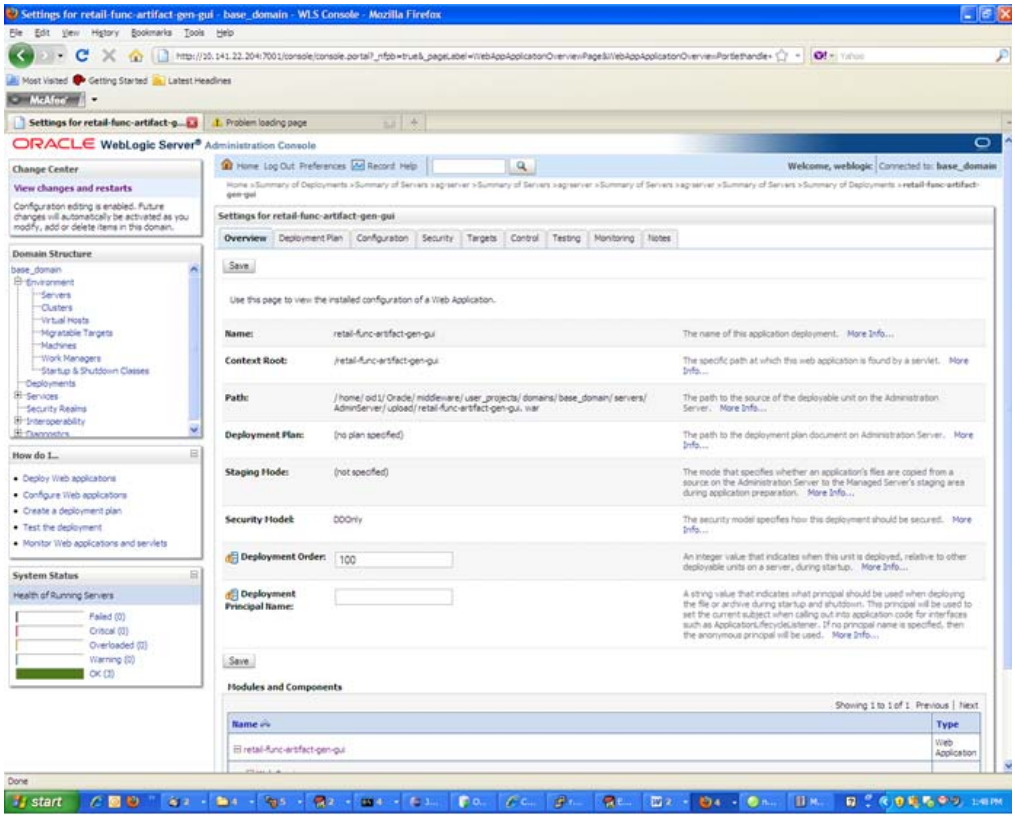
1. Navigate to the Deployments page.



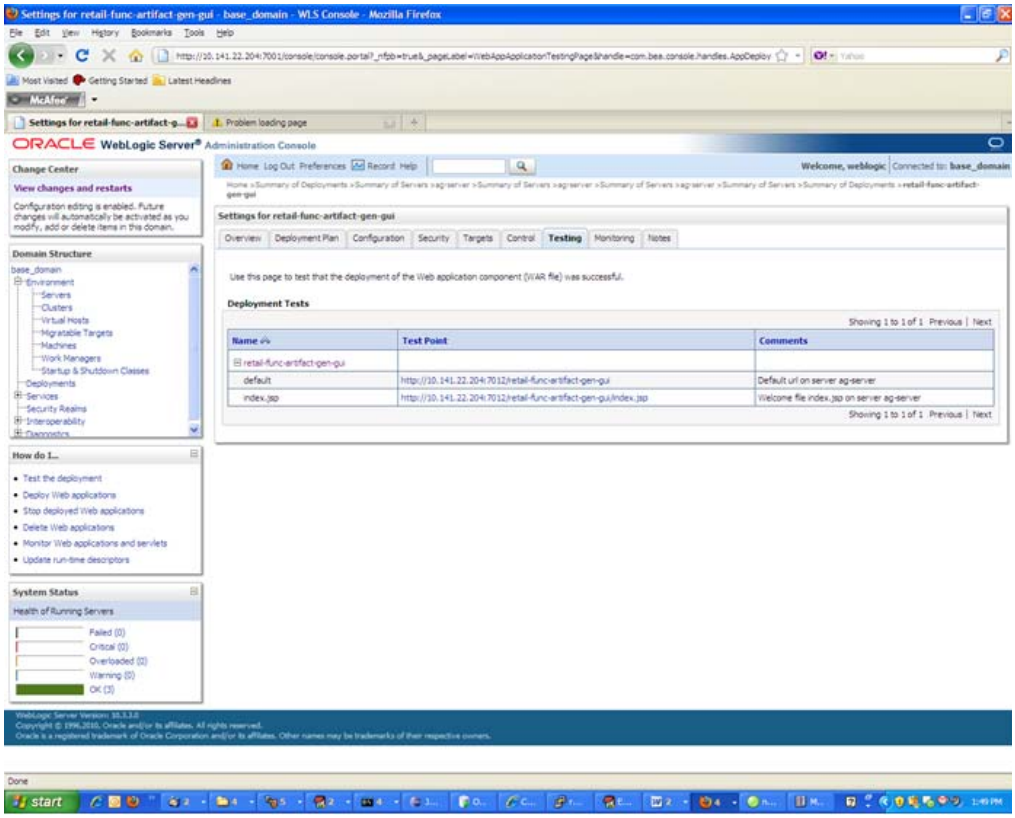
2. Locate the **retail-func-artifact** on the Summary of Deployments page.



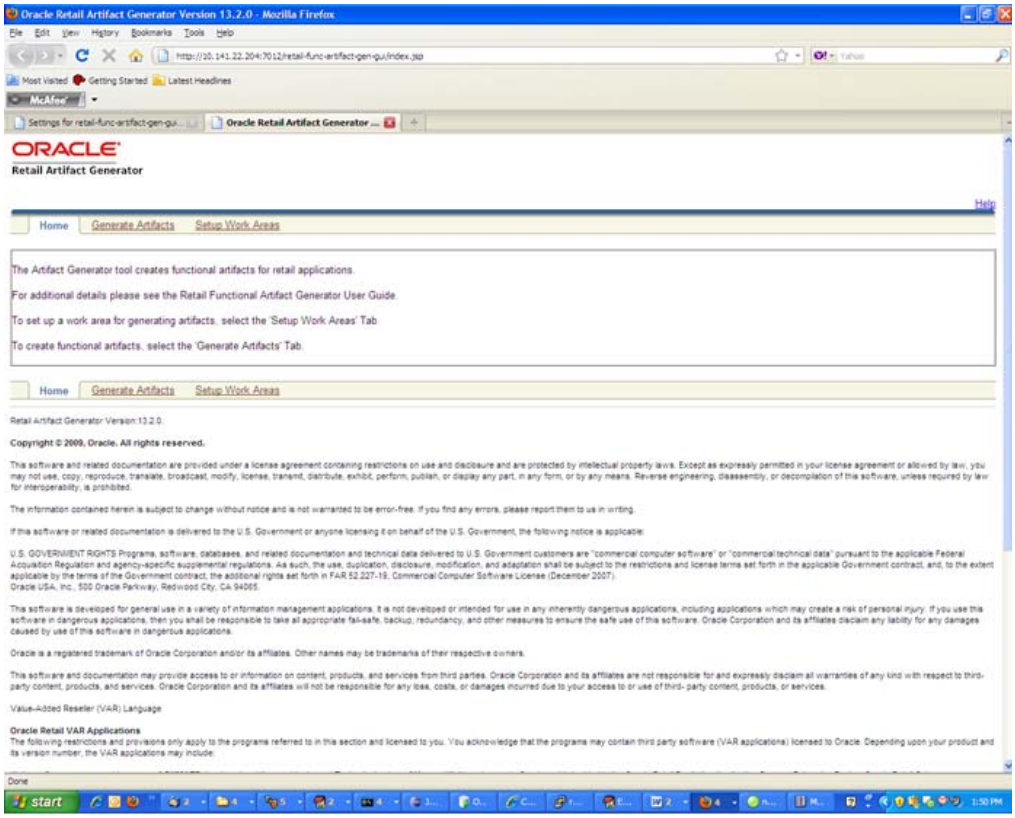
3. Click the name, **retail-func-artifact-gen-gui**, to move to the **Settings for the rib-func-artifact-gen-gui**.



4. Select the **Testing** tab.



5. Click **index.jsp URL** in the Test Point. The URL should open to the Retail Artifact Generator Home page.

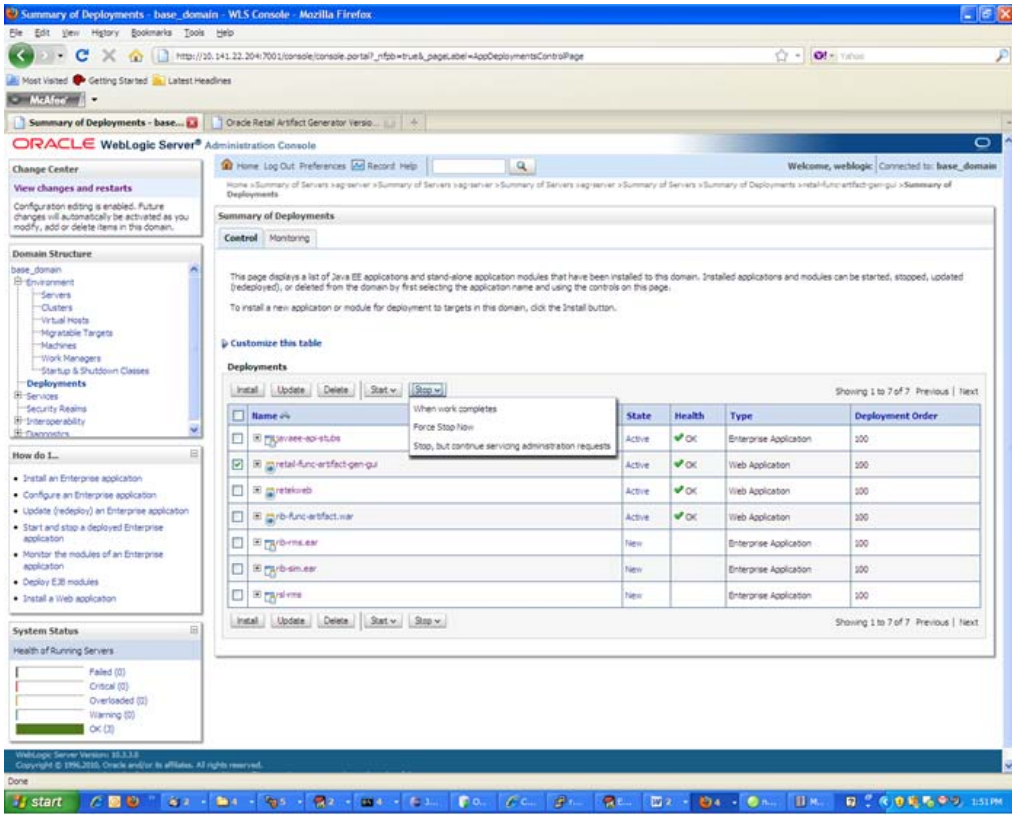


The installation is complete.

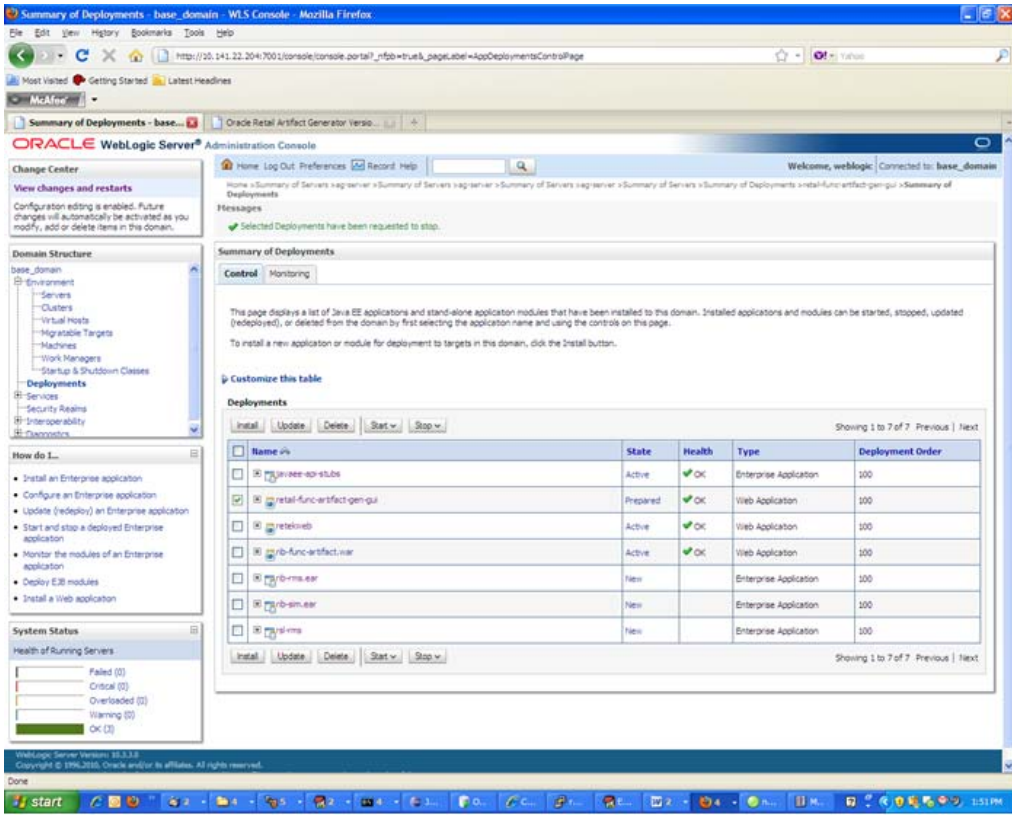
Redeploy the Application

If the retail-func-artifact-gen-gui application has already been deployed, follow these steps:

1. If the retail-func-artifact-gen-gui application is running, select **Stop** and **When Work Completes** or **Force Stop Now**, depending on the environment. The recommended option always is **When Work Completes.**



2. Select **Delete**. Now, the retail-func-artifact-gen-gui should not show on the Summary of Deployment page.



3. Return to the appropriate step in the section, Deploy the Artifact Generator Application."

B

Appendix: References

The following is a list of reference materials providing more information on the topics covered in this guide.

- [XML] Extensible Markup Language (XML) 1.0 (Second Edition), http://www.w3.org/TR/REC-xml
- [XMLSchema1] W3C Recommendation, XML Schema Part 1: Structures, http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/
- [XMLSchema2] W3C Recommendation, XML Schema Part 2: Datatypes, http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/