Dialogic®

JSR 309 Connector Software for Dialogic® PowerMedia™ XMS

User's Guide

October 2013 05-2708-005

www.dialogic.com

Copyright and Legal Notice

Table of Contents

Revision History

Revision	Release Date	Notes
05-2708-005	October 2013	Updates to support PowerMedia XMS Release 2.1. Global change: • Renamed this document from Quick Start Guide to User's Guide. System Requirements: • Updated to include support for OCCAS 5.1.0. Test Servlets: • Added, removed, and updated demos. • Added new section for Configuring the Test Servlets. • Updated instructions for demo prompts. Appendix: JSR 309 Connector Environment Setup: • Updated to include support for OCCAS 5.1.0.
05-2708-004	April 2013	Supported Features: • Added new supported configurations in Supported Configurations table. Test Servlets: • Added new section for DlgcReferenceConferenceDemo.
05-2708-003	March 2013	Supported Features: • Added new video file playback and video conference features.
05-2708-002	February 2013	Contents of the Distribution and Installation and Configuration: • Added new files to support Simple Logging Facade framework.

Revision	**Release Date**	**Notes**
05-2708-001	January 2013	Updates to support PowerMedia XMS Release 2.0. Global change: • Renamed Dialogic® PowerMedia™ Media Server Connector (PowerMedia MSC) to JSR 309 Connector Software for Dialogic® PowerMedia™ XMS (JSR 309 Connector). • Integrated content from Release Notes into this Quick Start Guide. Supported Features: • Added new bridge conference feature. System Requirements: • Added new section for hardware and software requirements along with supported platforms. Installation and Configuration: • Added new content for installation and configuration. Test Servlets: • Added new section for DlgcEarlyMediaBridgeDemo. Appendix: JSR 309 Connector Environment Setup: • Added new section for instructions on how to set up the JSR 309 Connector environment.
05-2708-001-01	October 2012	Updates to support PowerMedia MSC Release 3.0 Beta.
Version 0.3	April 2011	Updates to support PowerMedia MSC Release 1.0 Beta - Updates.
Version 0.2	December 2010	Updates to support PowerMedia MSC Release 1.0 Beta.
Version 0.1	October 2010	Initial release of this document.
Last modified: October 2013		

Refer to www.dialogic.com for product updates and for information about support policies, warranty information, and service offerings.

1. Welcome

This User's Guide is written for users of the JSR 309 Connector Software for Dialogic® PowerMedia™ XMS (also referred to herein as "JSR 309 Connector"), which is used in conjunction with the Dialogic® PowerMedia™ Extended Media Server (also referred to herein as "PowerMedia XMS" or "XMS").

This User's Guide describes the JSR 309 Connector, provides installation and configuration information, and describes the test servlets included in this release.

Assumptions

This User's Guide assumes that you have the following knowledge and experience:

- Familiarity with the Java Specification Request (JSR) 309 documentation version 1.0.
- Familiarity with Oracle Communications Converged Application Server (OCCAS) development and administration.
- Prior experience with JSR 289 (SIP Servlets).
- Prior experience with Java Platform Enterprise Edition (Java EE) development.
- Familiarity with PowerMedia XMS administration and configuration.

Related Information

See the following for more information:

- PowerMedia XMS datasheet at www.dialogic.com.
- PowerMedia XMS documentation at www.dialogic.com/manuals.
- Dialogic technical support at www.dialogic.com/support.
- JSR 309 documentation on the Java Community Process website at www.jcp.org.

2. Overview

This section provides an overview of the Java Specification Request (JSR) 309 and describes the JSR 309 Connector features and limitations.

Terminology

A brief description of terminology used in this document is provided for reference.

Servlet – A Java class which conforms to the Java Servlet Interface, by which a Java class may respond to HTTP requests. Applications using servlets may be packaged in a WAR file as a web application.

Java Specification Request (JSR) – A formal document created by members of the Java Community Process (JCP) that adds features and functionality to the Java platform.

JEE or J2EE – Java Platform, Enterprise Edition. A platform for server programming in the Java programming language.

Web Application Server – Application Server based on JEE or J2EE.

MSML – Media Server Markup Language.

OCCAS – Oracle Communications Converged Application Server.

Conference Control Leg – This terminology describes when a conference is created using a mixer component and the mixer is configured to create a conference control leg. A conference control leg requires the use of one connection resource from the Media Server.

No Conference Control Leg – This terminology describes when a conference is created using a mixer component and the mixer is configured to **not** create a conference control leg. In this case, the JSR 309 Connector uses the call legs to send conference configuration. Therefore, it does not require an extra Media Server resource for conference control.

JSR 309 Media Server Control API

Java Specification Request (JSR) 309 is a standard Java media server control API for multimedia application development. It provides a generic media server abstraction interface that is independent of the underlying media server control protocol. The multimedia applications, such as IVR, voice-mail, audio conferencing, and call center, are typically deployed in a SIP-based infrastructure.

The JSR 309 API provides three areas of functionality:

- Network Connection to establish media streams.
- Media Group functions such as to play, record, and control media content.
- Media Mixer functions to join media functions to a network connection so as to create conferences and call bridges.

For details on the JSR 309 API, see the documentation on the Java Community Process website at www.jcp.org.

For a list of JSR 309 API parameters supported by the JSR 309 Connector, see Supported Features.

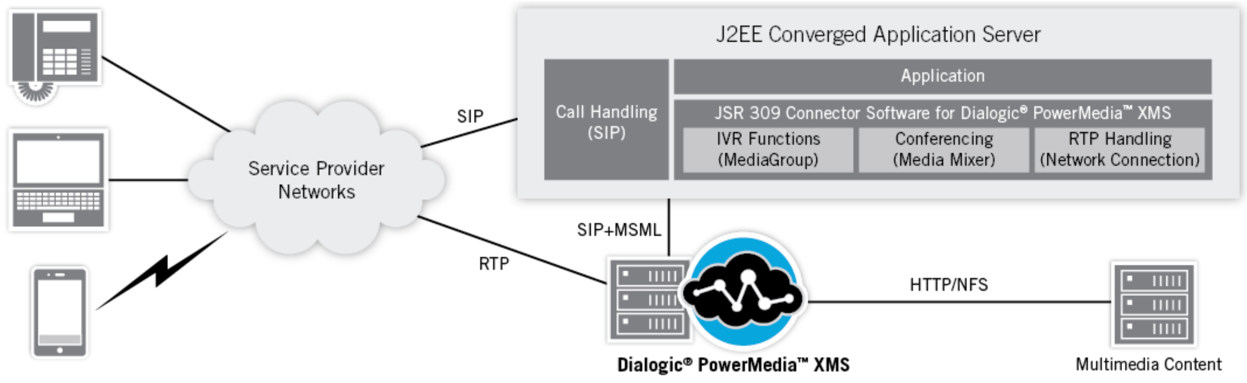
JSR 309 Connector

The JSR 309 Connector is the Dialogic implementation of the JSR 309 version 1.0 final specification. This software runs on application servers such as the OCCAS and enables a multimedia application on the application server to control the PowerMedia XMS using the JSR 309 API.

The JSR 309 Connector is designed to support an asynchronous programming model. Applications using the connector should be designed to interface using Listener objects for events on operation completion.

System Overview

The following figure illustrates the role of the JSR 309 Connector in a typical deployment.


J2EE Converged Application Server
Application
Call Handling (SIP)
JSR 309 Connector Software for Dialogic® PowerMedia™ XMS
IVR Functions (MediaGroup)
Conferencing (Media Mixer)
RTP Handling (Network Connection)
Service Provider Networks
SIP
SIP+MSML
RTP
HTTP/NFS
Dialogic® PowerMedia™ XMS
Multimedia Content

The following components are included in this figure:

J2EE Converged Application Server – Handles SIP call control and other aspects of real-time multimedia communications using a Java EE environment with JSR 289 support.

Application – Runs on the J2EE Converged Application Server. Examples of applications: IVR, conferencing, announcements, and call centers.

JSR 309 Connector – Software connector that enables the J2EE Converged Application Server to control PowerMedia XMS through JSR 309-compliant API calls.

PowerMedia XMS – Performs the multimedia operations required to establish and maintain real-time communications while providing a high-quality user experience.

External Servers – Used for storing and streaming multimedia content.

SIP Servlet API for Call Handling – SIP stack used to communicate with SIP compliant user agents. JSR 289 is the standard API servlet used by Converged Communication Application Servers.

3. Supported Features

The JSR 309 Connector is compatible with the JSR 309 Media Server Control API version 1.0.

The JSR 309 Connector supports the following functionality:

- Driver loading with driver property support
- Audio file playback
- Video file playback
- Audio recording
- Conference mixing (n-party mixing based on loudest talker)
- Bridge conference
- Basic prompt and collect
- Video conference
- Signal detection
- Redundant media servers (active and alternate)
- Serialization
- Cluster support
- Video streaming to network connections
- Media mixing with video layout
- Media play/record
- Media handling for WebRTC

Video streams, video layout, and video rendering components are not supported.

Limitations

The following high-level functionality is **not** implemented in the JSR 309 Connector:

Conference Limitations

Side-bar and coach-pupil conference to conference formats are not supported.

Runtime Controls

Various JSR 309 APIs accept Runtime Controls (RTC) as formal parameters. Overall, RTC is not supported, except for the following which are translated to barge-in functionality:

- RTC(SignalDetector.DETECTION_OF_ONE_SIGNAL, Recorder.STOP)
- RTC(MediaGroup.SIGDET_STOPPLAY, <n/a>)

Signal Generator

Signal Generator API is not supported.

CodecPolicy

The CodecPolicy implementation is not supported. Thus, no codec filtering is done at the connector level.

VxmlDialog

VxmlDialog is not supported.

Supported Configurations

The JSR 309 API has three core JSR API resource containers, NC, MG, and MX, that can be joined together in three different modes, SEND, RECV, and DUPLEX. However, not all combinations are supported.

NC refers to Network Connection and provides functionality to establish media streams. MG refers to Media Group and provides functionality to play, record, and control media content. MX refers to Media Mixer and provides functionality to join media functions to a network connection so as to create conferences and call bridges.

SEND means the media streams can flow from joiner to joinee only. RECV means the media streams can flow from joinee to joiner only. DUPLEX means the media streams can flow both ways.

The following table shows the supported configurations:

	NC	**MG**	**MX**
NC (DUPLEX)	YES	YES	YES
NC (SEND)	YES	NO	YES
NC (RECV)	YES	NO	YES
MG (DUPLEX)	YES	NO	YES
MG (SEND)	NO	NO	NO
MG (RECV)	NO	NO	NO
MX (DUPLEX)	YES	YES	NO
MX (SEND)	YES	NO	NO
MX (RECV)	YES	NO	NO

The following join combinations are the same and can be used to set up a full duplex connection. In the examples below, a Media Mixer is connected in full duplex to a Network Connection for play/record capability.

```
networkConnection.join(Direction.RECV, mixer)
mixer.join(Direction.SEND, networkConnection)

networkConnection.join(Direction.SEND, mixer)
mixer.join(Direction.RECV, networkConnection)

networkConnection.join(Direction.DUPLEX, mixer)
mixer.join(Direction.DUPLEX, networkConnection)
```

As per the JSR 309 Connector implementation for any above join combination, please make note that the Allocation Event is always sent to the Network Connection component.

4. System Requirements

Hardware and Software Requirements

This User's Guide assumes that you have the following systems in place before installing JSR 309 Connector:

- A working OCCAS 5.1.0 system for development and testing.
- A working PowerMedia XMS Release 2.1 system.
- SIP phones or soft clients.

Supported Platforms

The JSR 309 Connector was developed using the Java SDK version 1.6.0. The JSR 309 Connector has been deployed and tested on OCCAS 5.1.0.

5. Contents of the Distribution

This section lists and describes the files in the JSR 309 Connector distribution.

Distributed Files

The JSR 309 Connector distribution consists of the *JSR309_MSC3_BuildXX.tar* file. This package contains the following directories:

- DlgcJSR309 - consists of various required components for the JSR 309 Connector as well as demos.
- DlgcJSR309DemoPrompts - consists of media files included in the JSR 309 Connector distribution.

Note: <Release Package> refers to the *.tar* distribution package provided for this release.

JSR 309 Connector Files	Description
<Release Package>/DlgcJSR309/applications/ *dlgmsc_tests.war*	Directory that contains a deployable web archive that can be used to test the supported functionality. The WAR file implements several test servlets; see Test Servlets for more information.
<Release Package>/DlgcJSR309/lib/ *dlgmsc.jar* *jain-sip-sdp-1.2.91.jar* *jsr173_1.0_api.jar* *log4j-1.2.15.jar* *msmltypes.jar* *slf4j-api-1.7.2.jar* *slf4j-log4j12-1.7.2.jar* *xbean.jar*	Directory that contains the JSR 309 Connector implementation for PowerMedia XMS and required third-party software libraries.
<Release Package>/DlgcJSR309/properties/ *dlgcJSR309.properties* *dlgc_demos.properties* *log4j.properties*	Directory that contains the properties files used to set up configuration for JSR 309 Connector, the provided demos, and common logging facility.
<Release Package>/DlgcJSR309/sample-src/ *dlgmsc_tests.zip*	Directory that contains the test servlets source files. Test servlets are only provided for reference. Please read the copyright notice included with the test servlets.

Demo Prompts

The demo prompts, to run test servlets, consists of media files included in the JSR 309 Connector distribution. To install the demo prompts, see Installing the Demo Prompts.

6. Installation and Configuration

This section describes how to install and use the JSR 309 Connector.

For system requirements and supported platforms, see System Requirements.

Preparing the J2EE Converged Application Server

The JSR 309 Connector has been deployed and tested on OCCAS 5.1.0. If you are **not** familiar with OCCAS or how to set it up, refer to Appendix: JSR 309 Connector Environment Setup for guidance and detailed instructions.

Installing the JSR 309 Connector

Follow this procedure to get the application (WAR file) in an OCCAS environment to correctly load the JSR 309 Connector (*dlgmsc.jar*).

The JSR 309 Connector supports PowerMedia XMS via the MSML protocol.

Step 1

Extract the *JSR309_MSC3_BuildXX.tar* distribution package for this release which will create a `<Release Package>` directory.

Note: <Release Package> refers to the *.tar* distribution package provided for this release.

Copy all the *.jar* files from `<Release Package>/DlgcJSR309/lib` directory to the OCCAS `<Domain Location>/lib` directory.

Note: <Domain Location> refers to the domain path as specified during OCCAS installation.

JAR Files	Description
dlgmsc.jar *jain-sip-sdp-1.2.91.jar* *jsr173_1.0_api.jar* *log4j-1.2.15.jar* *msmltypes.jar* *slf4j-api-1.7.2.jar* *slf4j-log4j12-1.7.2.jar* *xbean.jar*	JSR 309 Connector implementation for PowerMedia XMS and required third-party software libraries.

Step 2

Set up the properties files.

JSR 309 Connector uses the following properties files:

- dlgcJSR309.properties - used to configure IP addresses and ports for the OCCAS environment using JSR 309 Connector and dedicated PowerMedia XMS platform.
- log4j.properties - used for logging using Simple Logging Facade framework implementation of log4j.

Note: You can configure logging in the *log4j.properties* file. By default, logging is configured to use INFO level and output to the Console as well as to a *dlgmsc.log* file.

Follow these steps to set up the properties file:

1. Copy the *dlgcJSR309.properties* and *log4j.properties* files from `<Release Package>/DlgcJSR309/properties` directory to the OCCAS `<Domain Location>/config` directory.
2. Edit the *dlgcJSR309.properties* file according to your OCCAS and PowerMedia XMS configuration. The changes will include the OCCAS IP address and SipServlet container port running the JSR 309 Connector as well as the PowerMedia XMS IP address and port:
   # JSR 309 Connector address information (typically same as the SipServlet container)
   ```
   connector.sip.address=xxx.xxx.xxx.xxx
   connector.sip.port=5060
   ```
   # PowerMedia XMS address information
   ```
   mediaserver.1.sip.address=xxx.xxx.xxx.xxx
   mediaserver.1.sip.port=5060
   ```
3. Edit the `<Domain Location>/bin/startWebLogic.sh` OCCAS startup script. Look for the following line:
   ```
   CLASSPATH="${SAVE_CLASSPATH}"
   ```
   Add the following lines directly after:
   ```
   export DLG_PROPERTY_FILE=${DOMAIN_HOME}/config/dlgcJSR309.properties
   LOG4J_OPTIONS="-Dlog4j.configuration=${DOMAIN_HOME}/config/log4j.properties"
   XQUERYPATH=/root/Oracle/Middleware/modules/features/weblogic.server.modules.xquery_10.3.6
   .0.jar
   CLASSPATH="${SAVE_CLASSPATH}:${ORCL_HOME}/server/modules/mscontrol.jar:${ORCL_HOME}/serve
   r/lib/jsr309-descriptor-binding.jar:${XQUERYPATH}"
   ```

4. Add the following line in OCCAS startup script to enable some of the relevant items and to disable serialization (highlighted in bold):
   ```
   ${LOG4J_OPTIONS} -Dlog4j.debug -Dwlss.local.serialization=false
   ```

   from:
   ```
   if [ "${WLS_REDIRECT_LOG}" = "" ] ; then
           echo "Starting WLS with line:"
           echo "${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} -Dweblogic.Name=${SERVER_NAME}
   -Djava.security.policy=${WL_HOME}/server/lib/weblogic.policy ${JAVA_OPTIONS}
   ${PROXY_SETTINGS} ${SERVER_CLASS}"
           ${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} -Dweblogic.Name=${SERVER_NAME} -
   Djava.security.policy=${WL_HOME}/server/lib/weblogic.policy ${JAVA_OPTIONS}
   ${PROXY_SETTINGS} ${SERVER_CLASS}
   else
           echo "Redirecting output from WLS window to ${WLS_REDIRECT_LOG}"
           ${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} -Dweblogic.Name=${SERVER_NAME} -
   Djava.security.policy=${WL_HOME}/server/lib/weblogic.policy ${JAVA_OPTIONS}
   ${PROXY_SETTINGS} ${SERVER_CLASS}  >"${WLS_REDIRECT_LOG}" 2>&1
   fi
   ```

   to:
   ```
   if [ "${WLS_REDIRECT_LOG}" = "" ] ; then
           echo "Starting WLS with line:"
           echo "${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} -Dweblogic.Name=${SERVER_NAME}
   ${LOG4J_OPTIONS} -Dlog4j.debug -Dwlss.local.serialization=false -
   Djava.security.policy=${WL_HOME}/server/lib/weblogic.policy ${JAVA_OPTIONS}
   ${PROXY_SETTINGS} ${SERVER_CLASS}"
           ${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} -Dweblogic.Name=${SERVER_NAME}
   ${LOG4J_OPTIONS} -Dlog4j.debug -Dwlss.local.serialization=false -
   Djava.security.policy=${WL_HOME}/server/lib/weblogic.policy ${JAVA_OPTIONS}
   ${PROXY_SETTINGS} ${SERVER_CLASS}
   else
           echo "Redirecting output from WLS window to ${WLS_REDIRECT_LOG}"
           ${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} -Dweblogic.Name=${SERVER_NAME}
   ${LOG4J_OPTIONS} -Dlog4j.debug -Dwlss.local.serialization=false -
   Djava.security.policy=${WL_HOME}/server/lib/weblogic.policy ${JAVA_OPTIONS}
   ${PROXY_SETTINGS} ${SERVER_CLASS}  >"${WLS_REDIRECT_LOG}" 2>&1
   ```

   For more details on disabling serialization, refer to Disabling Serialization.
5. Save the startup script file then restart the WebLogic Server.

7. Test Servlets

This section describes the test servlets (basic sample applications) and requirements for running test servlets in the JSR 309 Connector.

Test Servlets Overview

Test servlets are provided to illustrate the use of the JSR 309 Connector. These test servlets are included in the *dlgmsc_tests.war*.

Please read the copyright notice included with the test servlets.

Configuring the Test Servlets

1. Copy the *dlgmsc_tests.war* file from `<Release Package>/DlgcJSR309/applications` directory to the OCCAS `<Domain Location>/applications` directory.
2. Copy the *dlgc_demos.properties* file from `<Release Package>/DlgcJSR309/properties` directory to the OCCAS `<Domain Location>/applications` directory.
3. Edit the *startWebLogic.sh* file from `<Domain Location>/bin` directory and add the following line to point to demo properties file:

```
export DIALOGIC_DEMO_PROPERTY_FILE=${DOMAIN_HOME}/application/dlgc_demos.properties
```

See the following example with the line added to point to demo properties file (highlighted in bold):

```
export DIALOGIC_DEMO_PROPERTY_FILE=${DOMAIN_HOME}/application/dlgc_demos.properties
export DLG_PROPERTY_FILE=${DOMAIN_HOME}/config/dlgcJSR309.properties
LOG4J_OPTIONS="-Dlog4j.configuration=${DOMAIN_HOME}/config/log4j.properties"
XQUERYPATH=/root/Oracle/Middleware/modules/features/weblogic.server.modules.xquery_10.3.6.0
.jar
CLASSPATH="${SAVE_CLASSPATH}:${ORCL_HOME}/server/modules/mscontrol.jar:${ORCL_HOME}/server/
lib/jsr309-descriptor-binding.jar:${XQUERYPATH}"
```

Installing the Demo Prompts

Sample test servlets use custom demo prompts which need to be installed on dedicated PowerMedia XMS platform for this configuration. You can locate and install the demo prompts by performing the following:

1. Copy all the *.tar* files inside the `<Release Package>/DlgcJSR309DemoPrompts` directory to the PowerMedia XMS machine under `/var/lib/xms/media/en_US/verification` directory.
2. Untar each of the media files using the command; `tar -xvf <Filename>.tar`.

As an alternative method, you can install the demo prompts using the PowerMedia XMS web interface which allows you to upload one file at a time.

Note: When installing via the PowerMedia XMS web interface upload page, the directory structure must be identical to the unzip directory.

Running the Test Servlets

Using a SIP phone, special SIP URIs are used to initiate the test servlets. This information is available in the *sip.xml* file, which is included in the *dlgmsc_tests.war*. Verify that the media files have been installed on PowerMedia XMS.

You can locate and access the *sip.xml* file by performing the following:

1. Create a temp directory.
2. Copy the *dlgmsc_tests.war* and then run `jar –xvf dlgmsc_test.war` inside the temp directory.
3. Access the file contents located in `./WEB-INF/sip.xml`.

DlgcPlayerTest

This test servlet plays a PowerMedia XMS pre-set prompt.

Set up your SIP phone to point to the Web Application Server. Configure the SIP phone address (i.e. URI) to ***DlgcPlayerDemo***. Make sure that the Web Application Server is running the *dlgmsc_tests.war* application.

Using the demo property file, set the following:

```
player.test.prompt=
```

For example:

```
player.test.prompt=file:////var/lib/xms/media/en_US/verification/greeting.wav
```

The player will play this prompt. Make sure that the prompt file exists in the Media Server.

To test the application, dial the following:

```
DlgcPlayerDemo@<OCCAS5.1.0-IP-ADDRESS>
```

DlgcDtmfPromptAndCollectTest

This test servlet plays a prompt and collects DTMF digits.

Set up your SIP phone to point to the Web Application Server. Configure the SIP phone address (i.e. URI) to ***DlgcPromptCollectDemo***. Make sure that the Web Application Server is running the *dlgmsc_tests.war* application.

The DlgcPromptCollectDemo can be controlled using the demo property file as follows:

- The detectOnlyTest reads the number of signals property value and sends the pattern x (times number of signals). Note that no prompt is played. The following example generates a pattern to match of any five (5) DTMF entries:

  ```
  signalDetector.test=detectOnlyTest
  signalDetector.number_of_signals=5
  ```

- The detectPromptCollectTest plays a prompt and looks for a given pattern. It does not make use of the number of signals property.

  ```
  signalDetector.test=detectPromptCollectTest
  signalDetector.match_pattern=min=1;max=5;rtk=#
  ```

- The detectCollectWithPatternTest does not prompt the user and only uses the match_pattern.

```
signalDetector.test=detectCollectWithPatternTest
signalDetector.match_pattern=min=1;max=5;rtk=#
```

Note 1: You can configure the signal detector with the following properties (for example, the timeout values are based in milliseconds units):

```
signalDetector.initial_digit_timeout=5000
signalDetector.inter_digit_timeout=5000
signalDetector.max_duration=10000
```

Note 2: For the test that plays a prompt, you can control a loop (or how many times the test repeats the prompt and collect) by controlling the following property:

```
signalDetector.loopCounter=2
```

To test the application, dial the following:

```
DlgcPromptCollectDemo@<OCCAS5.1.0-IP-ADDRESS>
```

DlgcRecorderTest

This test servlet records a greeting.

Set up your SIP phone to point to the Web Application Server. Configure the SIP phone address (i.e. URI) to ***DlgcRecorderDemo***. Make sure that the Web Application Server is running the *dlgmsc_tests.war* application.

In the SIP phone, select your newly created test contact. You are prompted to record your greeting at the tone. After the tone, say your greeting, and enter #000 to play your greeting.

After the greeting is played back, the application completes by hanging up the phone. If you do not enter #000, the greeting continues to record until the timeout is reached.

The recording demo can be controlled in the demo property file by configuring the following record properties:

```
record.test.file=file:////tmp/recorder_jsr309_test_demo.ulaw
record.test.minDuration=6000
record.test.maxDuration=60000
record.test.initialTimeout=7000
record.test.finalTimeout=4000
record.test.silenceTerminationFlag=true
```

To test the application, dial the following:

```
DlgcRecorderDemo@<OCCAS5.1.0-IP-ADDRESS>
```

DlgcDtmfAsyncTest

This test servlet illustrates the asynchronous DTMF capabilities.

Set up your SIP phone to point to the Web Application Server. Configure the SIP phone address (i.e. URI) to ***DlgcAsyncDtmfDemo***. Make sure that the Web Application Server is running the *dlgmsc_tests.war* application.

In the SIP phone, select your newly created test contact. Notice that there are no prompts. You will be connected. The application waits for you to press DTMF digits. For each DTMF pressed, the application will receive the DTMF and print the collected DTMF to the screen.

Selecting the number 0 hangs up the connection.

To test the application, dial the following:

```
DlgcAsyncDtmfDemo@<OCCAS5.1.0-IP-ADDRESS>
```

Conference Demos

The following table depicts the conference demos that are delivered with JSR 309 Connector.

Demo Name	Functionality	Requires
JMCConferenceServlet	Demonstrates how to create and managed multiple conferences.	Media files needs to be installed in the PowerMedia XMS for menu to work.
DlgcAvConferenceDemo	Implements an advanced conference.	Media files needs to be installed in the PowerMedia XMS for menu to work. The demo property file must be configured.
DialogicBridgeConference	Shows how to create a two leg conference without using a mixer.	Media files needs to be installed in the PowerMedia XMS for menu to work.

JMCConferenceServlet

This test servlet illustrates how to create and managed multiple conferences using a mixer control leg. A mixer control leg is an extra SIP connection used to control the conference mixer.

Set up your SIP phone to point to the Web Application Server. Configure the SIP phone address (i.e. URI) to ***DlgcMultiConferenceDemo***. Make sure that the Web Application Server is running the *dlgmsc_tests.war* application.

In the SIP phone, select your newly created test conference contact. Notice that you will need at least two SIP phones. The first connection entering the conference will not hear anything until the other legs join in.

This conference performs the following:

1. Establishes a network connection and joins it with a media group.
2. Plays a prompt for new number (conference pin) and collects signals. Any pin number can be provided. Initially no conferences exist. Conferences are created as users call in and provide pin numbers. Callers will only hear other callers who provide the same pin number.
3. Creates a conference if a new pin is used, or adds a leg to an existing conference.

To test the application, dial the following:

```
DlgcMultiConferenceDemo@<OCCAS5.1.0-IP-ADDRESS>
```

DlgcReferenceConferenceWithOutBCallServlet

This test servlet illustrates how to implement an advanced conference that does not required a mixer control leg and the legs are connected directly into a conference without requiring any initial IVR functionality.

Set up your SIP phone to point to the Web Application Server. Configure the SIP phone address (i.e. URI) to ***DlgcAvConferenceDemo***. Make sure that the Web Application Server is running the *dlgmsc_tests.war* application.

In the SIP phone, select your newly created test conference contact. Notice that you will need at least two SIP phones. The first connection entering the conference will not hear anything until the other legs join in.

This simple conference performs the following:

1. Can join multiple legs into a conference.
2. Once in conference, the user can enter *00 to hear the conference menu and apply some of the menu options.

Menu Supported by DlgcAvConferenceDemo:

*00 - Plays announcement of menu options then goes back into conference

*01 - Allows the user to toggle between mute/unmute

*02 - Allows the user to unjoin the conference

*03 - Allows the user to rejoin the conference

*05 - Plays a song – Once the song completes, the leg goes back into conference

*06 - Plays announcement of "The conference size is"

*99 - Allows the user to stop any plays and return back to conference

*77 – Records the conference

*88 – Stops the conference recording

*44 – Makes an outbound call and joins the outbound call to the conference (requires configuration in the demo property file)

The demo can be controlled by configuring the following properties in the demo property file.

- Change the initial direction of legs by entering the following properties in the application demo property file:

```
demos.join.direction.leg1=<duplex,recv,send>
demos.join.direction.leg1=<duplex,recv,send>
demos.join.direction.leg1=<duplex,recv,send>
```

- To make an outbound call, make sure you have another accessible SIP phone that can receive calls and configure the following attributes:

```
application.sipTOA_Address.sip.address=146.152.245.3 # IP address of the SIP Phone
application.sipTOA_Port.sip.port=5060
application.sipTOA.sip.username=kapanga  #(any name will do)
application.early_media_bridge.sip.address=146.152.122.127 #OCCAS Addr
application.early_media_bridge.sip.port=5060 #OCCAS SIP PORT
```

- To run a video conference, make sure you set the following configuration:

  ```
  media.mixer.mode=AUDIO_VIDEO  # possible values AUDIO,AUDIO_VIDEO
  media.mixer.conf.video.size=VGA  # possible values VGA, 720p
  media.mixer.conf.recordfile=file:////tmp/confRecording  # recording the conference file
  full path. This also works for audio only conference.
  ```

  Note 1: To play the conference recording after the recording is completed, change the following attribute to point to the recording path.

  ```
  player.test.prompt=file:////tmp/confRecording then run DlgcPlayerDemo
  ```

To test the application, dial the following:

```
DlgcAvConferenceDemo@<OCCAS5.1.0-IP-ADDRESS>
```

DialogicBridgeConference

This test servlet illustrates how to implement a simple conference that does not required a mixer. That is two legs are directly joined into a conference.

Set up your SIP phone to point to the Web Application Server. Configure the SIP phone address (i.e. URI) to ***DlgcBridgeDemo***. Make sure that the Web Application Server is running the *dlgmsc_tests.war* application.

In the SIP phone, select your newly created test conference contact. Notice that you will need at least two SIP phones. The first connection entering the conference will not hear anything until the other legs join in.

This simple conference performs the following:

- It basically joins two calling legs into a simple conference.
- In order for the leg to enter the bridge, each leg must enter *03 after making the call.

To test the application, dial the following:

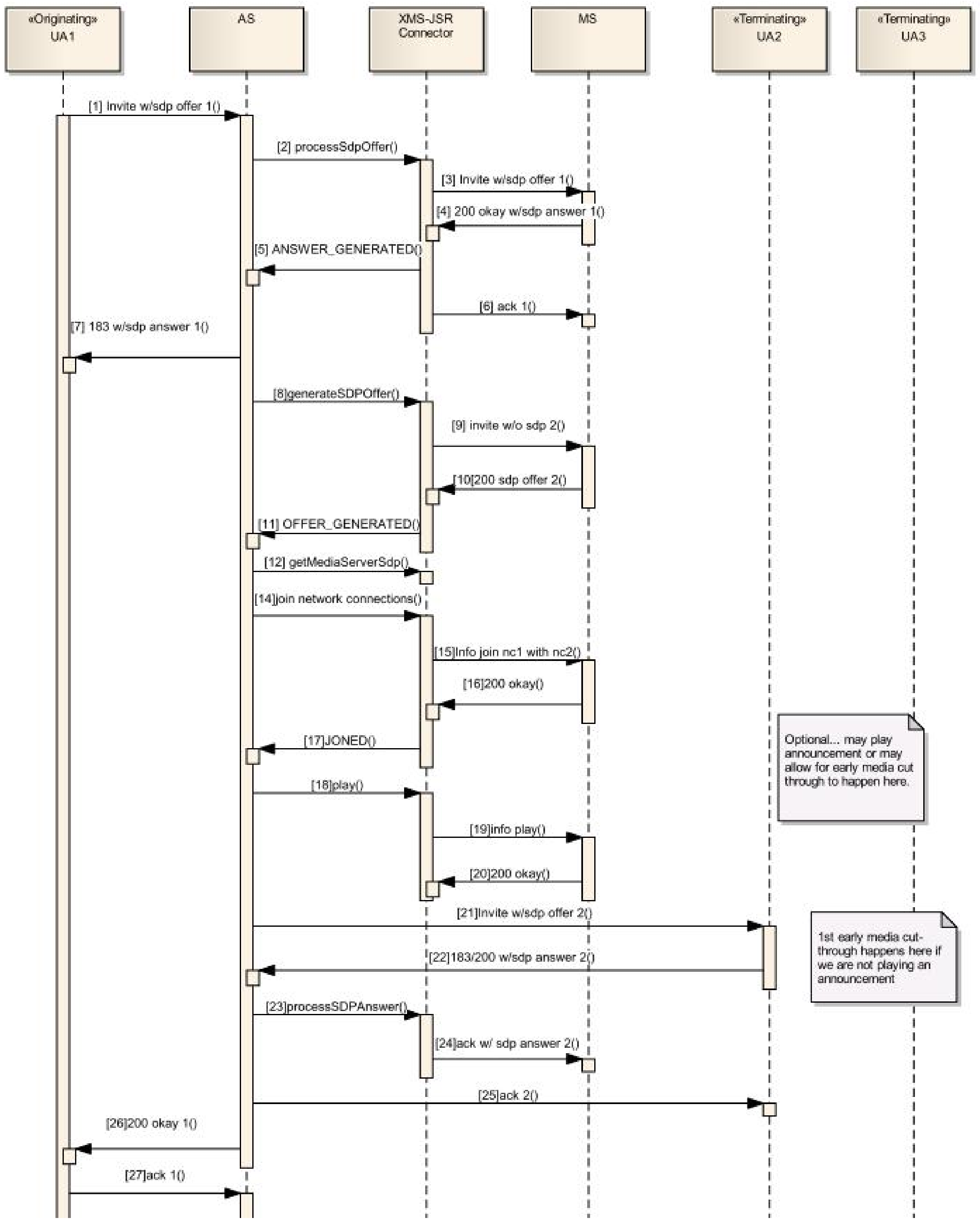
```
DlgcBridgeDemo@<OCCAS5.1.0-IP-ADDRESS>
```

DlgcEarlyMediaBridgeDemo

This test servlet is similar to the DialogicBridgeConference defined above, except that it simulates an early media scenario. For more details on using early media, refer to Early Media Guidelines.

Set up your SIP phone to point to the Web Application Server. Configure the SIP phone address (i.e. URI) to ***DlgcEarlyMediaBridgeDemo***. Make sure that the Web Application Server is running the *dlgmsc_tests.war* application.

The following sequence diagram illustrates DlgcEarlyMediaBridgeDemo:


«Originating» UA1
AS
XMS-JSR Connector
MS
«Terminating» UA2
«Terminating» UA3
[1] Invite w/sdp offer 1()
[2] processSdpOffer()
[3] Invite w/sdp offer 1()
[4] 200 okay w/sdp answer 1()
[5] ANSWER_GENERATED()
[6] ack 1()
[7] 183 w/sdp answer 1()
[8]generateSDPOffer()
[9] invite w/o sdp 2()
[10[200 sdp offer 2()
[11] OFFER_GENERATED()
[12] getMediaServerSdp()
[14]join network connections()
[15]Info join nc1 with nc2()
[16]200 okay()
[17]JONED()
[18]play()
[19]info play()
[20]200 okay()
[21]Invite w/sdp offer 2()
[22]183/200 w/sdp answer 2()
[23]processSDPAnswer()
[24]ack w/ sdp answer 2()
[25]ack 2()
[26]200 okay 1()
[27]ack 1()
Optional... may play announcement or may allow for early media cut through to happen here.
1st early media cut-through happens here if we are not playing an announcement

Menu Supported by DlgcEarlyMediaBridgeDemo:

*00 - Plays announcement of menu options

*77 - Plays announcement of how the demo works

*88 - Plays announcement informing the user if the application is in a bridge or mixer conference

*99 - Transfers the two call leg from a bridge conference to a full conference using a mixer.

Note: Once in a mixer conference, the test application does not allow you to go back to a bridge conference. The following property configuration must be set for this demo to work:

```
application.sipTOA_Address.sip.address=146.152.245.3 # IP address of the SIP phone
application.sipTOA_Port.sip.port=5060
application.sipTOA.sip.username=kapanga  #(any name will do)
application.early_media_bridge.sip.address=146.152.122.127 #OCCAS Addr
application.early_media_bridge.sip.port=5060 #OCCAS SIP PORT
```

To test the application, dial the following:

```
DlgcEarlyMediaBridgeDemo@<OCCAS5.1.0-IP-ADDRESS>
```

8. Developer Guidelines

This section provides application development guidelines for the JSR 309 Connector.

General Guidelines

Consider the following guidelines when designing and implementing your application:

- Note that the UNSOLICITED_OFFER_GENERATED event type handling is no longer needed when moving a leg from conference to IVR with the JSR 309 Connector.
    - The application must handle the UNSOLICITED_OFFER_GENERATED event type. This event type is provided in the SdpPortManagerEvent object indicating that the media server has requested a change in the SDP; therefore, the application must send re-INVITE to its user agent.
- If your application does not release connector resources, you will likely lose resources on your PowerMedia XMS.

Early Media Guidelines

Early media is supported by the JSR 289 and is enabled in the application.

To use early media, a media stream is established by the application while the SIP call leg is in 1xx state. Therefore, media will start flowing to the SIP endpoint before the call goes to the 2xx (connected) state.

A high-level early media call flow is as follows:

- From the JSR 289 (SIP side), the application waits for the INVITE with SDP offer.
- The application creates a Player using JSR 309 and passes in the SDP.
- The application sends the JSR 309 SDP answer back to the caller using a 183 response and using PRACK.
- The application instructs the PowerMedia XMS to play an announcement when the ACK (for the PRACK) is received.
- When the PowerMedia XMS has finished the announcement, it sends a BYE to the connector which signals the application.
- The application sends a 487 Call Terminated response to the caller rather than a 200 OK.

Redundant Media Servers Guidelines

The JSR 309 Connector supports redundant (active/alternate) media servers.

If the primary (active) media server becomes non-responsive, the connector software automatically routes a new invite request to the next available redundant (alternate) media server. If no media servers are available, all new calls will fail until one of the configured media servers becomes available.

If the media server fails, the application will be notified via the NETWORK_STREAM error.

You may configure one or more media servers. The first media server is considered the primary (active) one and all others are used as redundant (alternate) media servers. The mediaserver.redundancy parameter in the properties file controls this feature. For more information, see Installation and Configuration.

Configuring Network Time Protocol (NTP) for Accurate SIP Timers

In order for the SIP protocol stack to function properly, you must accurately synchronize system clocks on all engine and SIP data tier servers to a common time source, to within one or two milliseconds. Differences in system clock settings can cause a number of severe issues such as:

- SIP timers firing prematurely on servers with the fast clock settings.
- Poor distribution of timer processing in the engine tier. For example, one engine tier server may process all expired timers, whereas other engine tier servers process no timers.

You should use a Network Time Protocol (NTP) client or daemon on each OCCAS instance and synchronize to a common NTP server.

Because the initial T1 timer value of 500 milliseconds controls the retransmission interval for INVITE request and responses, and also sets the initial values of other timers, even small differences in system clock settings can cause improper SIP protocol behavior. For example, an engine tier server with a system clock 250 milliseconds faster than other servers will process more expired timers than other engine tier servers, will cause retransmits to begin in half the allotted time, and may force messages to timeout prematurely.

Configuring the Application Server

Using Serialization

Java object serialization is used to support replication of Java objects to another process. Serialization enables an application in a distributed or clustered environment to support application replication. By default serialization is turned on in the JSR 309 Connector.

To use serialization in your application, follow these instructions:

1. Add the **transient** keyword to the JSR 309 MsControlFactory, Driver, and Listener instance attributes. See Example A.

   Note that the JSR 309 specification defines MsControlFactory, Driver, and Listener as non-serializable; therefore, the application is required to use the transient keyword. The application servlet that contains these attributes must implement the Java serializable interface.
2. All JSR 309 Listener classes must implement the serializable interface. See Example B.

See additional guidelines in Guidelines for Using Serialization.

Disabling Serialization

Turning off serialization can improve performance. If your application does not need to participate in replication, you can disable local serialization by the container. To do so, add the following parameter to the OCCAS startup script:

```
-Dwlss.local.serialization=false
```

Following is the relevant portion of a startup script that has been modified to disable local serialization by the container.

```
if "%WLS_REDIRECT_LOG%"=="" (

       echo Starting WLS with line:

       echo %JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS%
-Dwlss.local.serialization=false -Dweblogic.Name=%SERVER_NAME%
-Djava.security.policy=%WL_HOME%\server\lib\weblogic.policy  %PROXY_SETTINGS% %SERVER_CLASS%

       %JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS% -Dwlss.local.serialization=false
-Dweblogic.Name=%SERVER_NAME% -Djava.security.policy=%WL_HOME%\server\lib\weblogic.policy
%PROXY_SETTINGS% %SERVER_CLASS%

) else (

       echo Redirecting output from WLS window to %WLS_REDIRECT_LOG%

       %JAVA_HOME%\bin\java %JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS% -Dwlss.local.serialization=false
-Dweblogic.Name=%SERVER_NAME% -Djava.security.policy=%WL_HOME%\server\lib\weblogic.policy
%PROXY_SETTINGS% %SERVER_CLASS%  >"%WLS_REDIRECT_LOG%" 2>&1
)
```

Guidelines for Using Serialization

Follow these rules and guidelines for using serialization in your user application:

1. Avoid using instance and static variables in read and write mode because different instances may exist on different JVMs. In other words, static and transient attributes are not replicated.
2. All non-transient fields must be serializable (or primitive).
3. Always explicitly define the following in your serialized classes. For example:

   ```
   private static final long serialVersionUID = <long integer
   ```

   You can use eclipse to generate a unique UID.
4. All classes in a hierarchy must be serializable. If the subclass implements serialization and the superclass does not, the subclass must take care of the serialization of the superclass attributes.
5. readObject implementations always start by calling the defaultReadObject() method.
6. The de-serialization process typically calls the readObject method. If you overwrite the readObject method, you must first call defaultReadObject() before adding any other instantiation to your object.
7. There is a performance cost to storing large object graphs in a Sip session or HTTP session. Large applications require using persistent sessions. Sessions must be read by the servlet whenever the object graph is used and rewritten whenever the object graph is updated.
8. Avoid using java.io.* because the files may not exist on all application servers. Instead use getResourceAsStream.
9. Avoid storing values in a ServletContext. A ServletContext is not serializable and also the different instances may exist in different JVMs.

10. Use SipSession setAttribute method to store the session state and use SipSession getAttribute method to restore it. Note that the setAttribute and getAttribute methods respectively lock/unlock the call state. Remember that setAttribute and getAttribute are expensive operations so use these methods judiciously.
11. In OCCAS the SAS.doAction() method also locks/unlocks the call state.
12. When using Maps to store serializable information, both keys and data must be serializable. Note the following rule:
    - Do not use an object reference as a key, because the object hashed ID changes between the serialization and de-serialization operations. This means that searching the Map may fail.

Configuration for a Distributed Environment

If your application needs to run in a distributed environment, you must add the <distributable/> tag in your application's *sip.xml* and make sure that serialization is set to true (that is, it is not set to false) in the OCCAS startup script.

Example Code

Example A

This example illustrates serialization best practices. The text in bold illustrates various rules and guidelines discussed in Using Serialization and Guidelines for Using Serialization.

```
public abstract class DlgcTest extends SipServlet implements Serializable
{
     private static final long serialVersionUID = -6161452986725649032L;
    //Since this value is a constant no need to serialize it; thus static is used
     protected static String dlgcDriverName = "com.dialogic.dlg309";
    //Note use of transient keyword: Factory, Listener, and Driver must be transient
     transient protected MsControlFactory mscFactory;
     transient protected DlgcSdpPortEventListener speListener;
     transient protected Driver dlgcDriver =null;

     private static Logger log = Logger.getLogger(DlgcTest.class);

     @Override
     public void init(ServletConfig cfg)
        throws ServletException
     {
        super.init(cfg);
        try
        {
           dlgcDriver = DriverManager.getDriver(dlgcDriverName);
           mscFactory = dlgcDriver.getFactory(null);
           speListener = new DlgcSdpPortEventListener();
        }
        catch (Exception e)
        {
           throw new ServletException(e);
        }
     }

     @Override
     public void doInvite(final SipServletRequest req)
        throws ServletException, IOException
     {
        log.info("doInvite");
        NetworkConnection networkConnection = null;
        SipSession sipSession = req.getSession();
        if (req.isInitial())
        {
           // We have a new call.
           try
           {
              MediaSession mediaSession = mscFactory.createMediaSession();
              networkConnection = mediaSession.createNetworkConnection(NetworkConnection.BASIC);
              sipSession.setAttribute("MEDIA_SESSION", mediaSession);
              sipSession.setAttribute("NETWORK_CONNECTION", networkConnection);

             //Even though this is not part of the best practices section
             //typically the application must store the UA SIP Session in the Media Session.
             //This is done because the UA SIP Session may be required inside a given Listener.
             //For example, the Listener may be required to send a Bye message to the UA.
              mediaSession.setAttribute("SIP_SESSION", sipSession);
              networkConnection.getSdpPortManager().addListener(speListener);
           }
           catch (MsControlException e)
           {
              req.createResponse(SipServletResponse.SC_SERVICE_UNAVAILABLE).send();
           }
        }
    .
    .
    .
      }
}
```

Example B

This example illustrates serialization best practices. The text in bold demonstrates the rule that all Listener classes must implement the serializable interface, as discussed in Using Serialization.

```
private class PlayerEventListener implements MediaEventListener<PlayerEvent>,  Serializable
{
   private static final long serialVersionUID = -50247033407045994L;
   @Override
   public void onEvent(PlayerEvent event)
   {
      log.info("PlayerEventListener::onEvent()");
      log.info("   EVENT TYPE : " + event.getEventType());
      …
      //This shows how the UA sip session is used in the Listener to send the bye to the phone UA
      MediaSession mediaSession = event.getSource().getMediaSession();
      SipSession session = (SipSession) mediaSession.getAttribute("SIP_SESSION");
      log.debug("Calling BYE..Hanging Phone");
      SipServletRequest request = session.createRequest("BYE");
      try
      {
         request.send();
      }
      catch (IOException e)
      {
         e.printStackTrace();
      }
      .
      .
      .
   }
 }
}
```

9. Troubleshooting

This section provides basic troubleshooting techniques for the JSR 309 Connector.

Logging

The JSR 309 Connector and sample applications generate log output to the *dlgmsc.log*. The default logging level is set to INFO.

You may also need to enable logging for SIP messages in the container so that the incoming requests that trigger the servlets are captured. You can enable SIP message logging through the OCCAS administration console.

SIP Errors

If the PowerMedia XMS returns "503 Service Unavailable", make sure your network is correctly set up by performing the following actions:

- Verify the available PowerMedia XMS licenses.
- Check the `/etc/hosts` file configuration.
- Make sure application properties file (i.e. *user1_dlgmsc.properties*) is referencing the appropriate PowerMedia XMS and OCCAS IP address and ports.

10. Appendix: JSR 309 Connector Environment Setup

This section describes, in detail, how to set up the JSR 309 Connector environment:

- Installing and Configuring the OCCAS
- Installing the JSR 309 Connector
- Installing and Configuring the Test Servlets
- Running the Test Servlets

For system requirements and supported platforms, see System Requirements.

This section does not go into details of OCCAS, but simply will help build an OCCAS system which could be used for verification purposes.

Steps to complete on OS level include:

- Install MySQL database
- Set up SSH for file configuration
- Enable NTP (Network Time Protocol)
- Enable ports in firewall (if applicable)

  Note: The ports that are required to be enabled in the firewall include SIP, TCP, and UDP ports 5060 and 5061 as well as 7001 which will be used by OCCAS.

If you need more details on OCCAS, refer to the OCCAS installation instructions available from www.oracle.com.

Installing and Configuring the OCCAS

This section describes the installation and configuration instructions for OCCAS 5.1.0. This section illustrates how to install and configure OCCAS in order to be able to go to the next step of Installing the JSR 309 Connector.

Note: If you are familiar with OCCAS or planning to deploy on an existing OCCAS setup, proceed to Installing the JSR 309 Connector.

Here are some highlights of the necessary steps:

- Pre-Installation Setup
- JDK Setup
- OCCAS Installation
- OCCAS Configuration
- OCCAS Startup
- Firewall Configuration
- OCCAS Verification

Pre-Installation Setup

Modify the `/etc/hosts` file:

```
xxx.xxx.xxx.xxx 'hostname'
```

Note: This must be the first line in the `/etc/hosts` file. If not, you might encounter "503 Service Unavailable" error.

Run the following command at the prompt:

```
service network restart
```

JDK Setup

Download the JDK *.rpm* file from www.oracle.com. For example, the following setup is based on *jdk-7u25-linux-x64.rpm* file.

Install the JDK *.rpm* file:

```
rpm -ivh jdk-7u25-linux-x64.rpm
```

Modify the *.bashrc* file and add the following line:

```
Export JAVA_HOME=/usr/java/jdk1.7.0_25
```

Save the *.bashrc* file and then execute:

```
source ./bashrc
```

This will take the changes into effect on the system.

Move the *occas5.1.0.zip* file into the root directory.

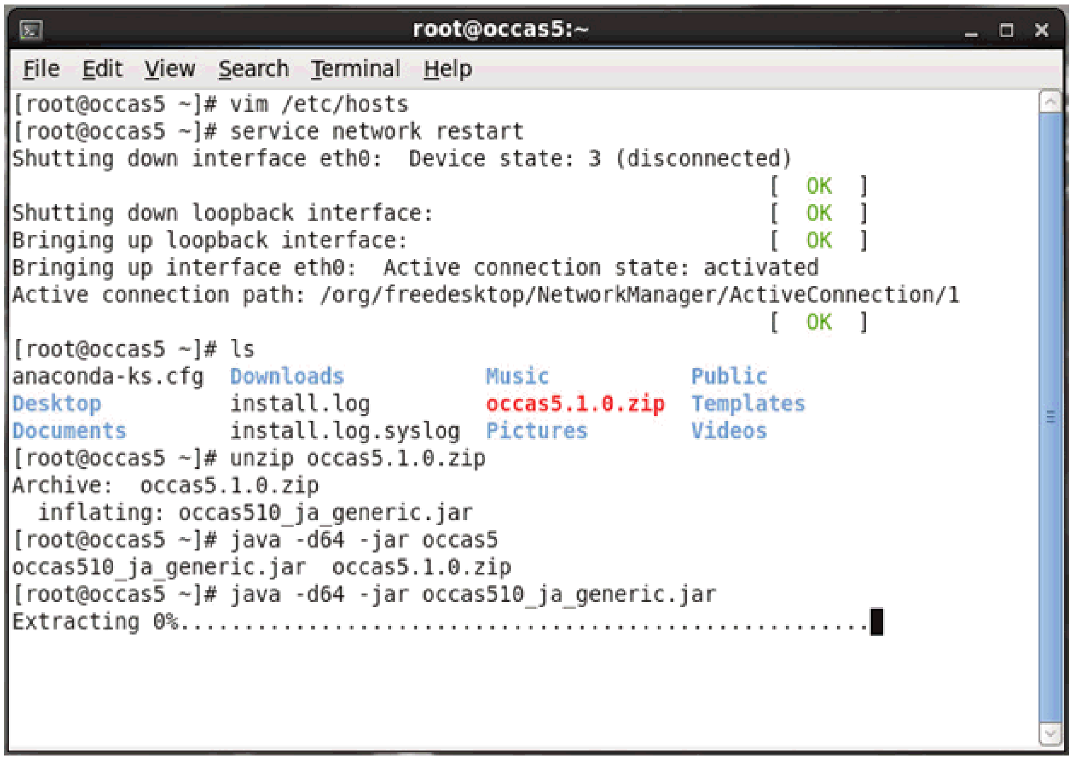
Unzip the *occas5.1.0.zip* file then proceed to OCCAS Installation.

OCCAS Installation

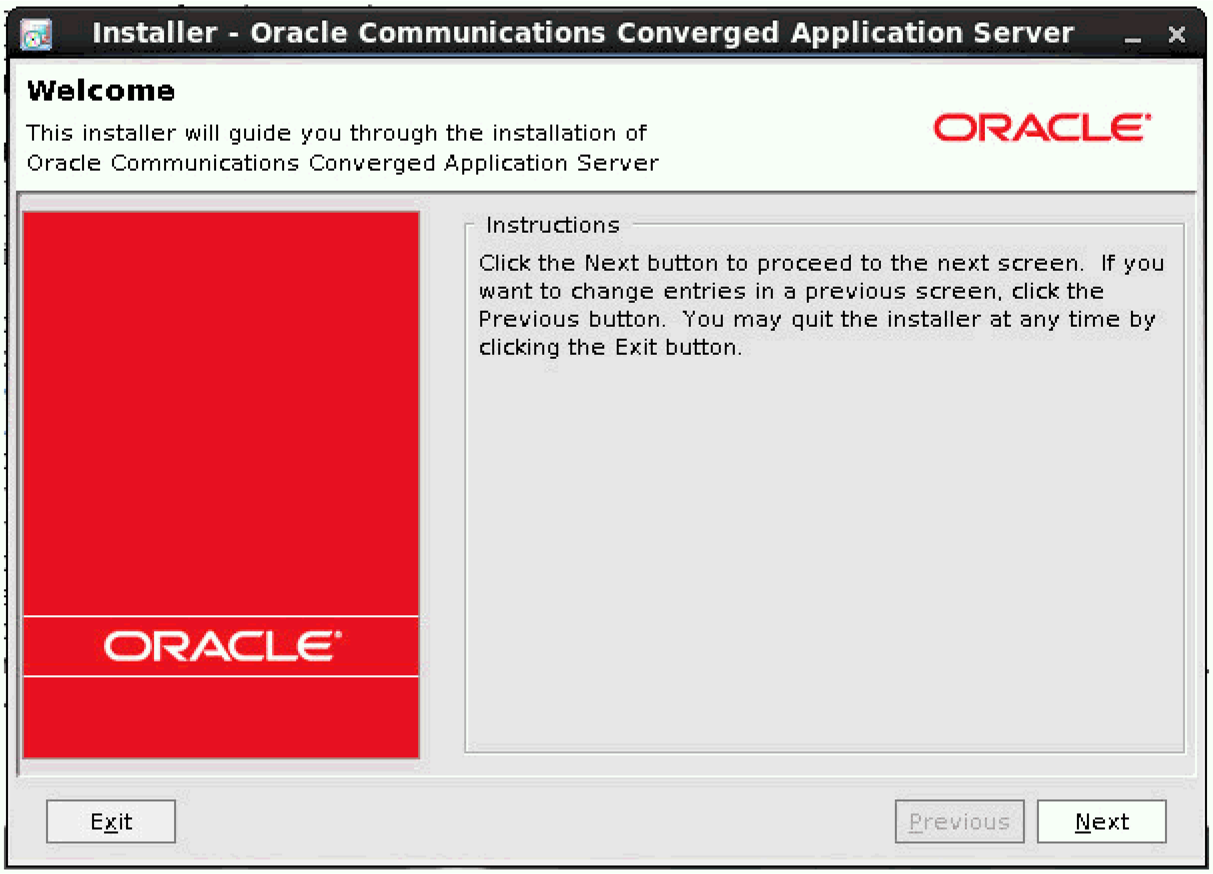
Install the *occas510_ja_generic.jar* file.

```
java -d64 -jar occas510_ja_generic.jar
```

Note: You may need to change permissions in order for the file to be executable.



```
[root@occas5 ~]# vim /etc/hosts
[root@occas5 ~]# service network restart
Shutting down interface eth0:  Device state: 3 (disconnected)
                                                           [  OK  ]
Shutting down loopback interface:                          [  OK  ]
Bringing up loopback interface:                            [  OK  ]
Bringing up interface eth0:  Active connection state: activated
Active connection path: /org/freedesktop/NetworkManager/ActiveConnection/1
                                                           [  OK  ]
[root@occas5 ~]# ls
anaconda-ks.cfg  Downloads          Music           Public
Desktop          install.log        occas5.1.0.zip  Templates
Documents        install.log.syslog Pictures        Videos
[root@occas5 ~]# unzip occas5.1.0.zip
Archive:  occas5.1.0.zip
  inflating: occas510_ja_generic.jar
[root@occas5 ~]# java -d64 -jar occas5
occas510_ja_generic.jar  occas5.1.0.zip
[root@occas5 ~]# java -d64 -jar occas510_ja_generic.jar
Extracting 0%.....................................................
```


Installer - Oracle Communications Converged Application Server
ORACLE
Exit
Previous
Next

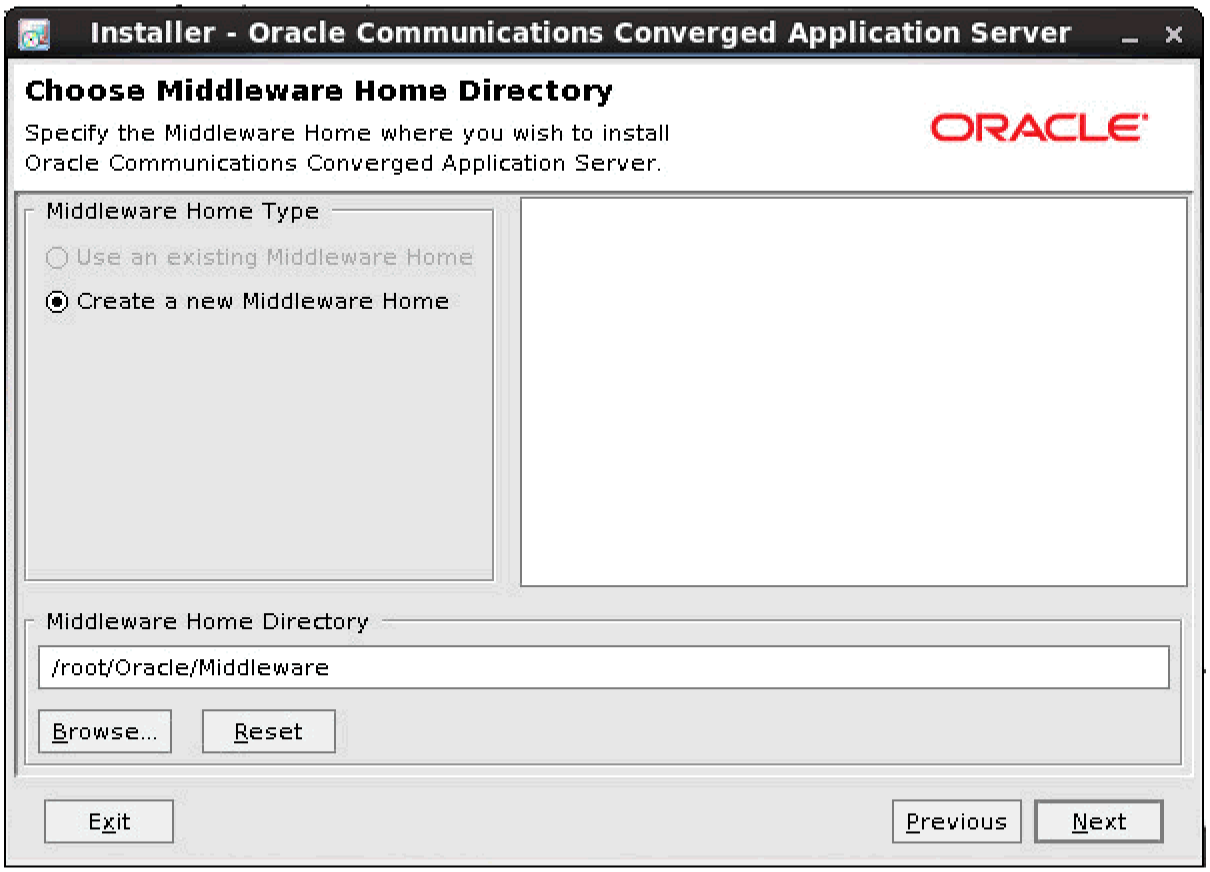
Welcome

This installer will guide you through the installation of Oracle Communications Converged Application Server

Instructions

Click the Next button to proceed to the next screen. If you want to change entries in a previous screen, click the Previous button. You may quit the installer at any time by clicking the Exit button.

Click on **Next.**


Installer - Oracle Communications Converged Application Server

Choose Middleware Home Directory

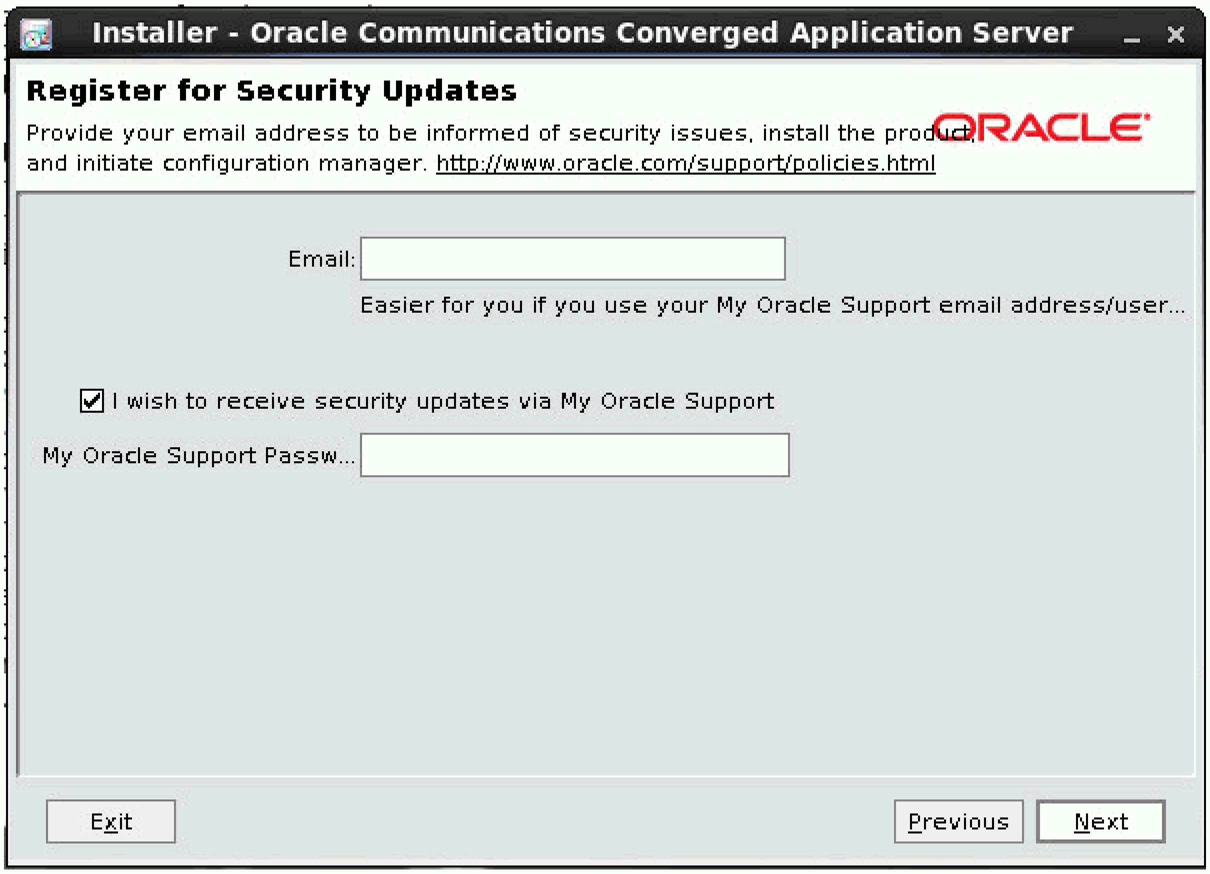
Specify the Middleware Home where you wish to install Oracle Communications Converged Application Server.

Middleware Home Type

- ○ Use an existing Middleware Home
- ● Create a new Middleware Home

Middleware Home Directory: /root/Oracle/Middleware

Select **Create a new Middleware Home** then click on **Next**.


Installer - Oracle Communications Converged Application Server

Register for Security Updates

Provide your email address to be informed of security issues, install the product and initiate configuration manager. http://www.oracle.com/support/policies.html

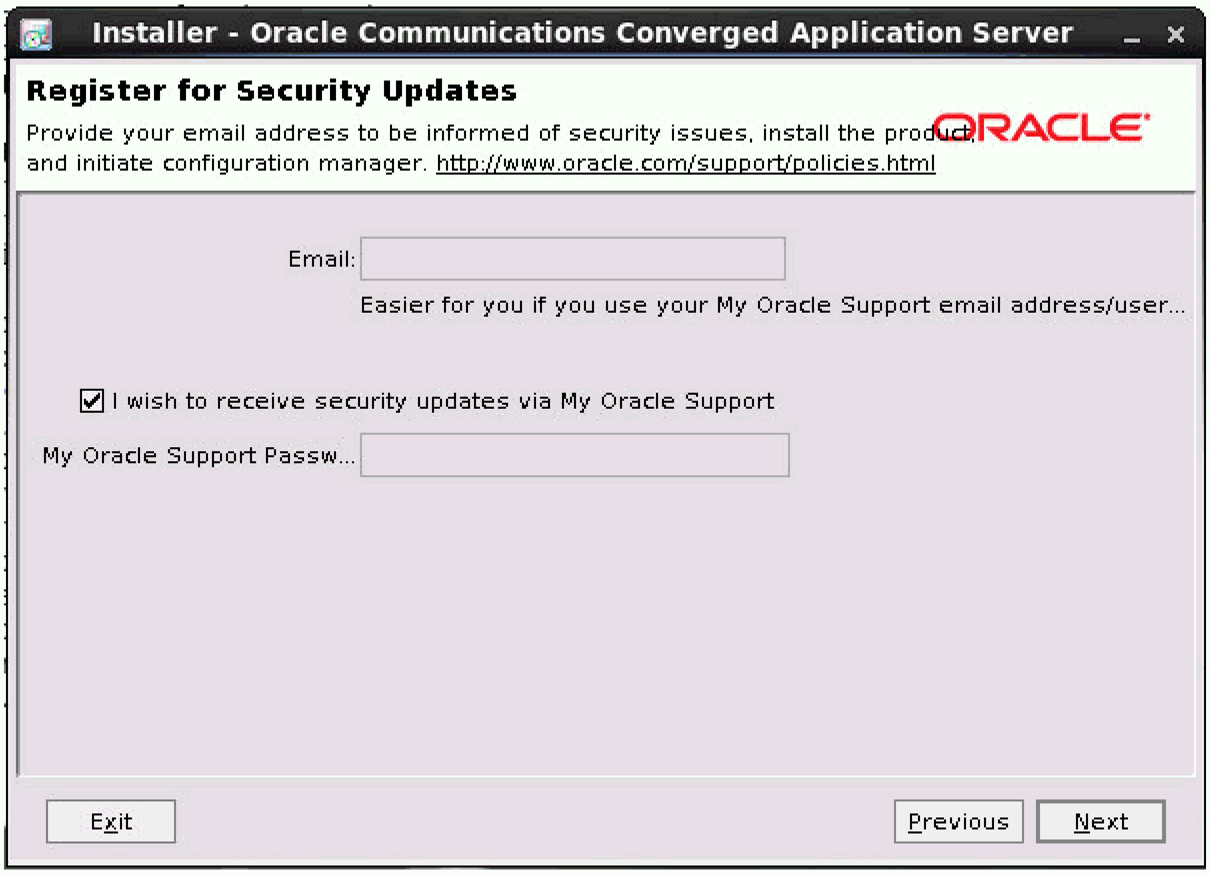
Email:

Easier for you if you use your My Oracle Support email address/user...

☑ I wish to receive security updates via My Oracle Support

My Oracle Support Passw...

De-select security updates (unless you have an account with Oracle) then click on **Next**.



Installer - Oracle Communications Converged Application Server

Register for Security Updates

Provide your email address to be informed of security issues, install the product and initiate configuration manager. http://www.oracle.com/support/policies.html

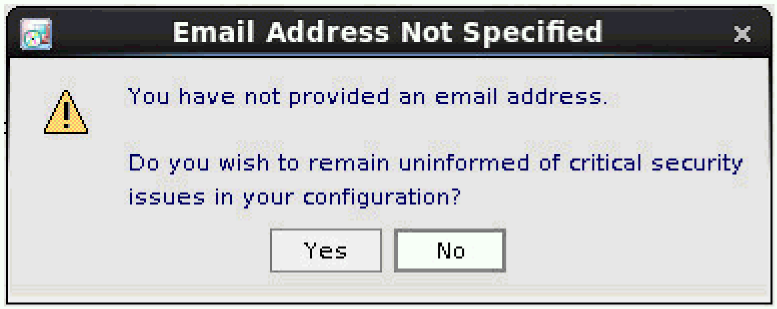
Email:

Easier for you if you use your My Oracle Support email address/user...

☑ I wish to receive security updates via My Oracle Support

My Oracle Support Passw...

Click on **Next**.

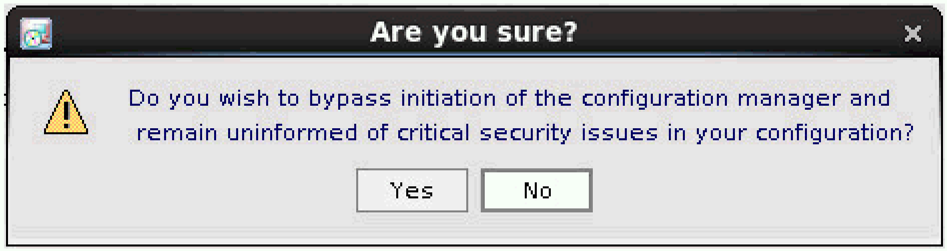


Email Address Not Specified

You have not provided an email address.

Do you wish to remain uninformed of critical security issues in your configuration?

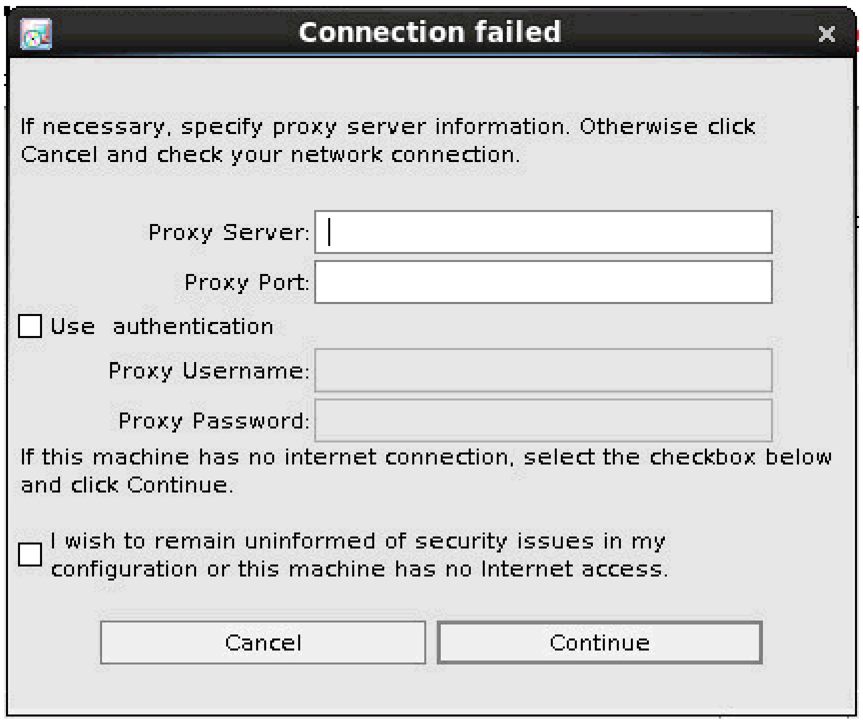
Click on **Yes**.



Are you sure?

Do you wish to bypass initiation of the configuration manager and remain uninformed of critical security issues in your configuration?

Click on **Yes**.



If necessary, specify proxy server information. Otherwise click Cancel and check your network connection.

Proxy Server:

Proxy Port:

☐ Use authentication

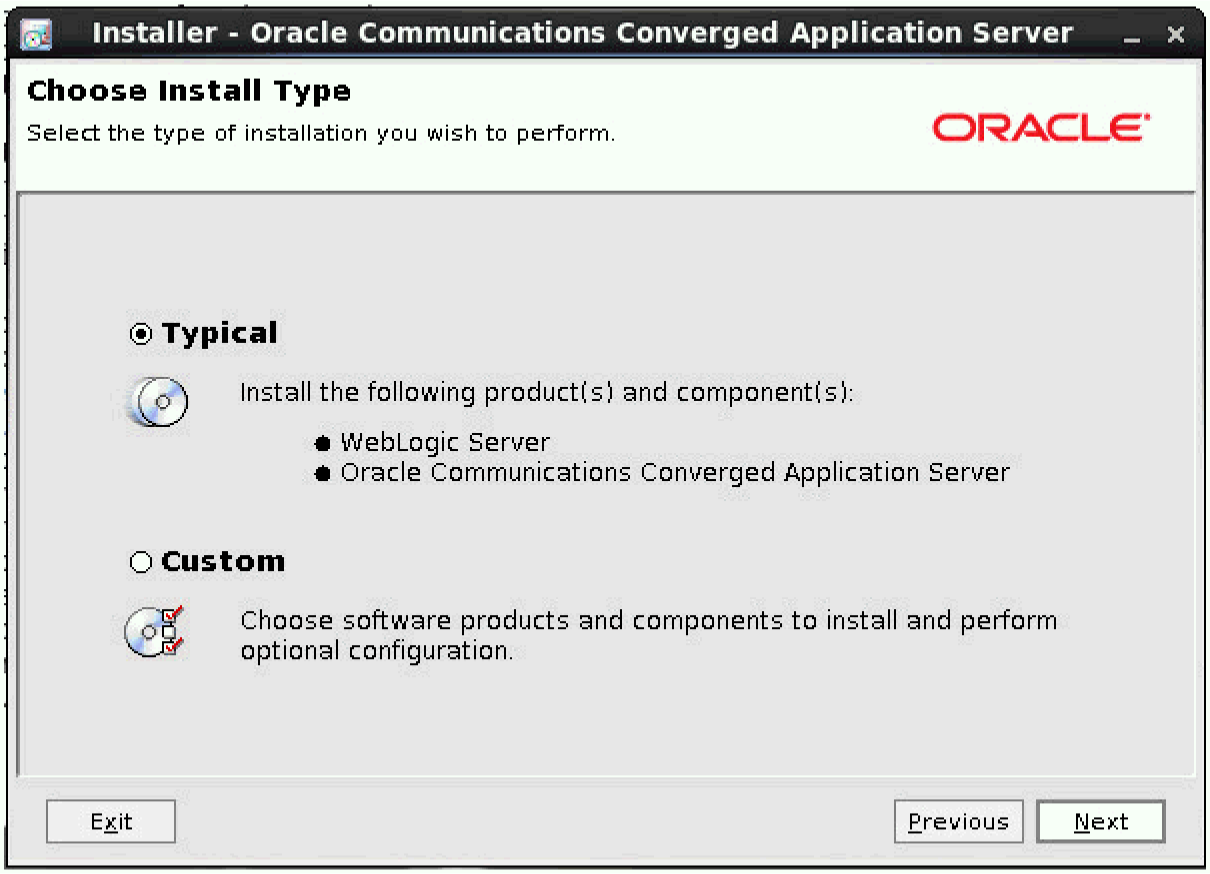
Proxy Username:

Proxy Password:

If this machine has no internet connection, select the checkbox below and click Continue.

☐ I wish to remain uninformed of security issues in my configuration or this machine has no Internet access.

Select **I wish to remain uninformed...** then click on **Continue.**


Installer - Oracle Communications Converged Application Server
ORACLE

Choose Install Type

Select the type of installation you wish to perform.

[✓ Typical]

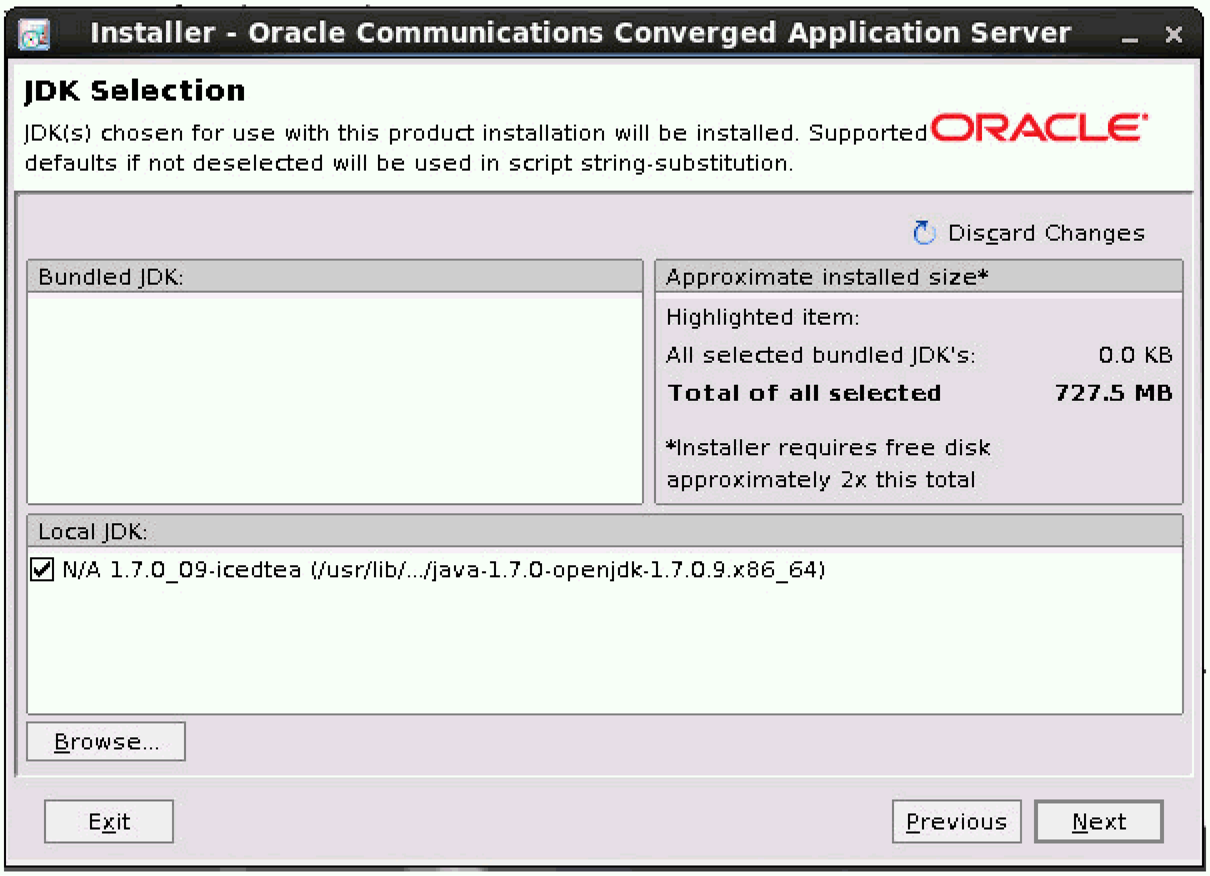
Install the following product(s) and component(s):

- WebLogic Server
- Oracle Communications Converged Application Server

Custom

Choose software products and components to install and perform optional configuration.

Select **Typical** then click on **Next.**


Installer - Oracle Communications Converged Application Server
ORACLE
Discard Changes
Browse...
Exit
Previous
Next

JDK Selection

JDK(s) chosen for use with this product installation will be installed. Supported defaults if not deselected will be used in script string-substitution.

Bundled JDK:

Approximate installed size*

Highlighted item:

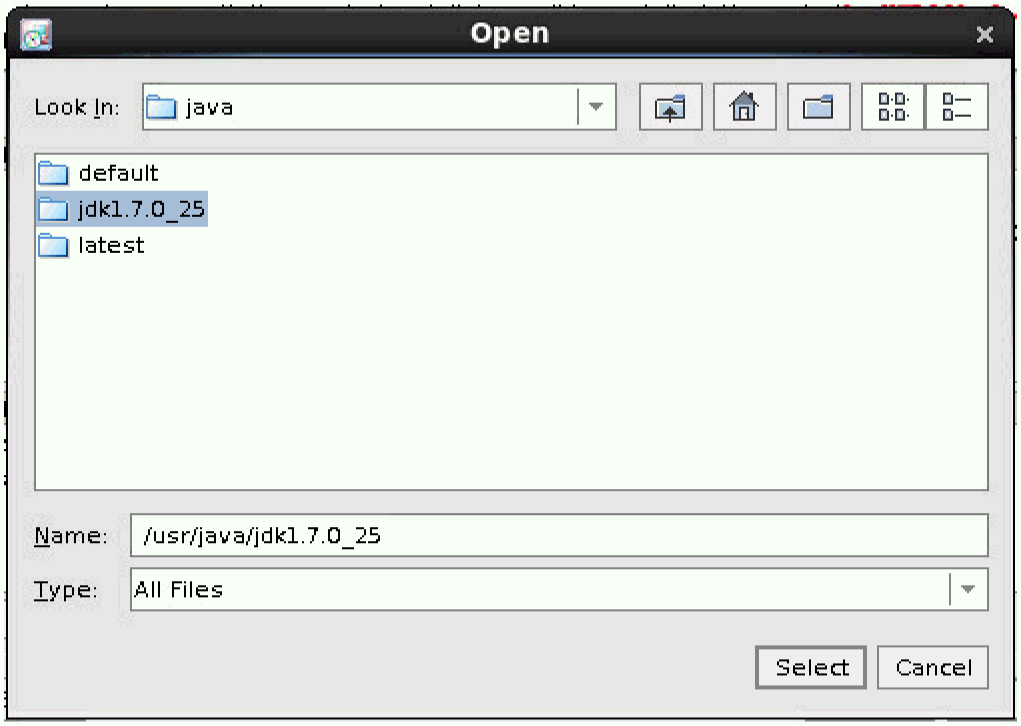
All selected bundled JDK's:	0.0 KB
Total of all selected	**727.5 MB**

*Installer requires free disk approximately 2x this total

Local JDK:

☑ N/A 1.7.0_09-icedtea (/usr/lib/.../java-1.7.0-openjdk-1.7.0.9.x86_64)

Browse to previously installed **jdk1.7.0_25** directory.



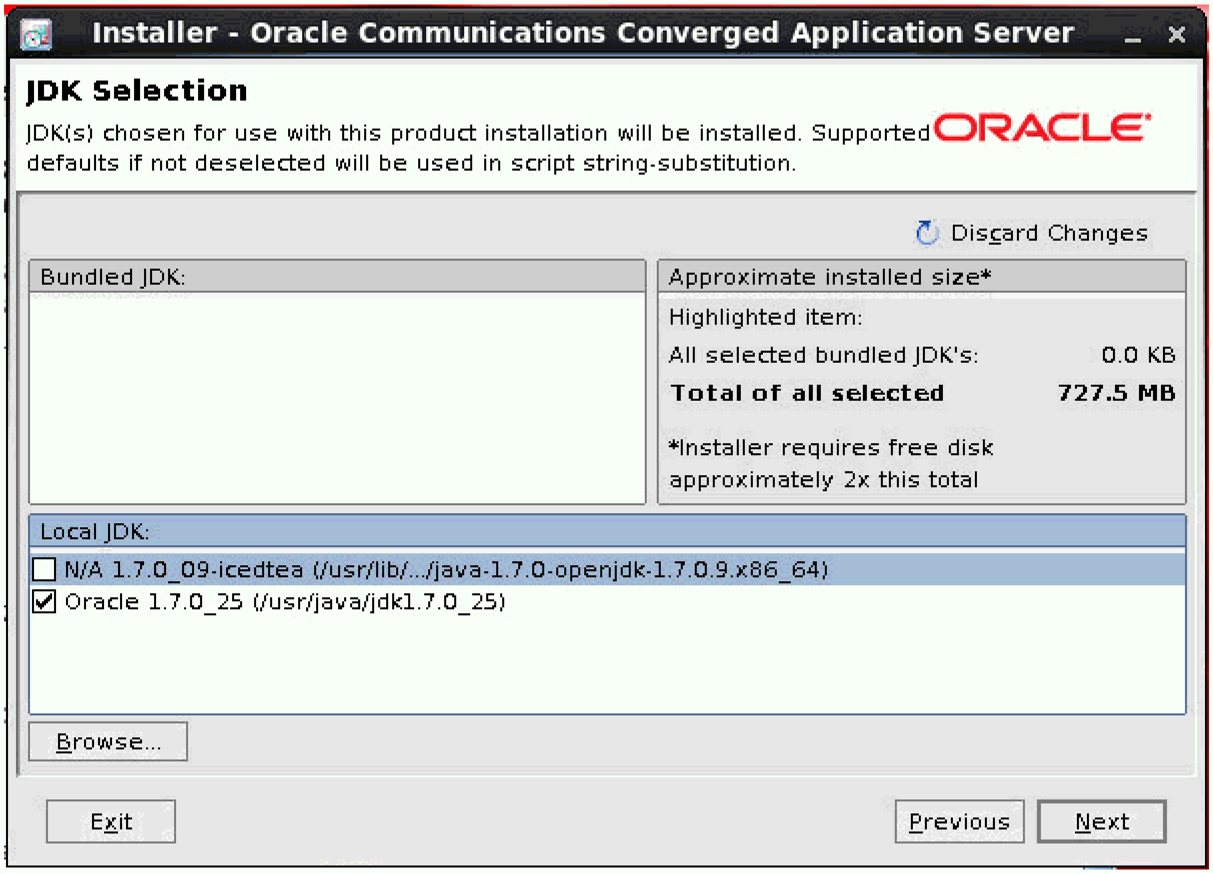
Look In: java

default
jdk1.7.0_25
latest

Name: /usr/java/jdk1.7.0_25

Type: All Files

Select **jdk1.7.0_25** directory then click on **Select**.


Installer - Oracle Communications Converged Application Server

JDK Selection

JDK(s) chosen for use with this product installation will be installed. Supported defaults if not deselected will be used in script string-substitution.

Approximate installed size*

Highlighted item:

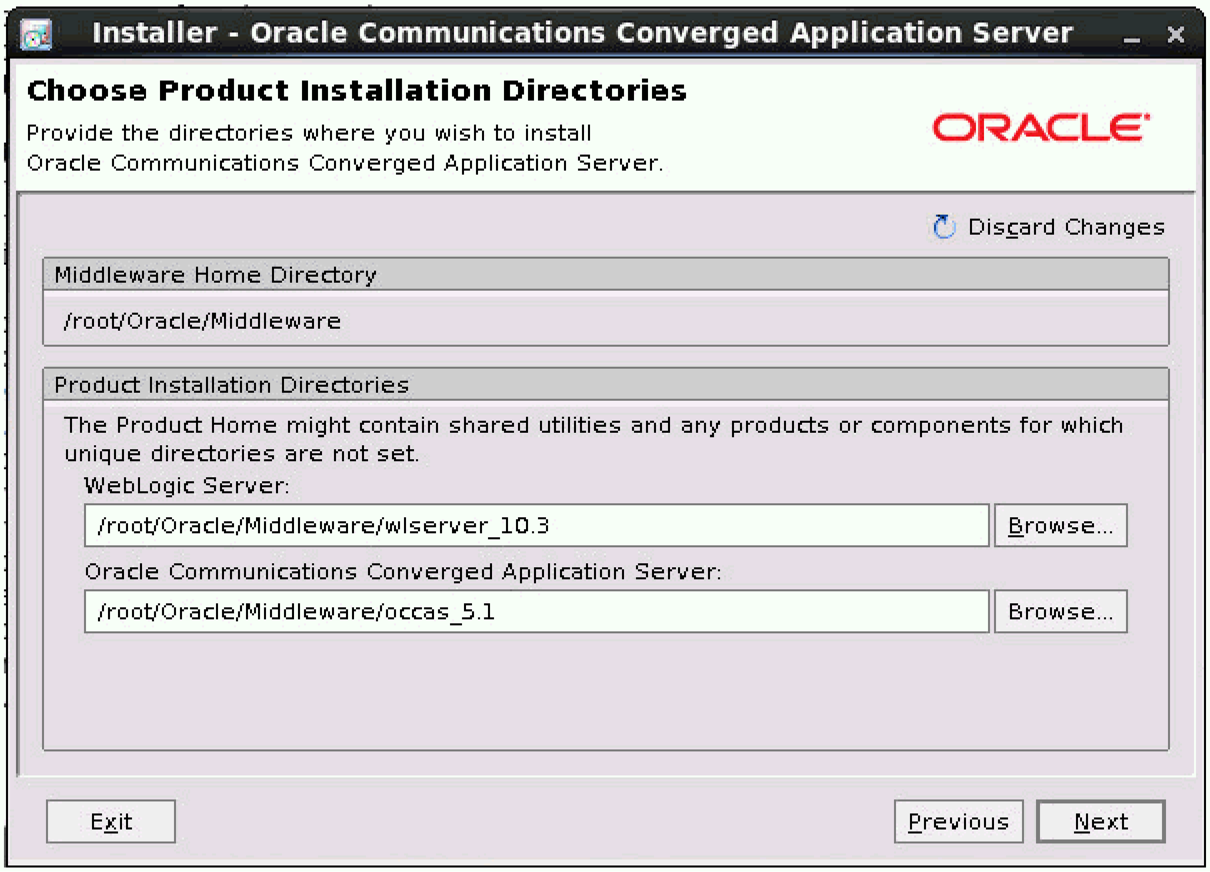
All selected bundled JDK's:	0.0 KB
Total of all selected	**727.5 MB**

*Installer requires free disk approximately 2x this total

Local JDK:

- [ ] N/A 1.7.0_09-icedtea (/usr/lib/.../java-1.7.0-openjdk-1.7.0.9.x86_64)
- [✓] Oracle 1.7.0_25 (/usr/java/jdk1.7.0_25)

Make sure the selected **jdk1.7.0_25** directory is the only JDK checked then click on **Next**.



Installer - Oracle Communications Converged Application Server

Choose Product Installation Directories

Provide the directories where you wish to install
Oracle Communications Converged Application Server.

Middleware Home Directory

/root/Oracle/Middleware

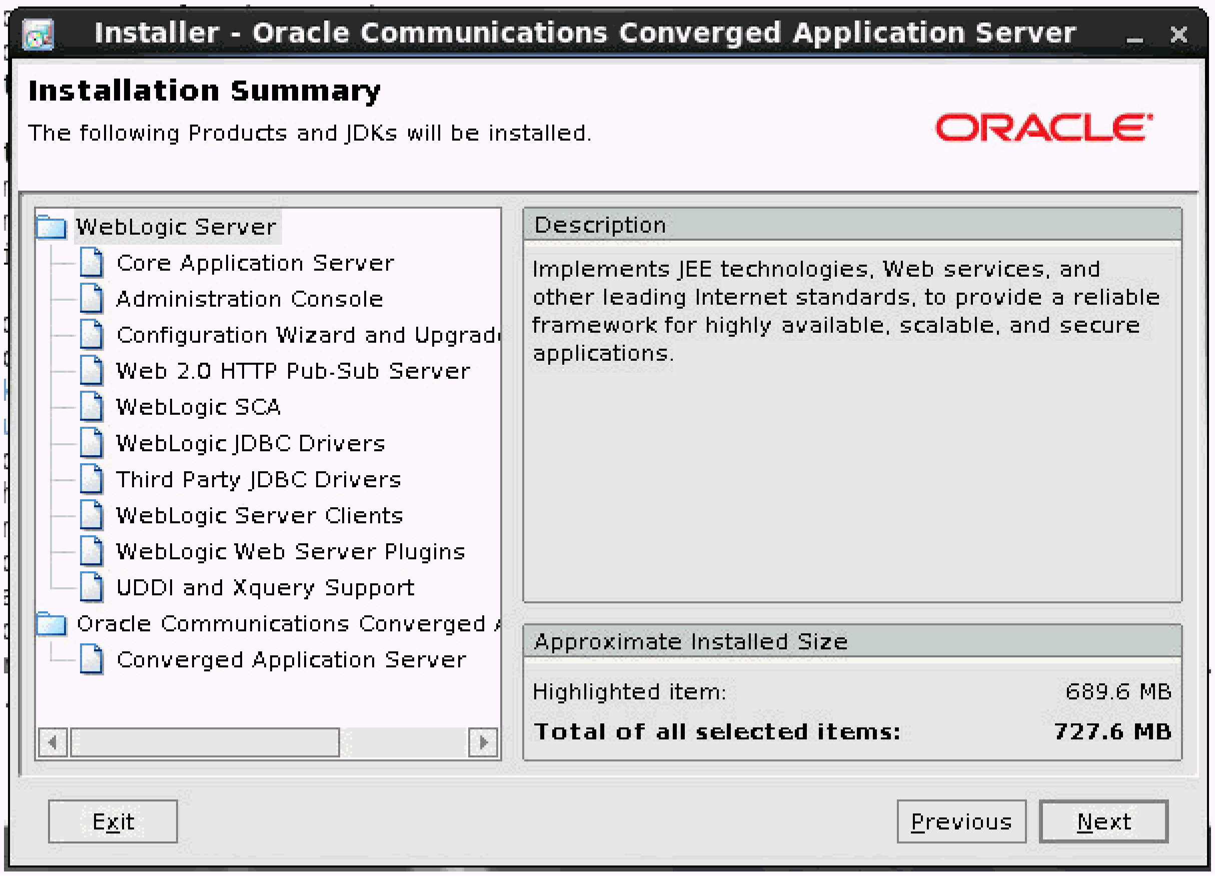
Product Installation Directories

The Product Home might contain shared utilities and any products or components for which unique directories are not set.

WebLogic Server: /root/Oracle/Middleware/wlserver_10.3

Oracle Communications Converged Application Server: /root/Oracle/Middleware/occas_5.1

Click on **Next**.



Installer - Oracle Communications Converged Application Server

Installation Summary

The following Products and JDKs will be installed.

- WebLogic Server
  - Core Application Server
  - Administration Console
  - Configuration Wizard and Upgrad
  - Web 2.0 HTTP Pub-Sub Server
  - WebLogic SCA
  - WebLogic JDBC Drivers
  - Third Party JDBC Drivers
  - WebLogic Server Clients
  - WebLogic Web Server Plugins
  - UDDI and Xquery Support
- Oracle Communications Converged
  - Converged Application Server

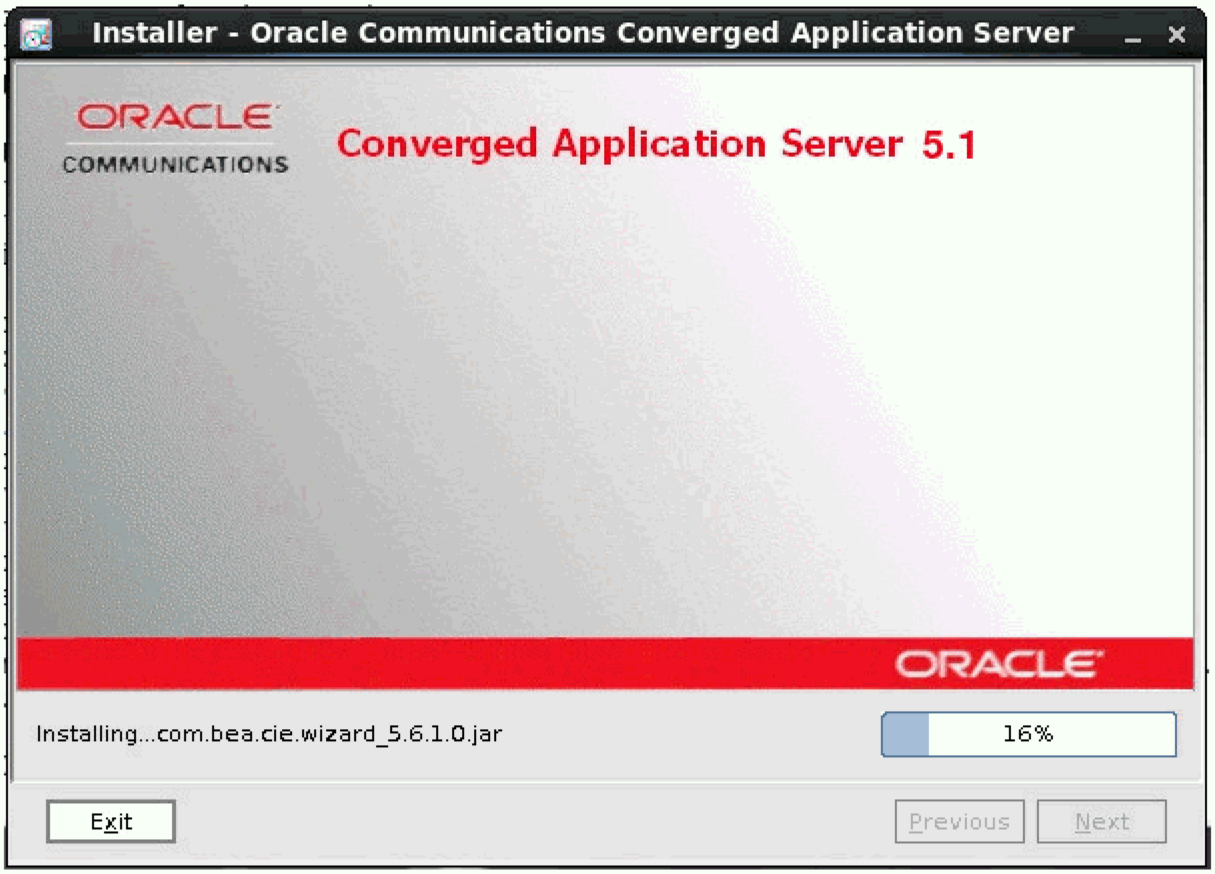
Description

Implements JEE technologies, Web services, and other leading Internet standards, to provide a reliable framework for highly available, scalable, and secure applications.

Approximate Installed Size

Highlighted item:	689.6 MB
Total of all selected items:	**727.6 MB**

Click on **Next**.


Installer - Oracle Communications Converged Application Server
ORACLE COMMUNICATIONS
Converged Application Server 5.1
ORACLE
Exit
Previous
Next

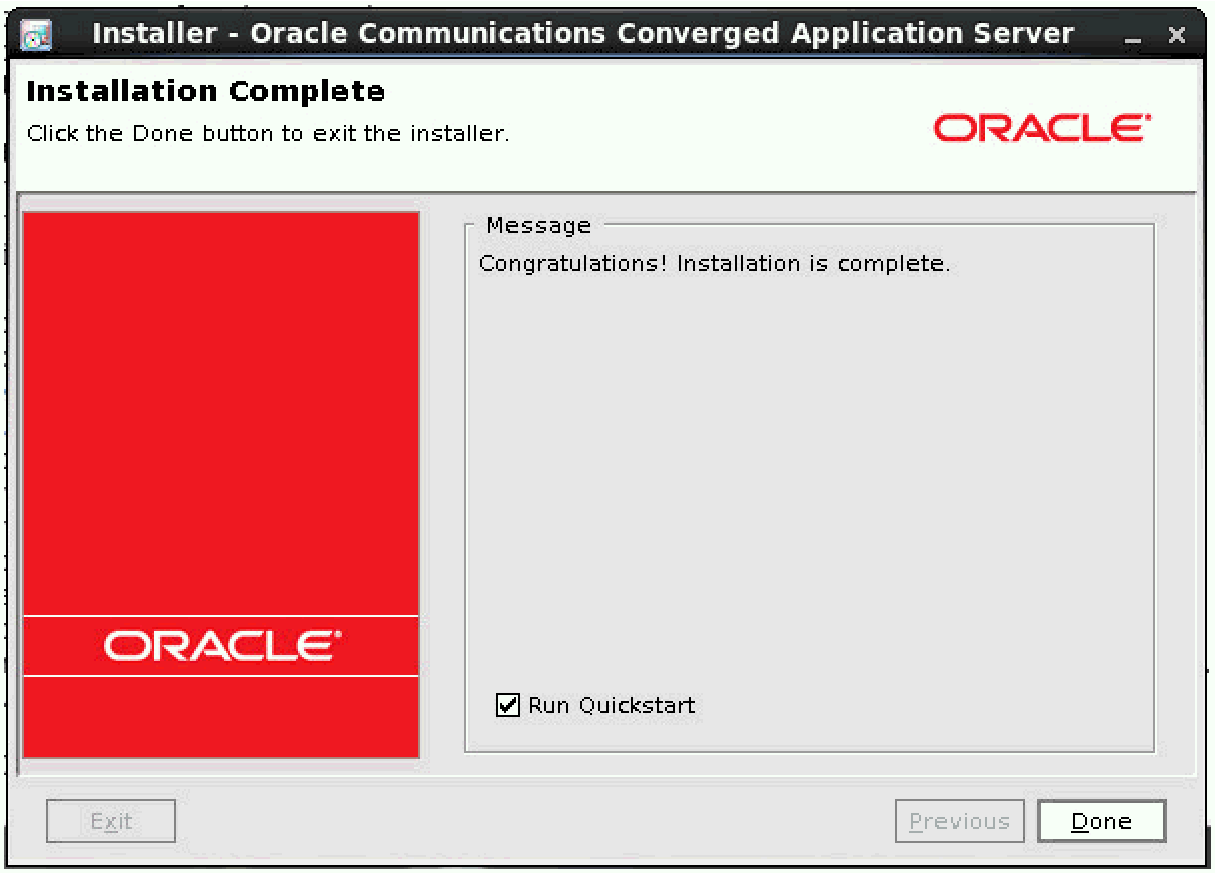
Installing...com.bea.cie.wizard_5.6.1.0.jar

16%

Click on **Next**.

At this point, OCCAS is installed. The steps in the next section will go over OCCAS Configuration.

OCCAS Configuration


Installer - Oracle Communications Converged Application Server

Installation Complete

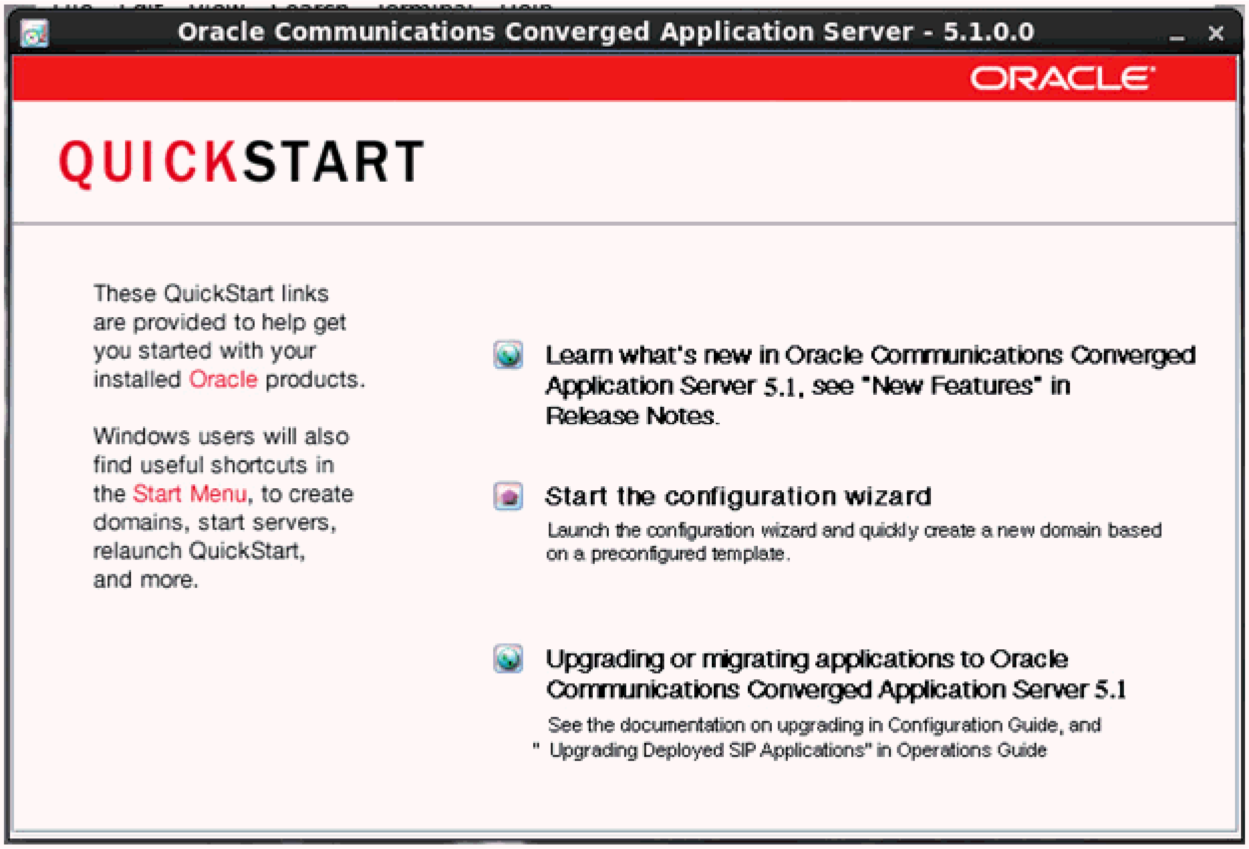
Click the Done button to exit the installer.

Message

Congratulations! Installation is complete.

☑ Run Quickstart

Select **Run Quickstart** then click on **Done**.


Oracle Communications Converged Application Server - 5.1.0.0
ORACLE
QUICKSTART

These QuickStart links are provided to help get you started with your installed Oracle products.

Windows users will also find useful shortcuts in the Start Menu, to create domains, start servers, relaunch QuickStart, and more.

Learn what's new in Oracle Communications Converged Application Server 5.1, see "New Features" in Release Notes.

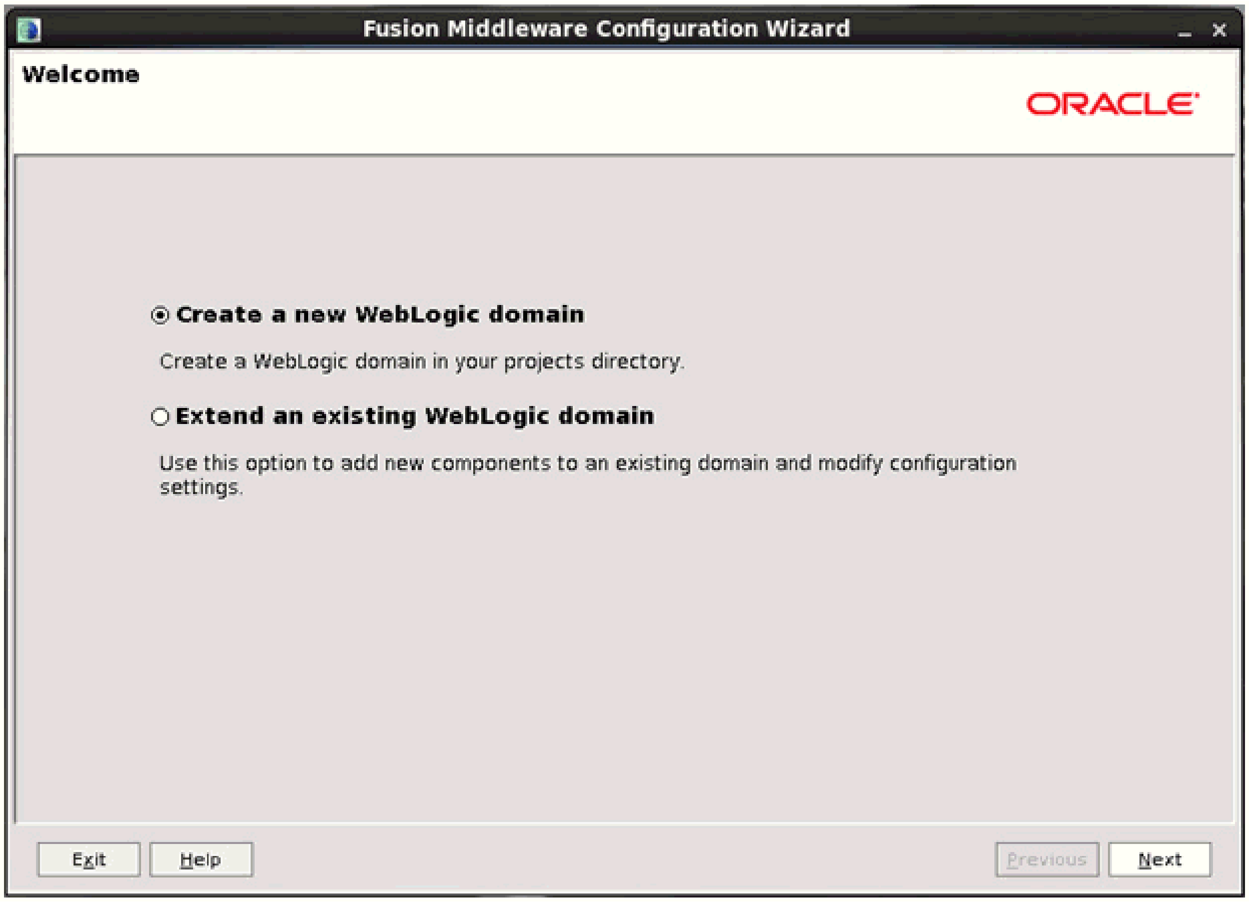
Start the configuration wizard

Launch the configuration wizard and quickly create a new domain based on a preconfigured template.

Upgrading or migrating applications to Oracle Communications Converged Application Server 5.1

See the documentation on upgrading in Configuration Guide, and " Upgrading Deployed SIP Applications" in Operations Guide

Click on **Start the configuration wizard**.


Fusion Middleware Configuration Wizard

Welcome

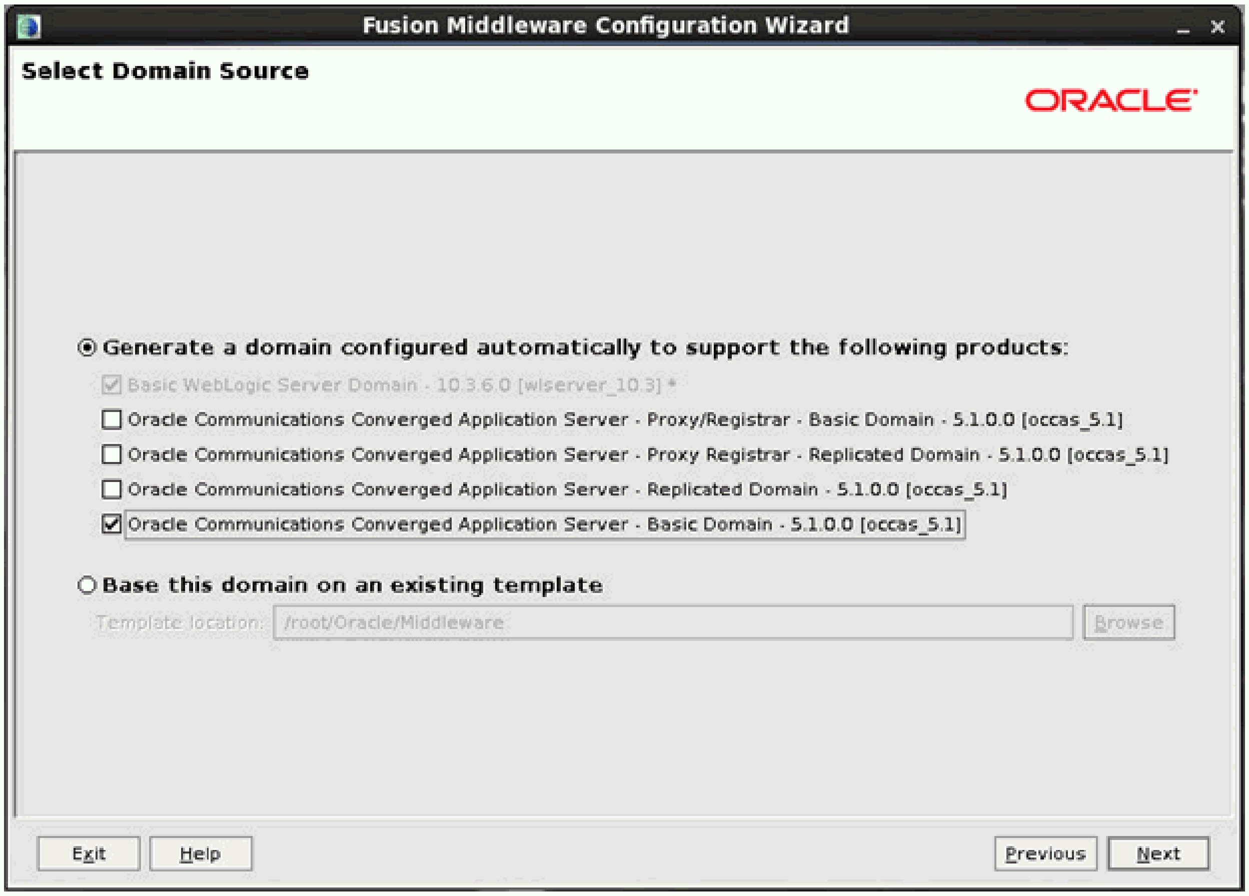
● Create a new WebLogic domain

Create a WebLogic domain in your projects directory.

○ Extend an existing WebLogic domain

Use this option to add new components to an existing domain and modify configuration settings.

Select **Create a new WebLogic domain** then click on **Next**.



Select Domain Source

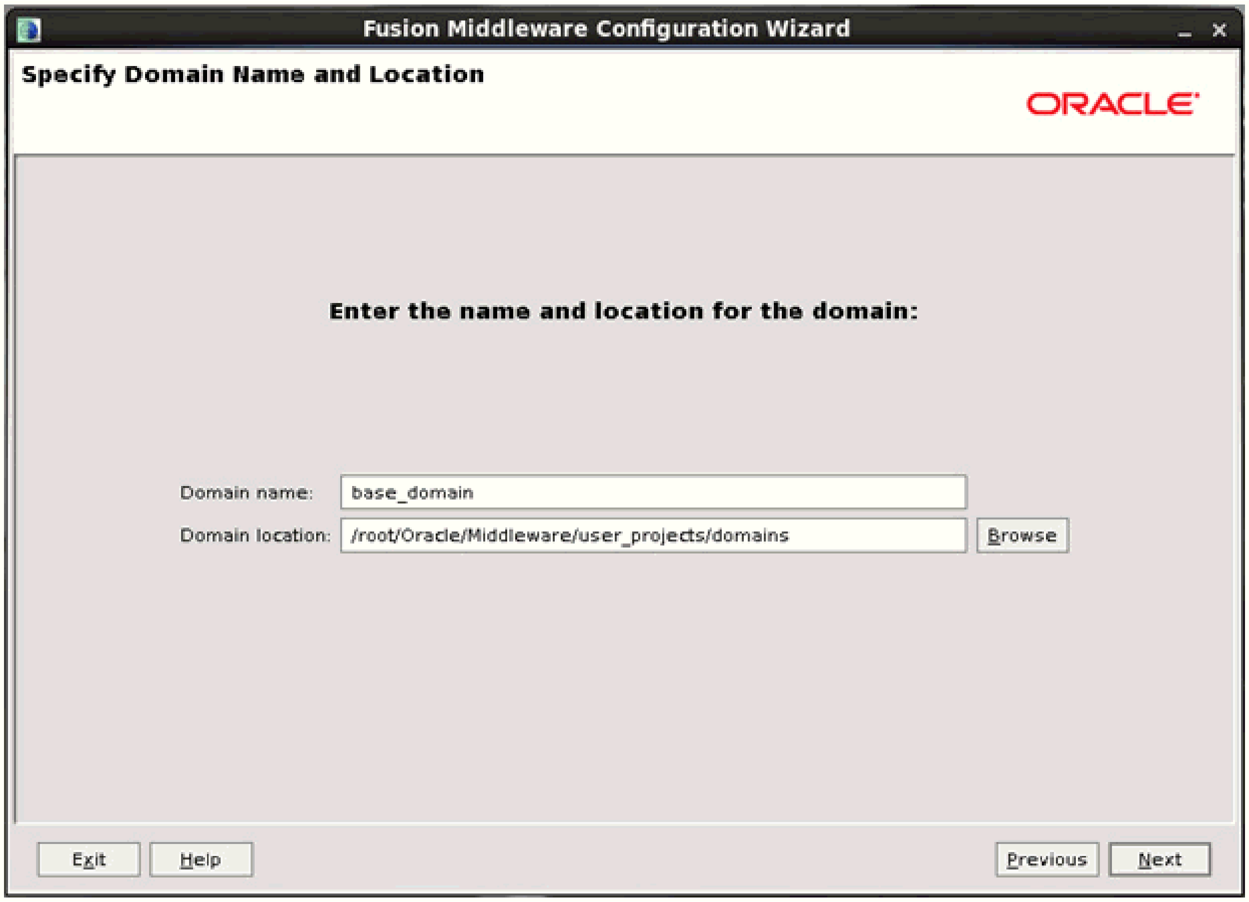
(●) Generate a domain configured automatically to support the following products:

- [✓] Basic WebLogic Server Domain - 10.3.6.0 [wlserver_10.3] *
- [ ] Oracle Communications Converged Application Server - Proxy/Registrar - Basic Domain - 5.1.0.0 [occas_5.1]
- [ ] Oracle Communications Converged Application Server - Proxy Registrar - Replicated Domain - 5.1.0.0 [occas_5.1]
- [ ] Oracle Communications Converged Application Server - Replicated Domain - 5.1.0.0 [occas_5.1]
- [✓] Oracle Communications Converged Application Server - Basic Domain - 5.1.0.0 [occas_5.1]

( ) Base this domain on an existing template

Template location: /root/Oracle/Middleware

Select **Generate a domain configured automatically to support the following products** and choose **Oracle Communications Converged Application Server - Basic Domain - 5.1.0.0 (occas_5.1)** then click on **Next**.

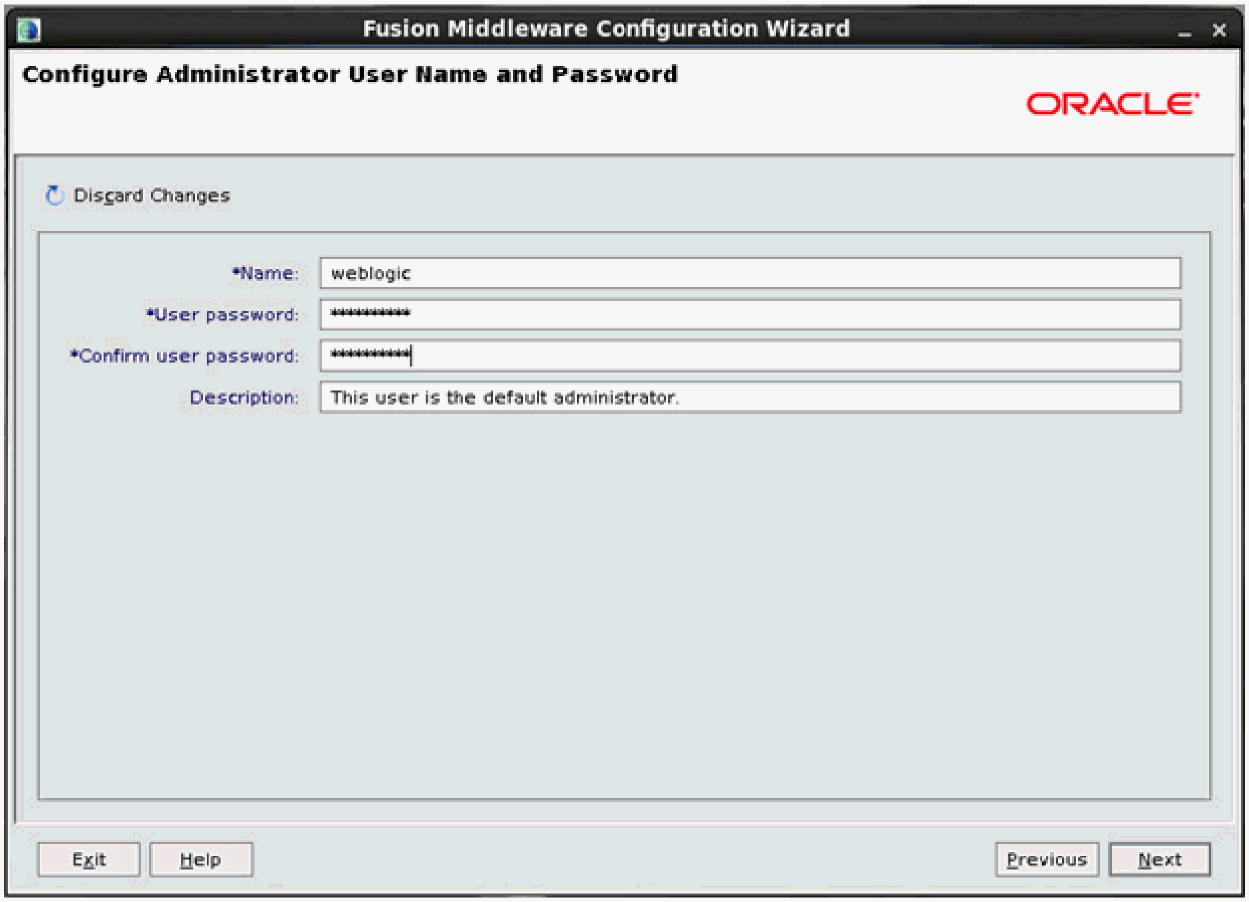

Fusion Middleware Configuration Wizard
Specify Domain Name and Location
ORACLE
Exit
Help
Previous
Next
Browse

Enter the name and location for the domain:

Domain name: base_domain

Domain location: /root/Oracle/Middleware/user_projects/domains

Click on **Next**.



Configure Administrator User Name and Password

*Name:	weblogic
*User password:	***********
*Confirm user password:	***********
Description:	This user is the default administrator.

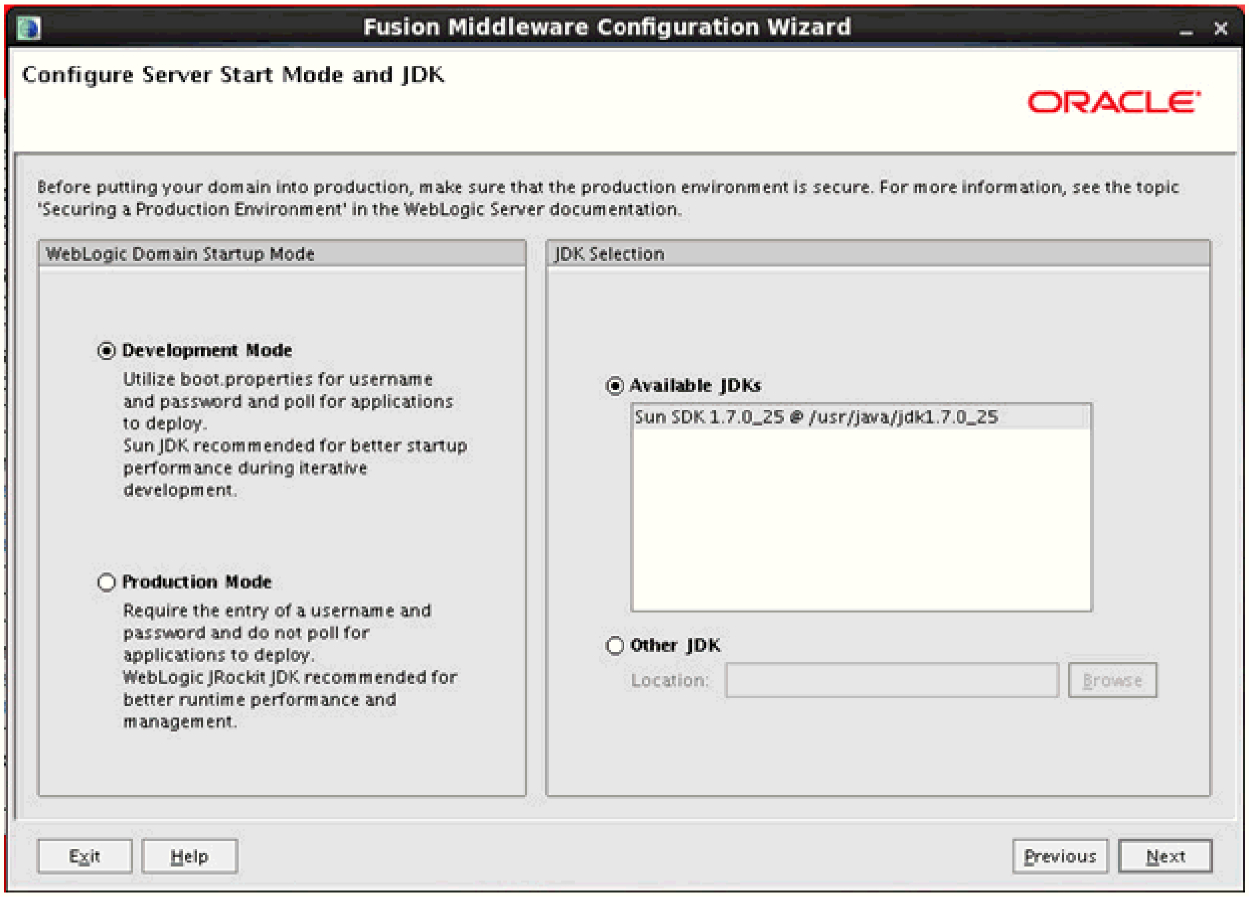
Specify **Name** and **User password** then click on Next.

The following is used as an example:

Name: weblogic

User password: Webl0gic!! ("0" is a zero)

Note: A strong password is required.



Configure Server Start Mode and JDK

Before putting your domain into production, make sure that the production environment is secure. For more information, see the topic 'Securing a Production Environment' in the WebLogic Server documentation.

WebLogic Domain Startup Mode

[✓ Development Mode]
Utilize boot.properties for username and password and poll for applications to deploy.
Sun JDK recommended for better startup performance during iterative development.

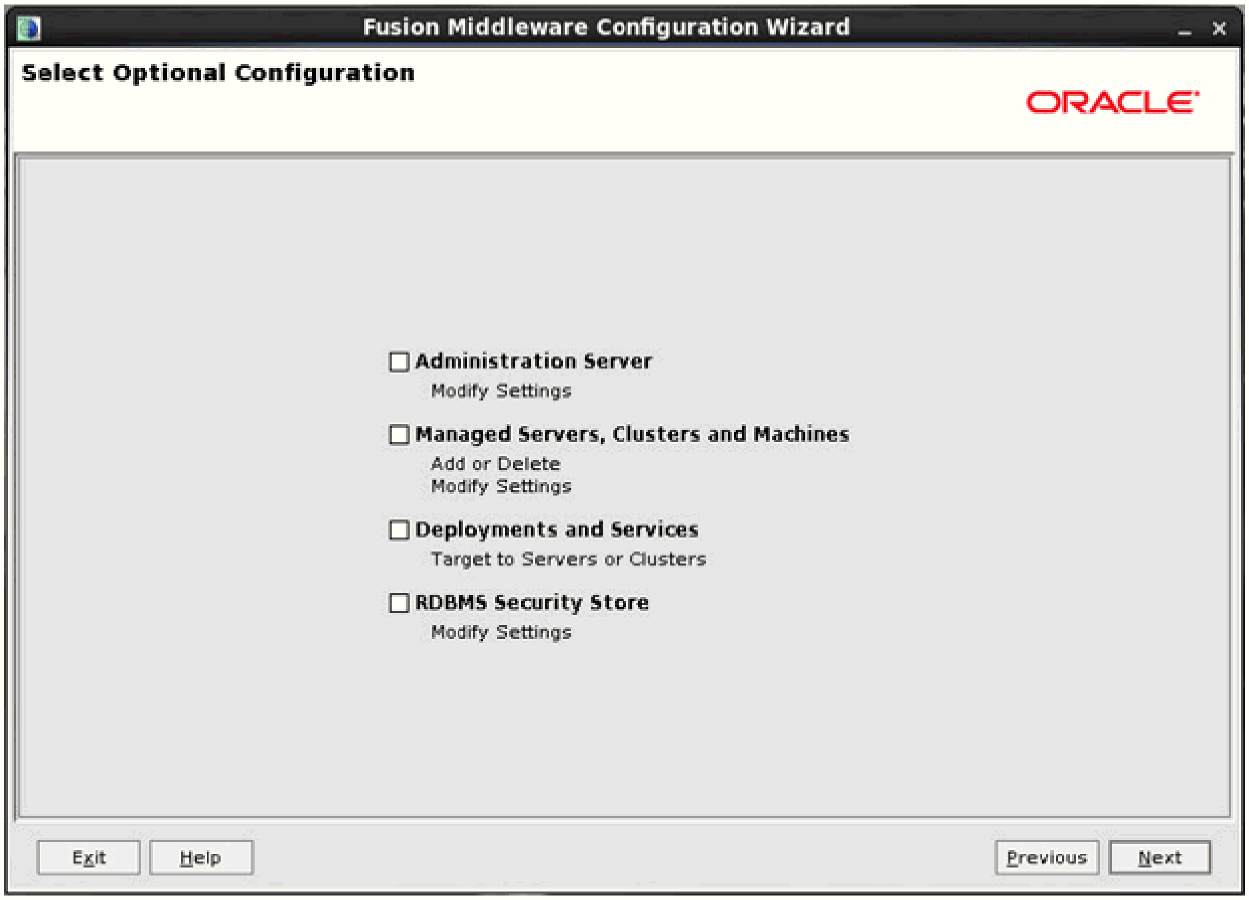
Production Mode
Require the entry of a username and password and do not poll for applications to deploy.
WebLogic JRockit JDK recommended for better runtime performance and management.

JDK Selection

[✓ Available JDKs]
Sun SDK 1.7.0_25 @ /usr/java/jdk1.7.0_25

Other JDK
Location:

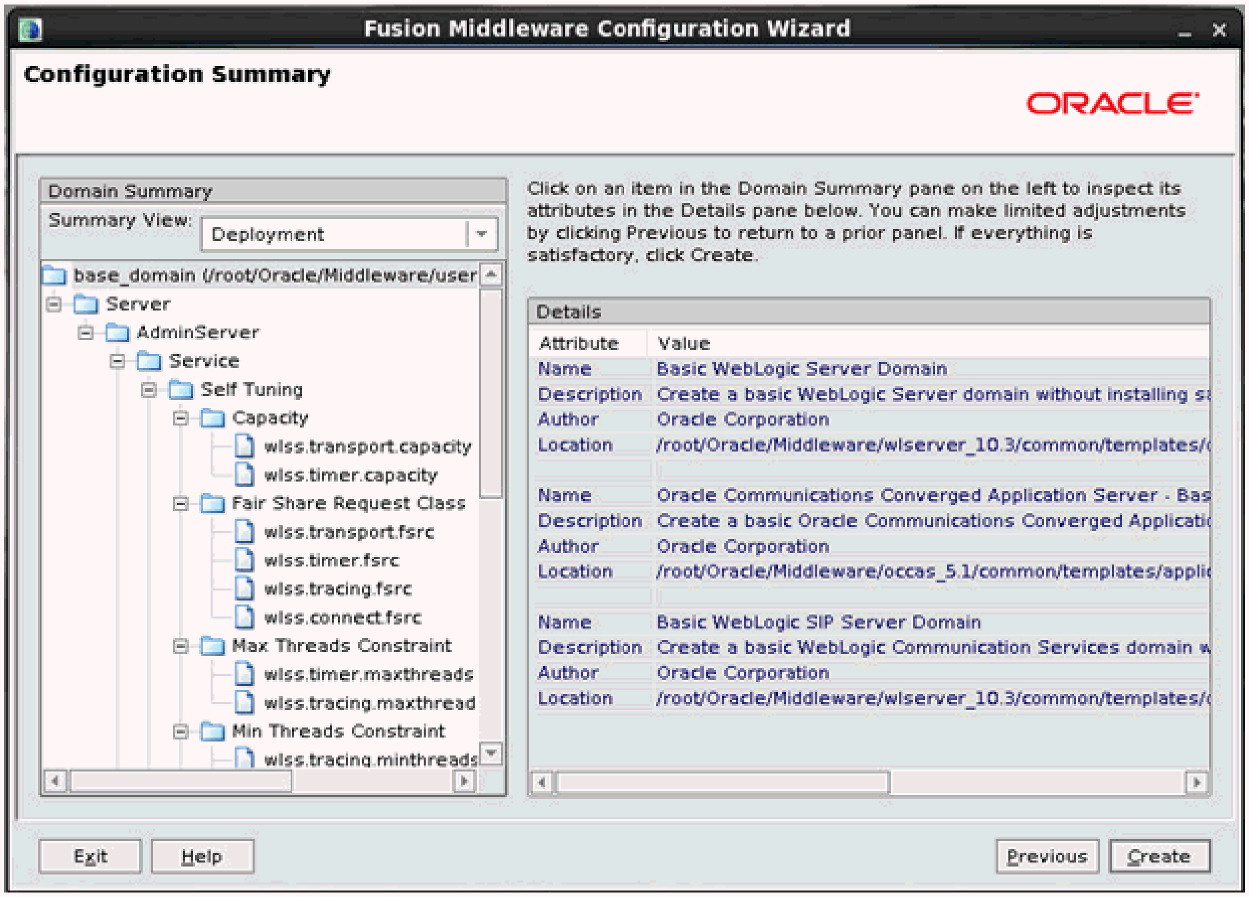
Select **Development Mode** then click on **Next**.



Select Optional Configuration

- [ ] Administration Server
  Modify Settings
- [ ] Managed Servers, Clusters and Machines
  Add or Delete
  Modify Settings
- [ ] Deployments and Services
  Target to Servers or Clusters
- [ ] RDBMS Security Store
  Modify Settings

Click on **Next**.



Configuration Summary

Domain Summary

Summary View: Deployment

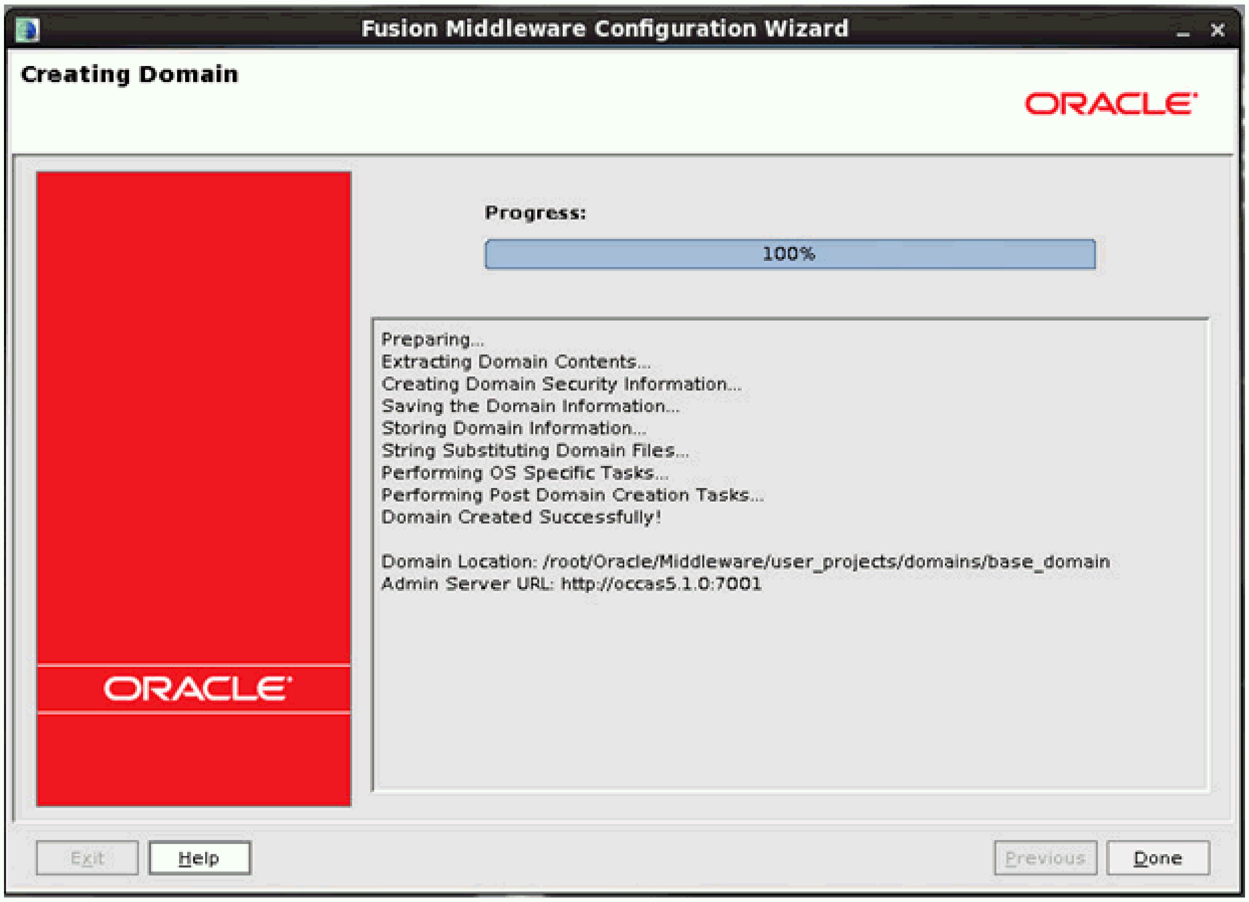
```
base_domain (/root/Oracle/Middleware/user
  Server
    AdminServer
      Service
        Self Tuning
          Capacity
            wlss.transport.capacity
            wlss.timer.capacity
          Fair Share Request Class
            wlss.transport.fsrc
            wlss.timer.fsrc
            wlss.tracing.fsrc
            wlss.connect.fsrc
          Max Threads Constraint
            wlss.timer.maxthreads
            wlss.tracing.maxthread
          Min Threads Constraint
            wlss.tracing.minthreads
```

Click on an item in the Domain Summary pane on the left to inspect its attributes in the Details pane below. You can make limited adjustments by clicking Previous to return to a prior panel. If everything is satisfactory, click Create.

Details

Attribute	Value
Name	Basic WebLogic Server Domain
Description	Create a basic WebLogic Server domain without installing s
Author	Oracle Corporation
Location	/root/Oracle/Middleware/wlserver_10.3/common/templates/
Name	Oracle Communications Converged Application Server - Bas
Description	Create a basic Oracle Communications Converged Applicati
Author	Oracle Corporation
Location	/root/Oracle/Middleware/occas_5.1/common/templates/appli
Name	Basic WebLogic SIP Server Domain
Description	Create a basic WebLogic Communication Services domain w
Author	Oracle Corporation
Location	/root/Oracle/Middleware/wlserver_10.3/common/templates/

Click on **Create.**



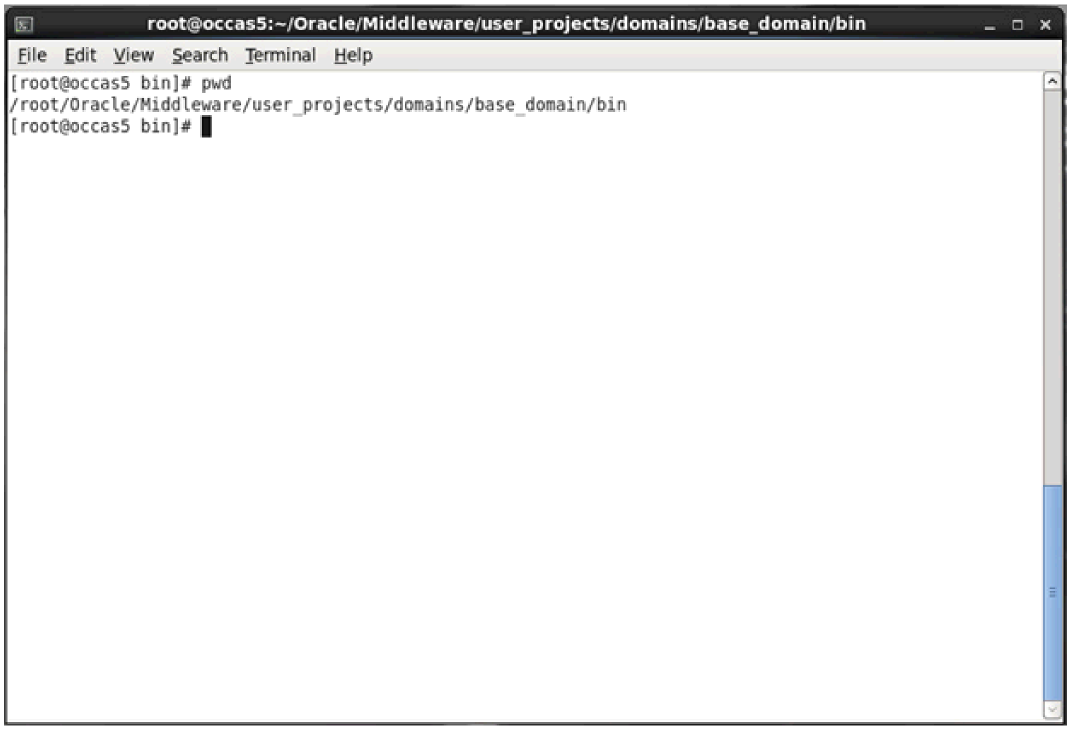
```
Preparing...
Extracting Domain Contents...
Creating Domain Security Information...
Saving the Domain Information...
Storing Domain Information...
String Substituting Domain Files...
Performing OS Specific Tasks...
Performing Post Domain Creation Tasks...
Domain Created Successfully!

Domain Location: /root/Oracle/Middleware/user_projects/domains/base_domain
Admin Server URL: http://occas5.1.0:7001
```

The OCCAS installation and configuration is complete. Click on **Done**.

OCCAS Startup

To start OCCAS, go to the `<Domain Location>/bin` directory:

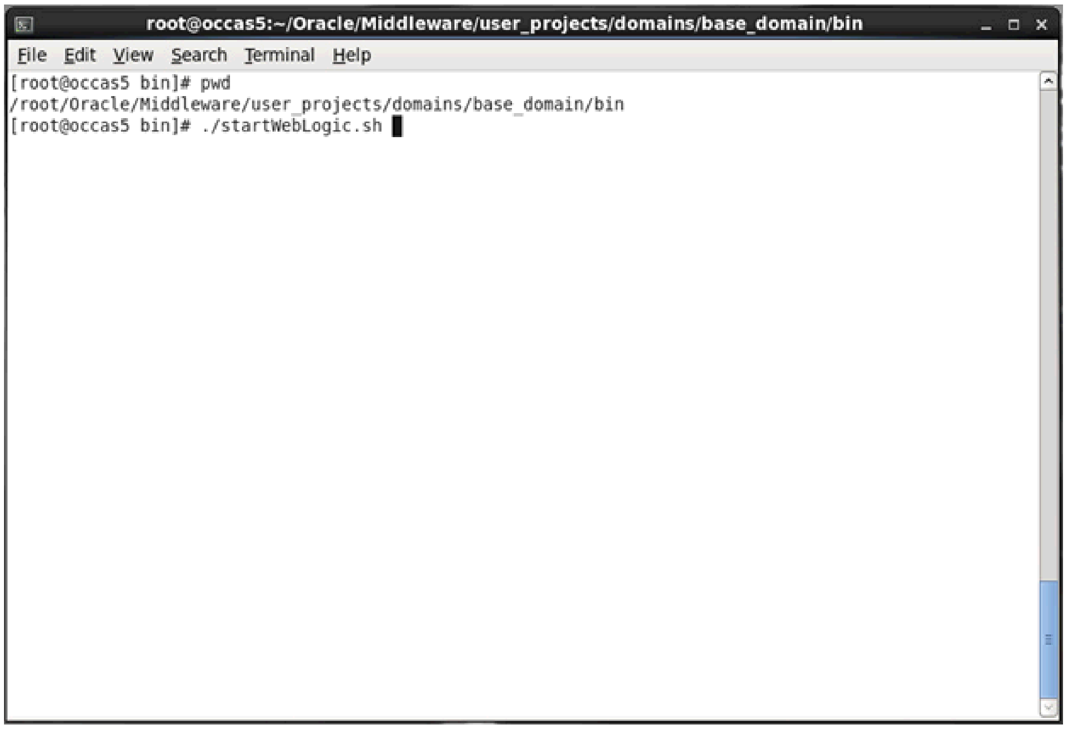
```
/root/Oracle/Middleware/user_projects/domains/base_domain/bin
```


root@occas5:~/Oracle/Middleware/user_projects/domains/base_domain/bin
File Edit View Search Terminal Help

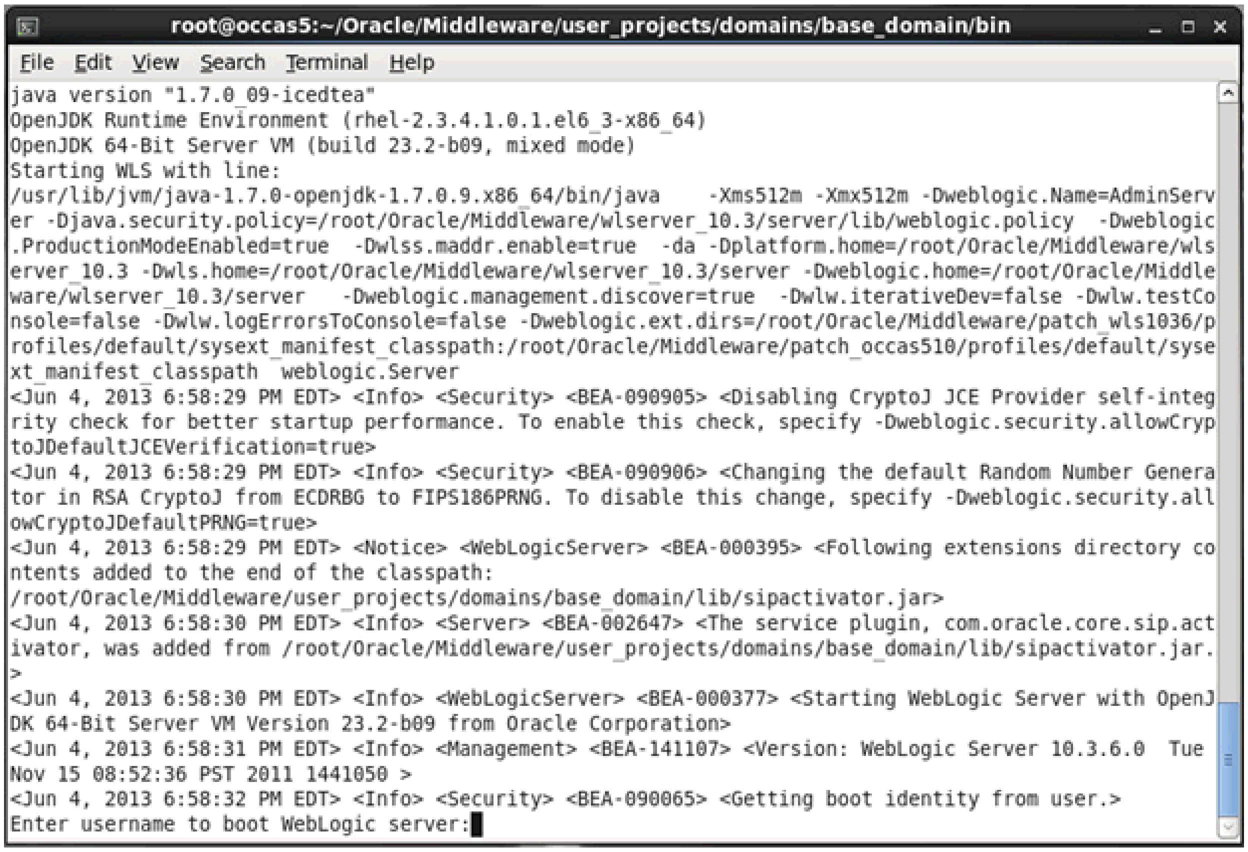
```
[root@occas5 bin]# pwd
/root/Oracle/Middleware/user_projects/domains/base_domain/bin
[root@occas5 bin]#
```

Run the following command:

```
./startWebLogic.sh
```


root@occas5:~/Oracle/Middleware/user_projects/domains/base_domain/bin
File Edit View Search Terminal Help

```
[root@occas5 bin]# pwd
/root/Oracle/Middleware/user_projects/domains/base_domain/bin
[root@occas5 bin]# ./startWebLogic.sh
```


root@occas5:~/Oracle/Middleware/user_projects/domains/base_domain/bin

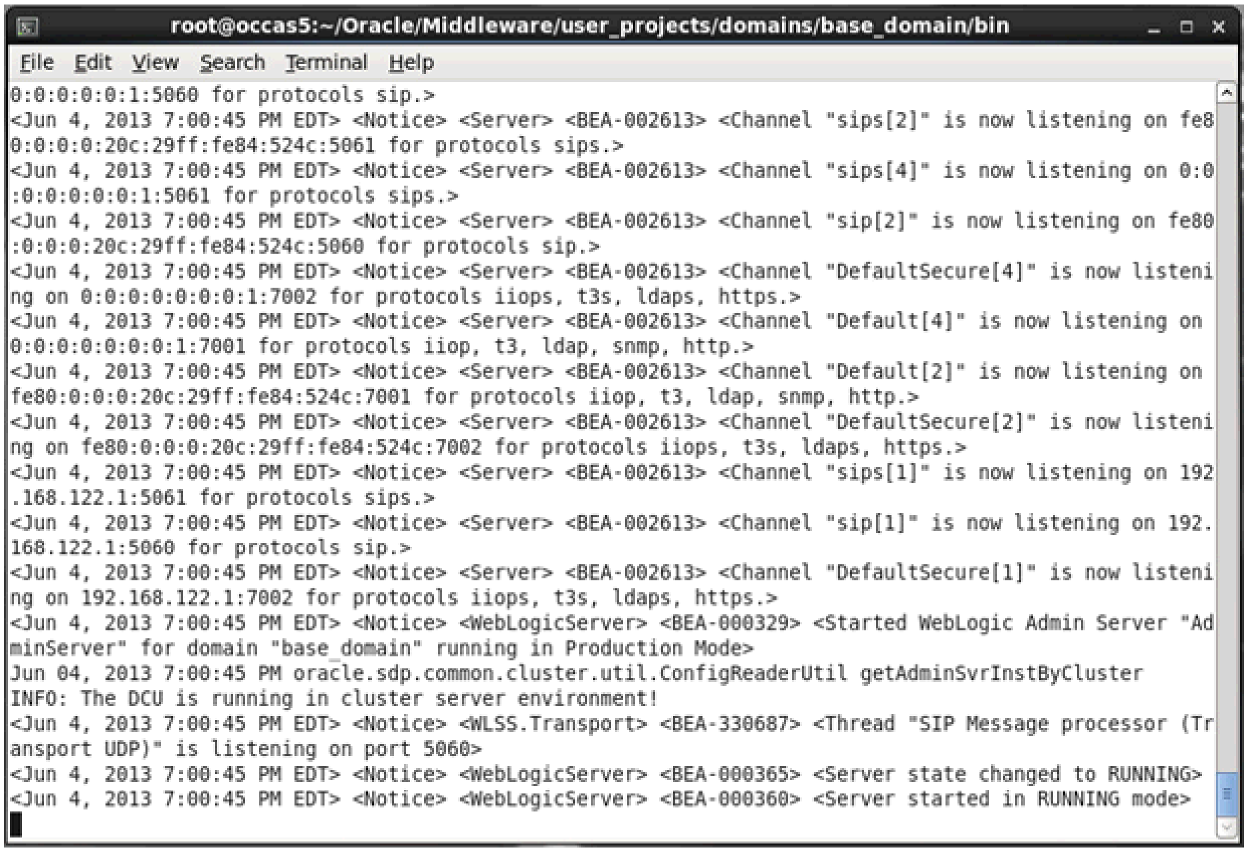
```
java version "1.7.0_09-icedtea"
OpenJDK Runtime Environment (rhel-2.3.4.1.0.1.el6_3-x86_64)
OpenJDK 64-Bit Server VM (build 23.2-b09, mixed mode)
Starting WLS with line:
/usr/lib/jvm/java-1.7.0-openjdk-1.7.0.9.x86_64/bin/java    -Xms512m -Xmx512m -Dweblogic.Name=AdminServ
er -Djava.security.policy=/root/Oracle/Middleware/wlserver_10.3/server/lib/weblogic.policy  -Dweblogic
.ProductionModeEnabled=true  -Dwlss.maddr.enable=true  -da -Dplatform.home=/root/Oracle/Middleware/wls
erver_10.3 -Dwls.home=/root/Oracle/Middleware/wlserver_10.3/server -Dweblogic.home=/root/Oracle/Middle
ware/wlserver_10.3/server   -Dweblogic.management.discover=true  -Dwlw.iterativeDev=false -Dwlw.testCo
nsole=false -Dwlw.logErrorsToConsole=false -Dweblogic.ext.dirs=/root/Oracle/Middleware/patch_wls1036/p
rofiles/default/sysext_manifest_classpath:/root/Oracle/Middleware/patch_occas510/profiles/default/syse
xt_manifest_classpath  weblogic.Server
<Jun 4, 2013 6:58:29 PM EDT> <Info> <Security> <BEA-090905> <Disabling CryptoJ JCE Provider self-integ
rity check for better startup performance. To enable this check, specify -Dweblogic.security.allowCryp
toJDefaultJCEVerification=true>
<Jun 4, 2013 6:58:29 PM EDT> <Info> <Security> <BEA-090906> <Changing the default Random Number Genera
tor in RSA CryptoJ from ECDRBG to FIPS186PRNG. To disable this change, specify -Dweblogic.security.all
owCryptoJDefaultPRNG=true>
<Jun 4, 2013 6:58:29 PM EDT> <Notice> <WebLogicServer> <BEA-000395> <Following extensions directory co
ntents added to the end of the classpath:
/root/Oracle/Middleware/user_projects/domains/base_domain/lib/sipactivator.jar>
<Jun 4, 2013 6:58:30 PM EDT> <Info> <Server> <BEA-002647> <The service plugin, com.oracle.core.sip.act
ivator, was added from /root/Oracle/Middleware/user_projects/domains/base_domain/lib/sipactivator.jar.
>
<Jun 4, 2013 6:58:30 PM EDT> <Info> <WebLogicServer> <BEA-000377> <Starting WebLogic Server with OpenJ
DK 64-Bit Server VM Version 23.2-b09 from Oracle Corporation>
<Jun 4, 2013 6:58:31 PM EDT> <Info> <Management> <BEA-141107> <Version: WebLogic Server 10.3.6.0  Tue
Nov 15 08:52:36 PST 2011 1441050 >
<Jun 4, 2013 6:58:32 PM EDT> <Info> <Security> <BEA-090065> <Getting boot identity from user.>
Enter username to boot WebLogic server:
```

Since the Development Mode installation was chosen, it is not necessary to enter username/password during script startup. If the Production Mode installation was chosen, you will have to specify username/password.

The following is used as an example:

Name: weblogic

User password: Webl0gic!! ("0" is a zero)

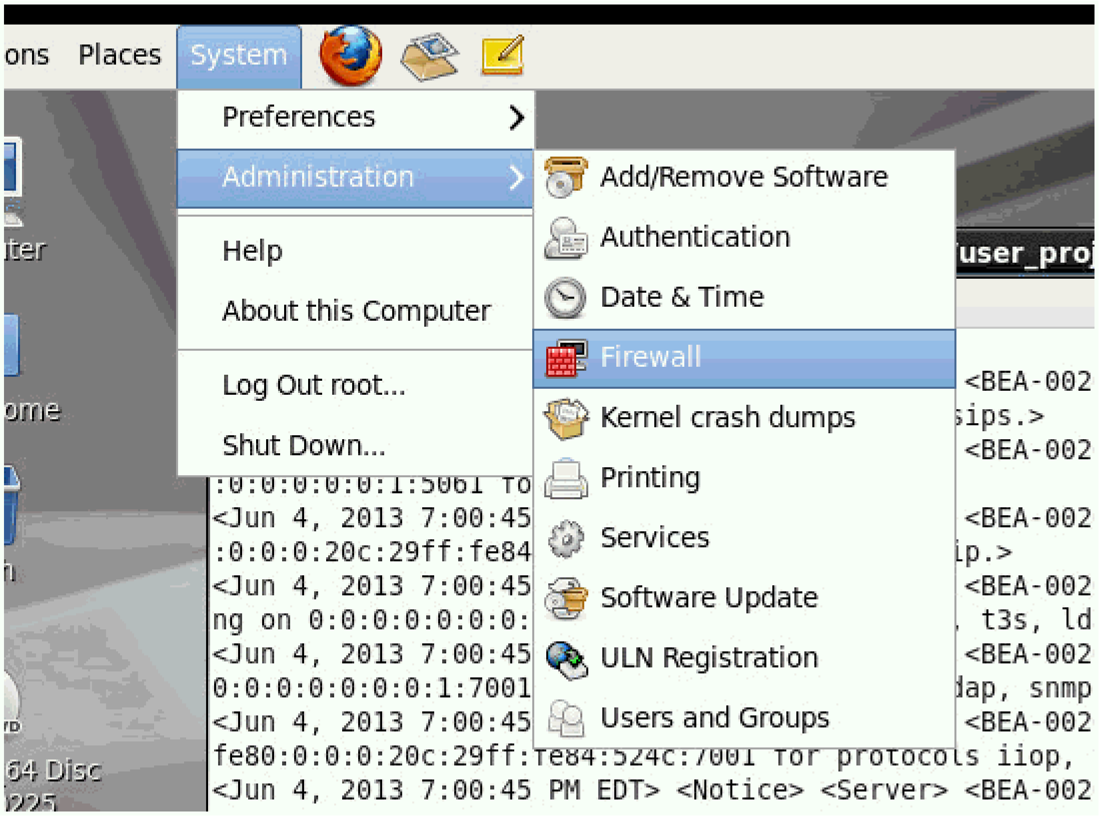

root@occas5:~/Oracle/Middleware/user_projects/domains/base_domain/bin

```
0:0:0:0:0:1:5060 for protocols sip.>
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <Server> <BEA-002613> <Channel "sips[2]" is now listening on fe8
0:0:0:0:20c:29ff:fe84:524c:5061 for protocols sips.>
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <Server> <BEA-002613> <Channel "sips[4]" is now listening on 0:0
:0:0:0:0:0:1:5061 for protocols sips.>
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <Server> <BEA-002613> <Channel "sip[2]" is now listening on fe80
:0:0:0:20c:29ff:fe84:524c:5060 for protocols sip.>
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <Server> <BEA-002613> <Channel "DefaultSecure[4]" is now listeni
ng on 0:0:0:0:0:0:0:1:7002 for protocols iiops, t3s, ldaps, https.>
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <Server> <BEA-002613> <Channel "Default[4]" is now listening on
0:0:0:0:0:0:0:1:7001 for protocols iiop, t3, ldap, snmp, http.>
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <Server> <BEA-002613> <Channel "Default[2]" is now listening on
fe80:0:0:0:20c:29ff:fe84:524c:7001 for protocols iiop, t3, ldap, snmp, http.>
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <Server> <BEA-002613> <Channel "DefaultSecure[2]" is now listeni
ng on fe80:0:0:0:20c:29ff:fe84:524c:7002 for protocols iiops, t3s, ldaps, https.>
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <Server> <BEA-002613> <Channel "sips[1]" is now listening on 192
.168.122.1:5061 for protocols sips.>
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <Server> <BEA-002613> <Channel "sip[1]" is now listening on 192.
168.122.1:5060 for protocols sip.>
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <Server> <BEA-002613> <Channel "DefaultSecure[1]" is now listeni
ng on 192.168.122.1:7002 for protocols iiops, t3s, ldaps, https.>
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <WebLogicServer> <BEA-000329> <Started WebLogic Admin Server "Ad
minServer" for domain "base_domain" running in Production Mode>
Jun 04, 2013 7:00:45 PM oracle.sdp.common.cluster.util.ConfigReaderUtil getAdminSvrInstByCluster
INFO: The DCU is running in cluster server environment!
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <WLSS.Transport> <BEA-330687> <Thread "SIP Message processor (Tr
ansport UDP)" is listening on port 5060>
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <WebLogicServer> <BEA-000365> <Server state changed to RUNNING>
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <WebLogicServer> <BEA-000360> <Server started in RUNNING mode>
```

To verify that OCCAS is started, check if **<Server started in RUNNING mode>** is displayed.

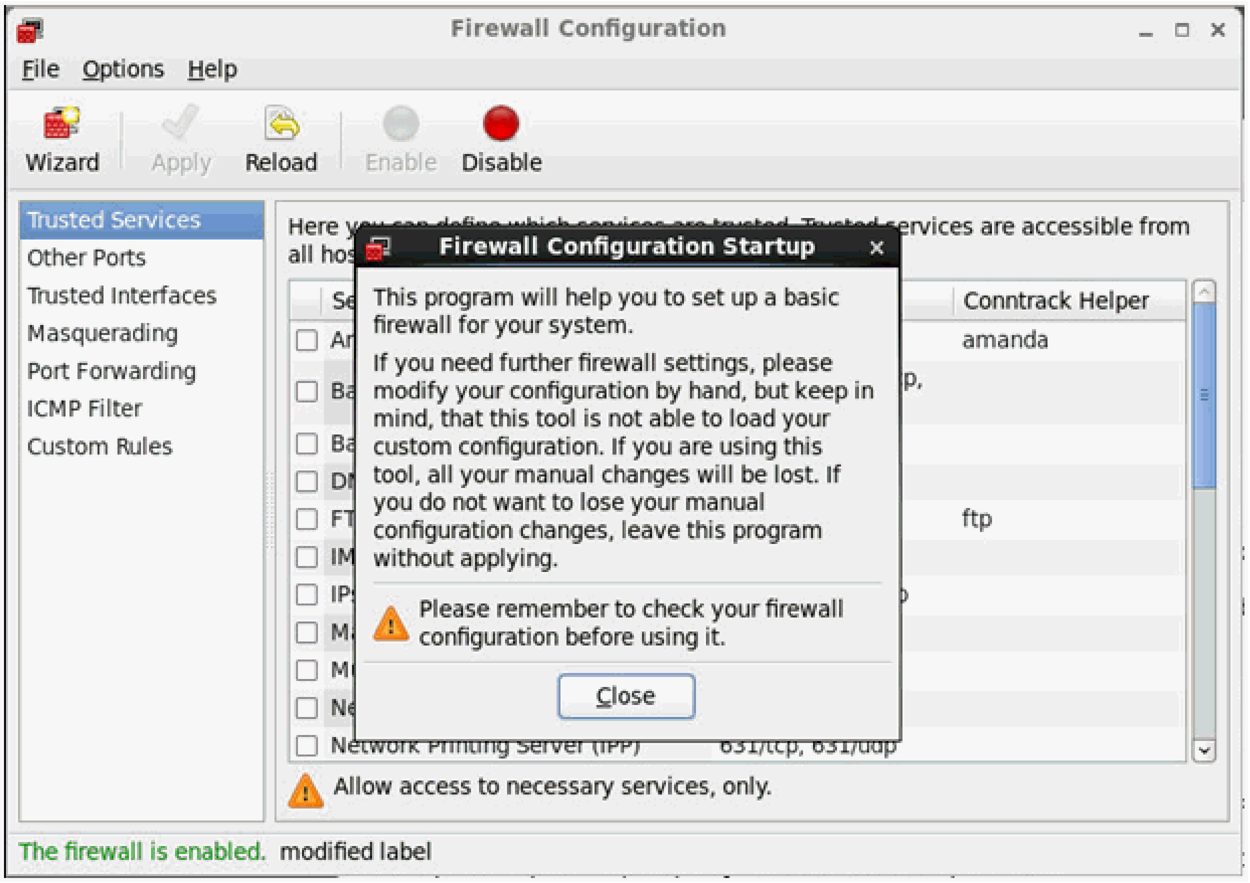
Firewall Configuration

Enable port 7001/tcp, 5060/udp, and 5061/upd in the Linux firewall.



```
:0:0:0:0:0:1:5061 fo
<Jun 4, 2013 7:00:45                                    <BEA-002
:0:0:0:20c:29ff:fe84                                    ip.>
<Jun 4, 2013 7:00:45                                    <BEA-002
ng on 0:0:0:0:0:0:0:                                    , t3s, ld
<Jun 4, 2013 7:00:45                                    <BEA-002
0:0:0:0:0:0:0:1:7001                                    dap, snmp
<Jun 4, 2013 7:00:45                                    <BEA-002
fe80:0:0:0:20c:29ff:fe84:524c:7001 for protocols iiop,
<Jun 4, 2013 7:00:45 PM EDT> <Notice> <Server> <BEA-002
```

Go to **System** > **Administration** and select **Firewall**.



Firewall Configuration Startup

This program will help you to set up a basic firewall for your system.

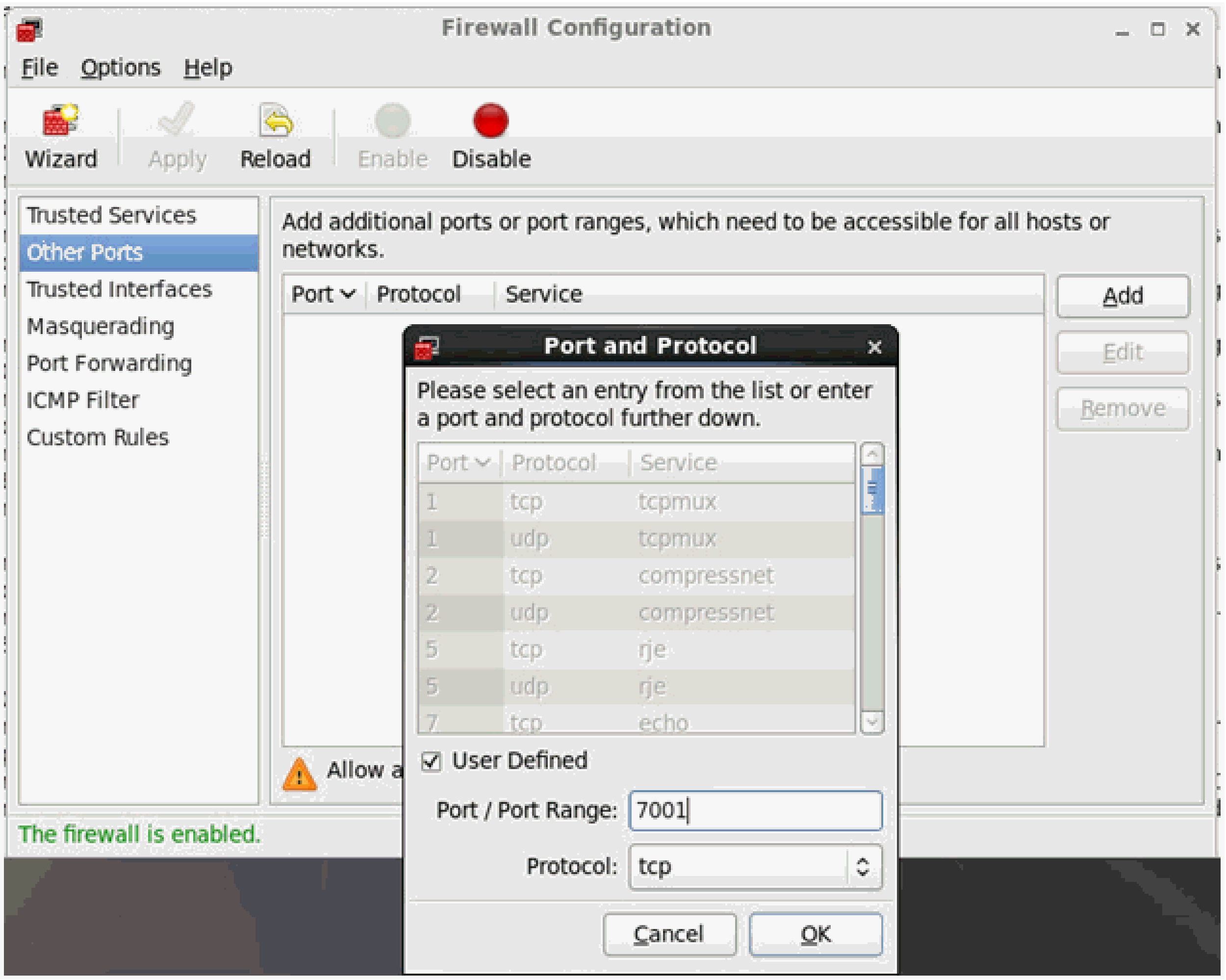
If you need further firewall settings, please modify your configuration by hand, but keep in mind, that this tool is not able to load your custom configuration. If you are using this tool, all your manual changes will be lost. If you do not want to lose your manual configuration changes, leave this program without applying.

Please remember to check your firewall configuration before using it.

Allow access to necessary services, only.

The firewall is enabled. modified label

Click on **Close**.



Add additional ports or port ranges, which need to be accessible for all hosts or networks.

Port	Protocol	Service

Please select an entry from the list or enter a port and protocol further down.

Port	Protocol	Service
1	tcp	tcpmux
1	udp	tcpmux
2	tcp	compressnet
2	udp	compressnet
5	tcp	rje
5	udp	rje
7	tcp	echo

☑ User Defined

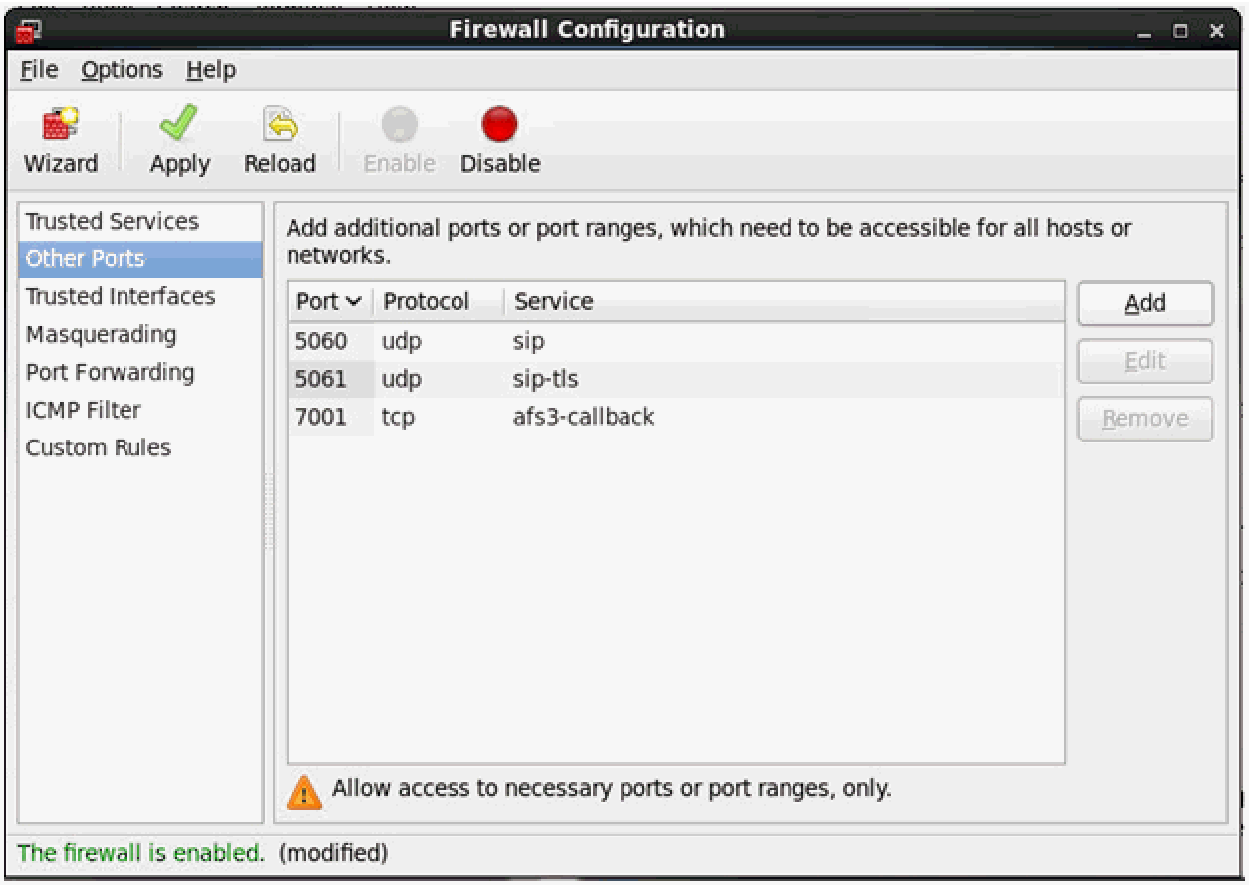
Port / Port Range: 7001

Protocol: tcp

The firewall is enabled.

Select **Other Ports** then click **Add**. Check the **User Defined** box, enter **7001** for **Port/Port Range**, choose **tcp** for **Protocol**, and then click **OK**.

Repeat the steps to add **5060** and **5061** for **Port/Port Range** but with **udp** as **Protocol** for each.



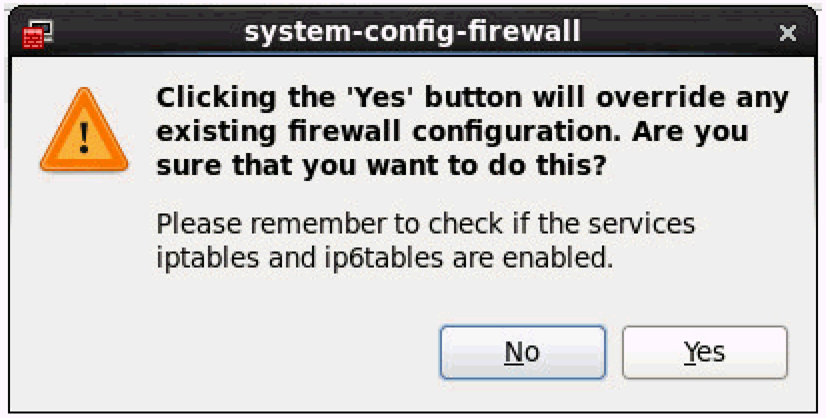
Add additional ports or port ranges, which need to be accessible for all hosts or networks.

Port	Protocol	Service
5060	udp	sip
5061	udp	sip-tls
7001	tcp	afs3-callback

Allow access to necessary ports or port ranges, only.

The firewall is enabled. (modified)

Click on **Apply**.



Clicking the 'Yes' button will override any existing firewall configuration. Are you sure that you want to do this?

Please remember to check if the services iptables and ip6tables are enabled.

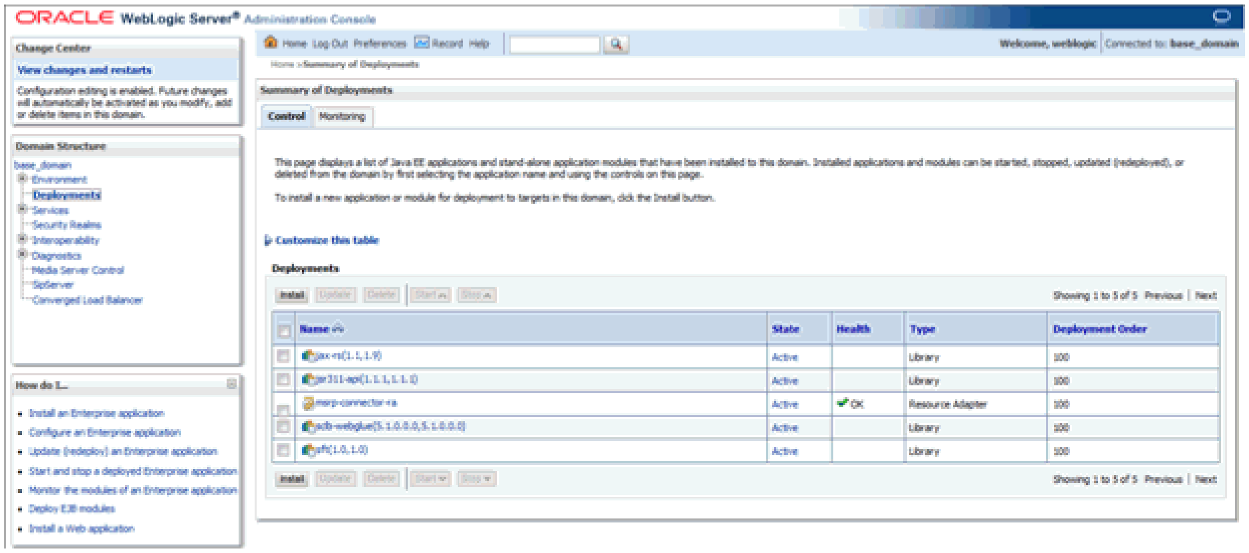
Click on **Yes** to activate the **Firewall Configuration**.

Now exit the **Firewall Configuration**.

OCCAS Verification

Access the **Administration Console** to verify the installation at:

http://<OCCAS5.1.0-IP-ADDRESS>:7001/console


ORACLE WebLogic Server® Administration Console
Home Log Out Preferences Record Help
Welcome, weblogic | Connected to: base_domain
Home >Summary of Deployments
Change Center
View changes and restarts
Domain Structure
base_domain
Environment
Deployments
Services
Security Realms
Interoperability
Diagnostics
Media Server Control
SipServer
Converged Load Balancer
How do I...
Install an Enterprise application
Configure an Enterprise application
Update (redeploy) an Enterprise application
Start and stop a deployed Enterprise application
Monitor the modules of an Enterprise application
Deploy EJB modules
Install a Web application
Summary of Deployments
Control
Monitoring
Customize this table
Deployments
Install
Update
Delete
Start
Stop
Showing 1 to 5 of 5 Previous | Next

Configuration editing is enabled. Future changes will automatically be activated as you modify, add or delete items in this domain.

This page displays a list of Java EE applications and stand-alone application modules that have been installed to this domain. Installed applications and modules can be started, stopped, updated (redeployed), or deleted from the domain by first selecting the application name and using the controls on this page.

To install a new application or module for deployment to targets in this domain, click the Install button.

Name	State	Health	Type	Deployment Order
jax-rs(1.1,1.9)	Active		Library	100
jsr311-api(1.1.1,1.1.1)	Active		Library	100
msrp-connector-ra	Active	✓ OK	Resource Adapter	100
scb-webglue(5.1.0.0.0,5.1.0.0.0)	Active		Library	100
sfs(1.0,1.0)	Active		Library	100

Go to **Domain Structure** and click on **Deployments** to make sure **State** and **Health** are similar to screen shot above.

Installing the JSR 309 Connector

Follow this procedure to get the application (WAR file) in an OCCAS environment to correctly load the JSR 309 Connector (*dlgmsc.jar*).

The JSR 309 Connector supports PowerMedia XMS via the MSML protocol.

Step 1

Extract the *JSR309_MSC3_BuildXX.tar* distribution package for this release which will create a `<Release Package>` directory.

Note: <Release Package> refers to the *.tar* distribution package provided for this release.

Copy all the *.jar* files from `<Release Package>/DlgcJSR309/lib` directory to the OCCAS `<Domain Location>/lib` directory.

Note: <Domain Location> refers to the domain path as specified during OCCAS installation.

JAR Files	Description
dlgmsc.jar *jain-sip-sdp-1.2.91.jar* *jsr173_1.0_api.jar* *log4j-1.2.15.jar* *msmltypes.jar* *slf4j-api-1.7.2.jar* *slf4j-log4j12-1.7.2.jar* *xbean.jar*	JSR 309 Connector implementation for PowerMedia XMS and required third-party software libraries.

Step 2

Set up the properties files.

JSR 309 Connector uses the following properties files:

- dlgcJSR309.properties - used to configure IP addresses and ports for the OCCAS environment using JSR 309 Connector and dedicated PowerMedia XMS platform.
- log4j.properties - used for logging using Simple Logging Facade framework implementation of log4j.

  Note: You can configure logging in the *log4j.properties* file. By default, logging is configured to use INFO level and output to the Console as well as to a *dlgmsc.log* file.

Follow these steps to set up the properties file:

1. Copy the *dlgcJSR309.properties* and *log4j.properties* files from `<Release Package>/DlgcJSR309/properties` directory to the OCCAS `<Domain Location>/config` directory.
2. Edit the *dlgcJSR309.properties* file according to your OCCAS and PowerMedia XMS configuration. The changes will include the OCCAS IP address and SipServlet container port running the JSR 309 Connector as well as the PowerMedia XMS IP address and port:

   # JSR 309 Connector address information (typically same as the SipServlet container)
   ```
   connector.sip.address=xxx.xxx.xxx.xxx
   connector.sip.port=5060
   ```

    # PowerMedia XMS address information
    ```
    mediaserver.1.sip.address=xxx.xxx.xxx.xxx
    mediaserver.1.sip.port=5060
    ```

3. Edit the `<Domain Location>/bin/startWebLogic.sh` OCCAS startup script. Look for the following line:
    ```
    CLASSPATH="${SAVE_CLASSPATH}"
    ```

    Add the following lines directly after:
    ```
    export DIALOGIC_DEMO_PROPERTY_FILE=${DOMAIN_HOME}/applications/dlgc_demos.properties
    export DLG_PROPERTY_FILE=${DOMAIN_HOME}/config/dlgcJSR309.properties
    LOG4J_OPTIONS="-Dlog4j.configuration=${DOMAIN_HOME}/config/log4j.properties"
    XQUERYPATH=/root/Oracle/Middleware/modules/features/weblogic.server.modules.xquery_10.3.6
    .0.jar
    CLASSPATH="${SAVE_CLASSPATH}:${ORCL_HOME}/server/modules/mscontrol.jar:${ORCL_HOME}/serve
    r/lib/jsr309-descriptor-binding.jar:${XQUERYPATH}"
    ```

4. Add the following line in OCCAS startup script to enable some of the relevant items and to disable serialization (highlighted in bold):
    ```
    ${LOG4J_OPTIONS} -Dlog4j.debug -Dwlss.local.serialization=false
    ```

    from:
    ```
    if [ "${WLS_REDIRECT_LOG}" = "" ] ; then
            echo "Starting WLS with line:"
            echo "${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} -Dweblogic.Name=${SERVER_NAME}
    -Djava.security.policy=${WL_HOME}/server/lib/weblogic.policy ${JAVA_OPTIONS}
    ${PROXY_SETTINGS} ${SERVER_CLASS}"
            ${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} -Dweblogic.Name=${SERVER_NAME} -
    Djava.security.policy=${WL_HOME}/server/lib/weblogic.policy ${JAVA_OPTIONS}
    ${PROXY_SETTINGS} ${SERVER_CLASS}
    else
            echo "Redirecting output from WLS window to ${WLS_REDIRECT_LOG}"
            ${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} -Dweblogic.Name=${SERVER_NAME} -
    Djava.security.policy=${WL_HOME}/server/lib/weblogic.policy ${JAVA_OPTIONS}
    ${PROXY_SETTINGS} ${SERVER_CLASS}  >"${WLS_REDIRECT_LOG}" 2>&1
    fi
    ```

    to:
    ```
    if [ "${WLS_REDIRECT_LOG}" = "" ] ; then
            echo "Starting WLS with line:"
            echo "${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} -Dweblogic.Name=${SERVER_NAME}
    ${LOG4J_OPTIONS} -Dlog4j.debug -Dwlss.local.serialization=false -
    Djava.security.policy=${WL_HOME}/server/lib/weblogic.policy ${JAVA_OPTIONS}
    ${PROXY_SETTINGS} ${SERVER_CLASS}"
            ${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} -Dweblogic.Name=${SERVER_NAME}
    ${LOG4J_OPTIONS} -Dlog4j.debug -Dwlss.local.serialization=false -
    Djava.security.policy=${WL_HOME}/server/lib/weblogic.policy ${JAVA_OPTIONS}
    ${PROXY_SETTINGS} ${SERVER_CLASS}
    else
            echo "Redirecting output from WLS window to ${WLS_REDIRECT_LOG}"
            ${JAVA_HOME}/bin/java ${JAVA_VM} ${MEM_ARGS} -Dweblogic.Name=${SERVER_NAME}
    ${LOG4J_OPTIONS} -Dlog4j.debug -Dwlss.local.serialization=false -
    Djava.security.policy=${WL_HOME}/server/lib/weblogic.policy ${JAVA_OPTIONS}
    ${PROXY_SETTINGS} ${SERVER_CLASS}  >"${WLS_REDIRECT_LOG}" 2>&1
    ```

    For more details on disabling serialization, refer to Disabling Serialization.

5. Save the startup script file then restart the WebLogic Server.

Installing and Configuring the Test Servlets

Make sure that desired application exists in `<Domain Location>/application`.

Access the **Administration Console** at:

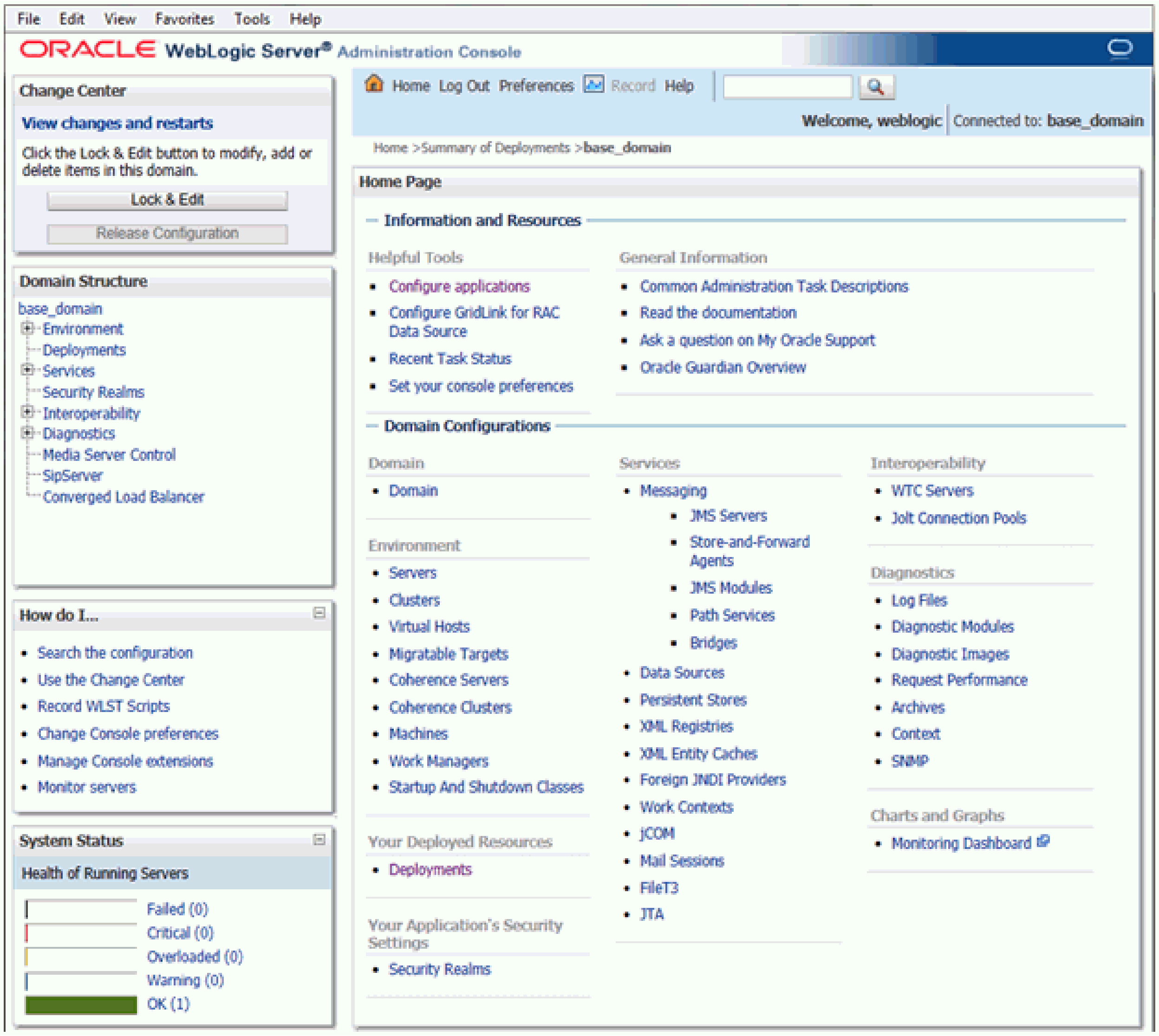
http://<OCCAS5.1.0-IP-ADDRESS>:7001/console

Use the username/password as set during configuration.

The following is used as an example:

Name: weblogic

User password: Webl0gic!! ("0" is a zero)



Click the Lock & Edit button to modify, add or delete items in this domain.

Home >Summary of Deployments >base_domain

How do I...
- Search the configuration
- Use the Change Center
- Record WLST Scripts
- Change Console preferences
- Manage Console extensions
- Monitor servers

System Status — Health of Running Servers

Failed (0)	
Critical (0)	
Overloaded (0)	
Warning (0)	
OK (1)	🟩

Helpful Tools
- Configure applications
- Configure GridLink for RAC Data Source
- Recent Task Status
- Set your console preferences

General Information
- Common Administration Task Descriptions
- Read the documentation
- Ask a question on My Oracle Support
- Oracle Guardian Overview

Domain
- Domain

Environment
- Servers
- Clusters
- Virtual Hosts
- Migratable Targets
- Coherence Servers
- Coherence Clusters
- Machines
- Work Managers
- Startup And Shutdown Classes

Your Deployed Resources
- Deployments

Your Application's Security Settings
- Security Realms

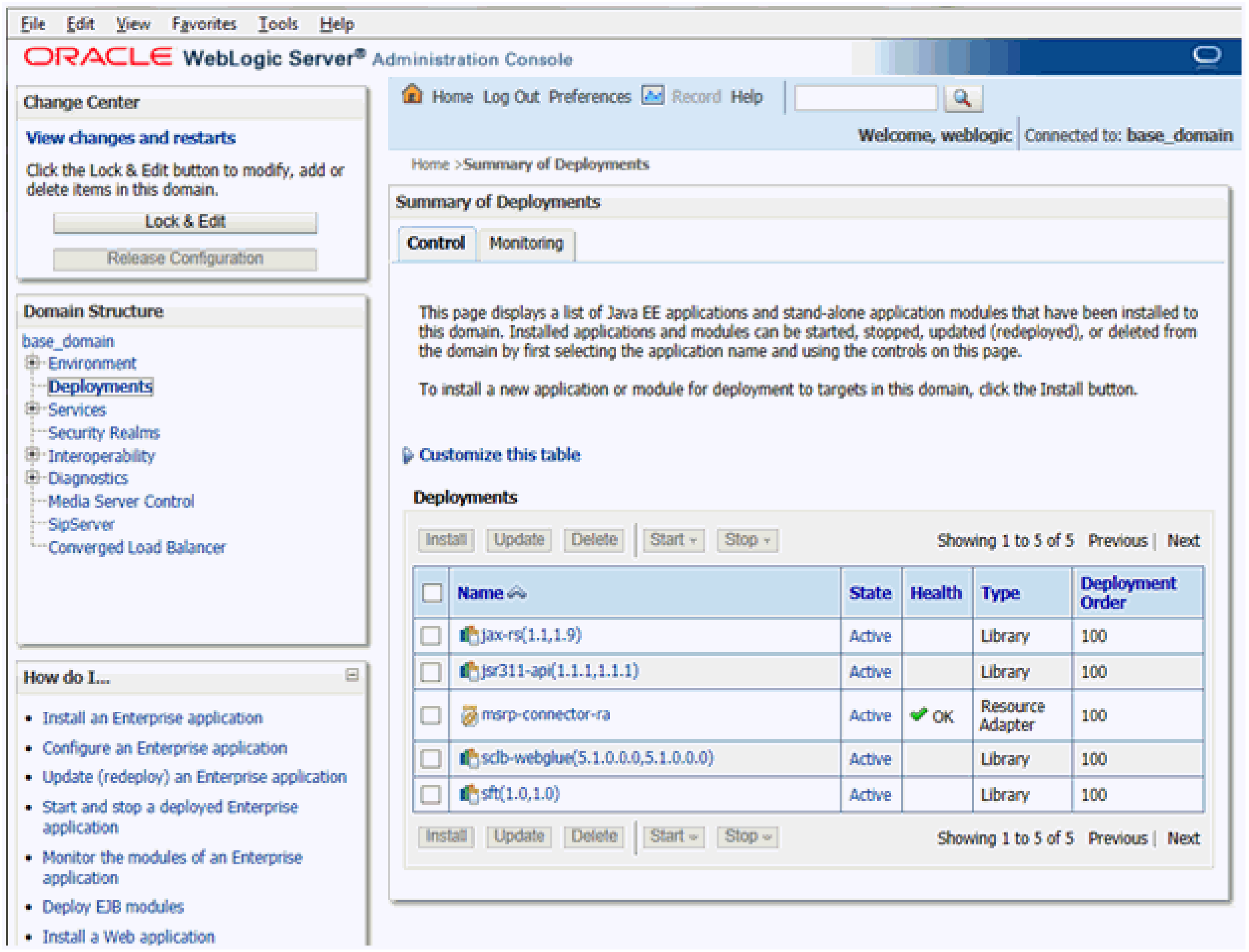
Services
- Messaging
  - JMS Servers
  - Store-and-Forward Agents
  - JMS Modules
  - Path Services
  - Bridges
- Data Sources
- Persistent Stores
- XML Registries
- XML Entity Caches
- Foreign JNDI Providers
- Work Contexts
- jCOM
- Mail Sessions
- FileT3
- JTA

Interoperability
- WTC Servers
- Jolt Connection Pools

Diagnostics
- Log Files
- Diagnostic Modules
- Diagnostic Images
- Request Performance
- Archives
- Context
- SNMP

Charts and Graphs
- Monitoring Dashboard

Go to **Deployments** under **Domain Structure** then click on **Lock & Edit**.


File Edit View Favorites Tools Help
ORACLE WebLogic Server® Administration Console
Home Log Out Preferences Record Help
Welcome, weblogic | Connected to: base_domain
Home >Summary of Deployments

Change Center

View changes and restarts

Click the Lock & Edit button to modify, add or delete items in this domain.

Lock & Edit

Release Configuration

Domain Structure

base_domain
- Environment
- Deployments
- Services
- Security Realms
- Interoperability
- Diagnostics
- Media Server Control
- SipServer
- Converged Load Balancer

How do I...

- Install an Enterprise application
- Configure an Enterprise application
- Update (redeploy) an Enterprise application
- Start and stop a deployed Enterprise application
- Monitor the modules of an Enterprise application
- Deploy EJB modules
- Install a Web application

Summary of Deployments

Control | Monitoring

This page displays a list of Java EE applications and stand-alone application modules that have been installed to this domain. Installed applications and modules can be started, stopped, updated (redeployed), or deleted from the domain by first selecting the application name and using the controls on this page.

To install a new application or module for deployment to targets in this domain, click the Install button.

Customize this table

Deployments

Install | Update | Delete | Start | Stop — Showing 1 to 5 of 5 Previous | Next

	Name	State	Health	Type	Deployment Order
☐	jax-rs(1.1,1.9)	Active		Library	100
☐	jsr311-api(1.1.1,1.1.1)	Active		Library	100
☐	msrp-connector-ra	Active	✓ OK	Resource Adapter	100
☐	sclb-webglue(5.1.0.0.0,5.1.0.0.0)	Active		Library	100
☐	sft(1.0,1.0)	Active		Library	100

Install | Update | Delete | Start | Stop — Showing 1 to 5 of 5 Previous | Next

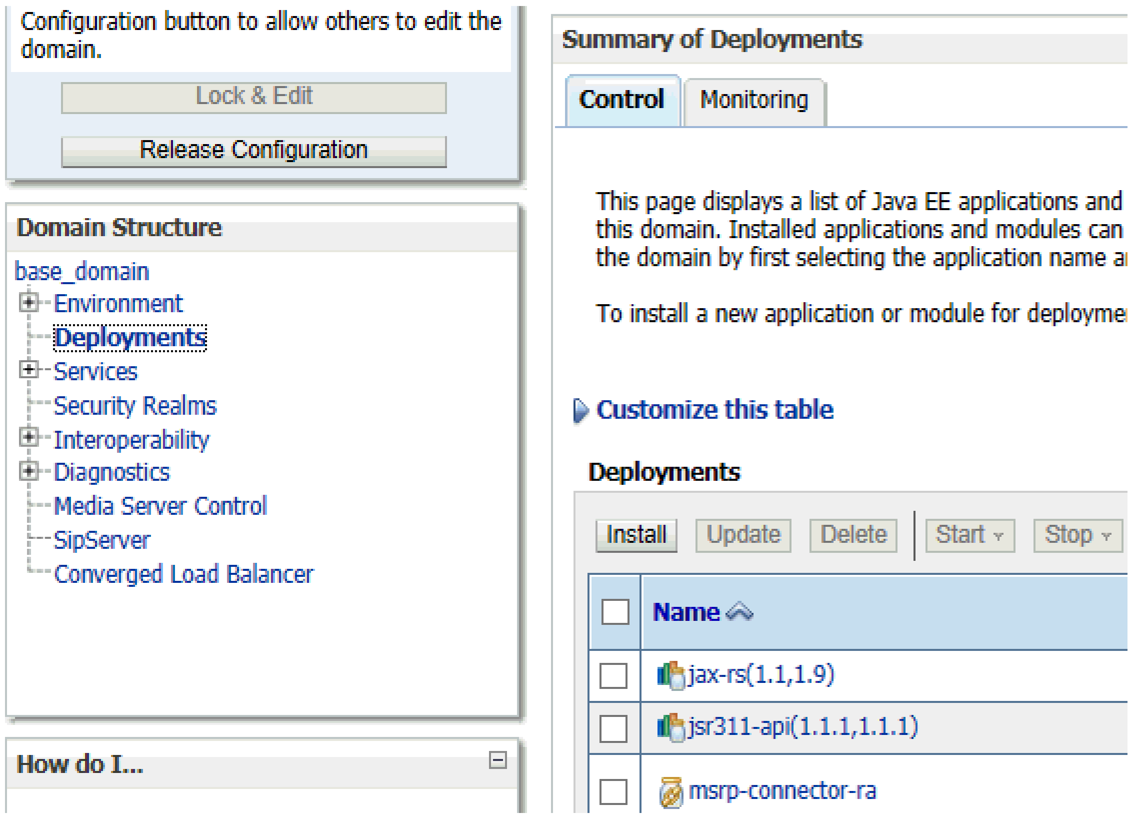
Make sure the existing services are as shown to verify that the OCCAS components have started.

If OCCAS was installed in **Production Mode**, an additional process will need to be followed when making changes to the configuration and setup. This additional process includes **Lock & Edit** to change the configuration and complete the required changes, followed by using **Release Configuration**.

If OCCAS was installed in **Development Mode**, the process is done automatically. Skip the steps that discuss **Lock & Edit**, **Release Configuration**, and activating of an application.

The following examples are based on OCCAS being installed in **Production Mode**.

Click on **Lock & Edit**.

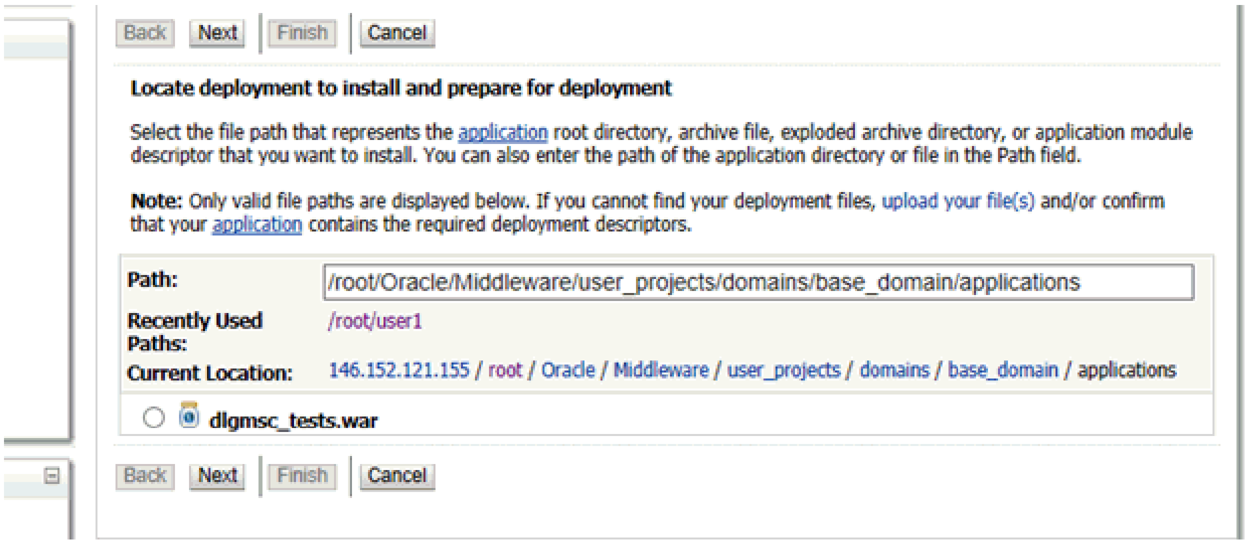

Configuration button to allow others to edit the domain.

This page displays a list of Java EE applications and
this domain. Installed applications and modules can
the domain by first selecting the application name a

To install a new application or module for deployme

	Name
☐	jax-rs(1.1,1.9)
☐	jsr311-api(1.1.1,1.1.1)
☐	msrp-connector-ra

Click on **Install** under **Deployments**.



Locate deployment to install and prepare for deployment

Select the file path that represents the application root directory, archive file, exploded archive directory, or application module descriptor that you want to install. You can also enter the path of the application directory or file in the Path field.

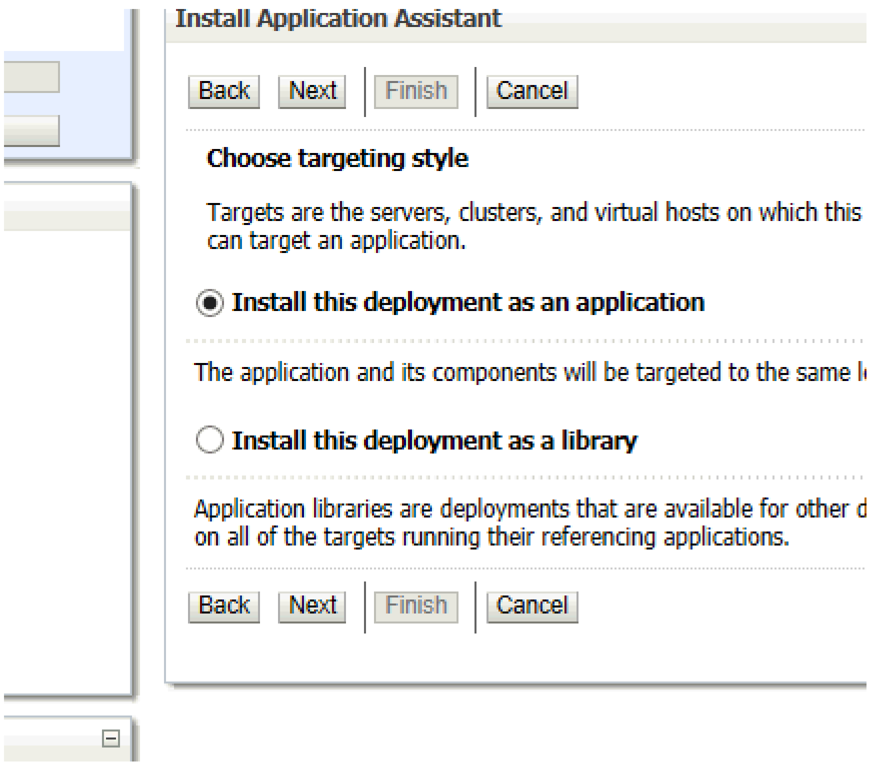
Note: Only valid file paths are displayed below. If you cannot find your deployment files, upload your file(s) and/or confirm that your application contains the required deployment descriptors.

Path:	/root/Oracle/Middleware/user_projects/domains/base_domain/applications
Recently Used Paths:	/root/user1
Current Location:	146.152.121.155 / root / Oracle / Middleware / user_projects / domains / base_domain / applications

○ dlgmsc_tests.war

Navigate to `<Domain Location>/applications` under **Current Location**.

Select **dlgmsc_tests.war** then click on **Next**.


Install Application Assistant

Choose targeting style

Targets are the servers, clusters, and virtual hosts on which this can target an application.

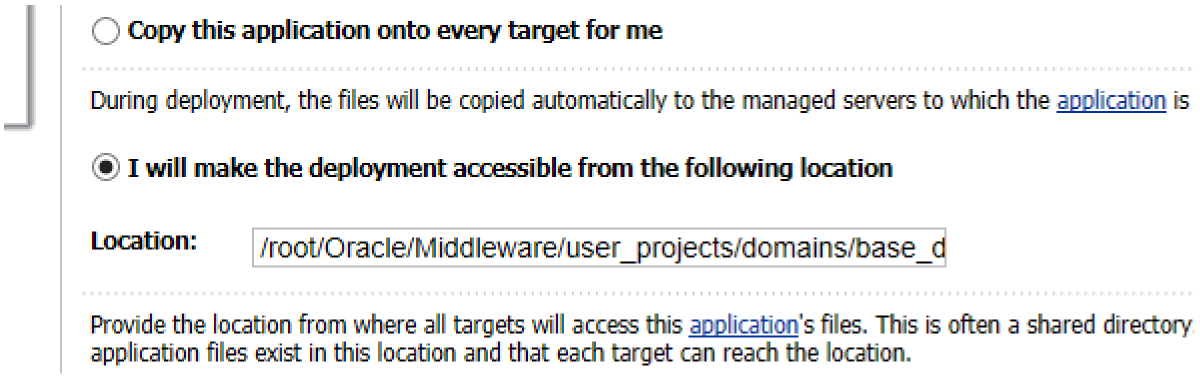
(●) Install this deployment as an application

The application and its components will be targeted to the same l

( ) Install this deployment as a library

Application libraries are deployments that are available for other d on all of the targets running their referencing applications.

Select **Install this deployment as an application** then click on **Next**.


( ) Copy this application onto every target for me

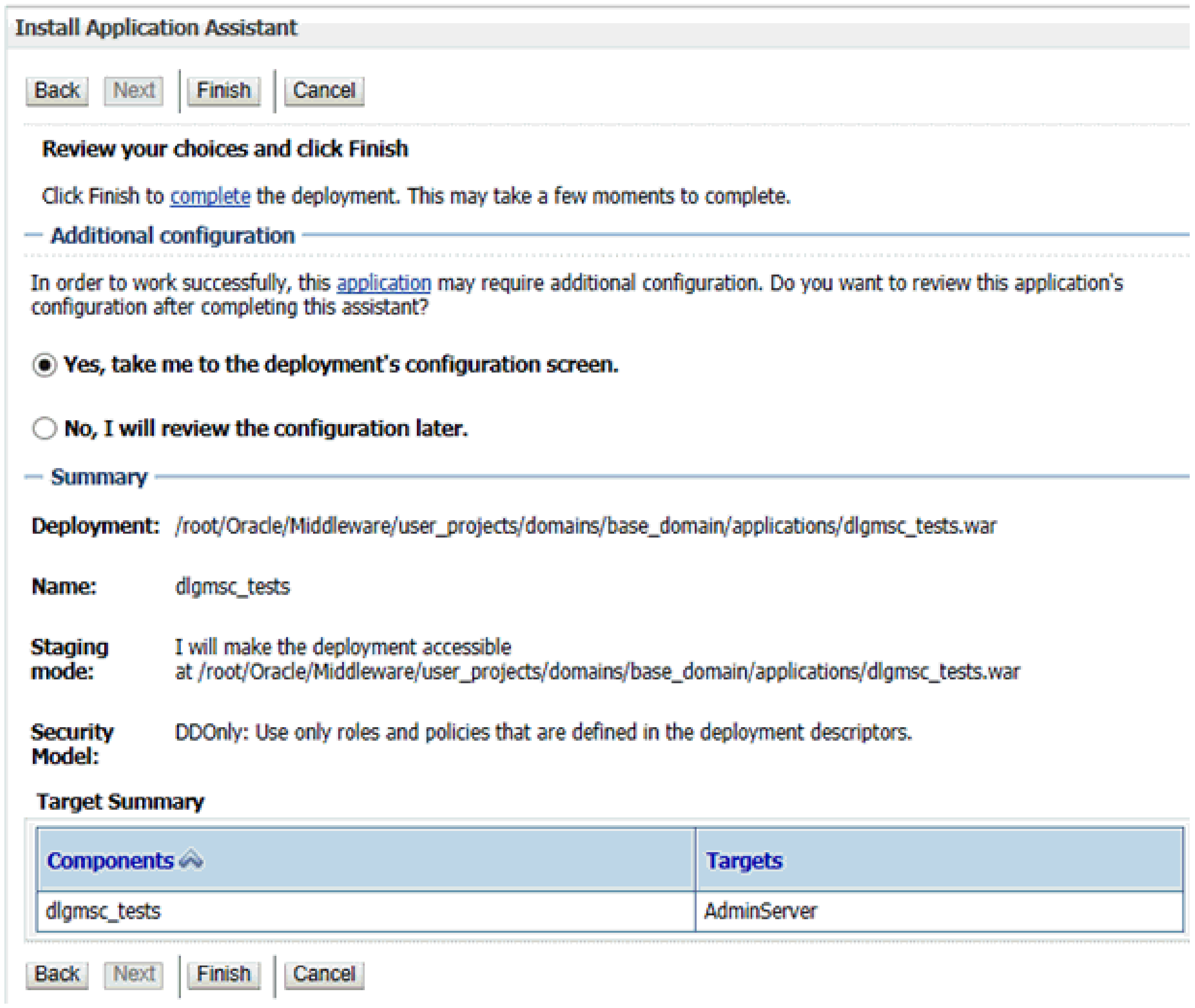
During deployment, the files will be copied automatically to the managed servers to which the application is

(●) I will make the deployment accessible from the following location

Location: /root/Oracle/Middleware/user_projects/domains/base_d

Provide the location from where all targets will access this application's files. This is often a shared directory application files exist in this location and that each target can reach the location.

Select **I will make the deployment accessible from the following location** then click on **Next**.



Review your choices and click Finish

Click Finish to complete the deployment. This may take a few moments to complete.

Additional configuration

In order to work successfully, this application may require additional configuration. Do you want to review this application's configuration after completing this assistant?

[✓ Yes, take me to the deployment's configuration screen.]

No, I will review the configuration later.

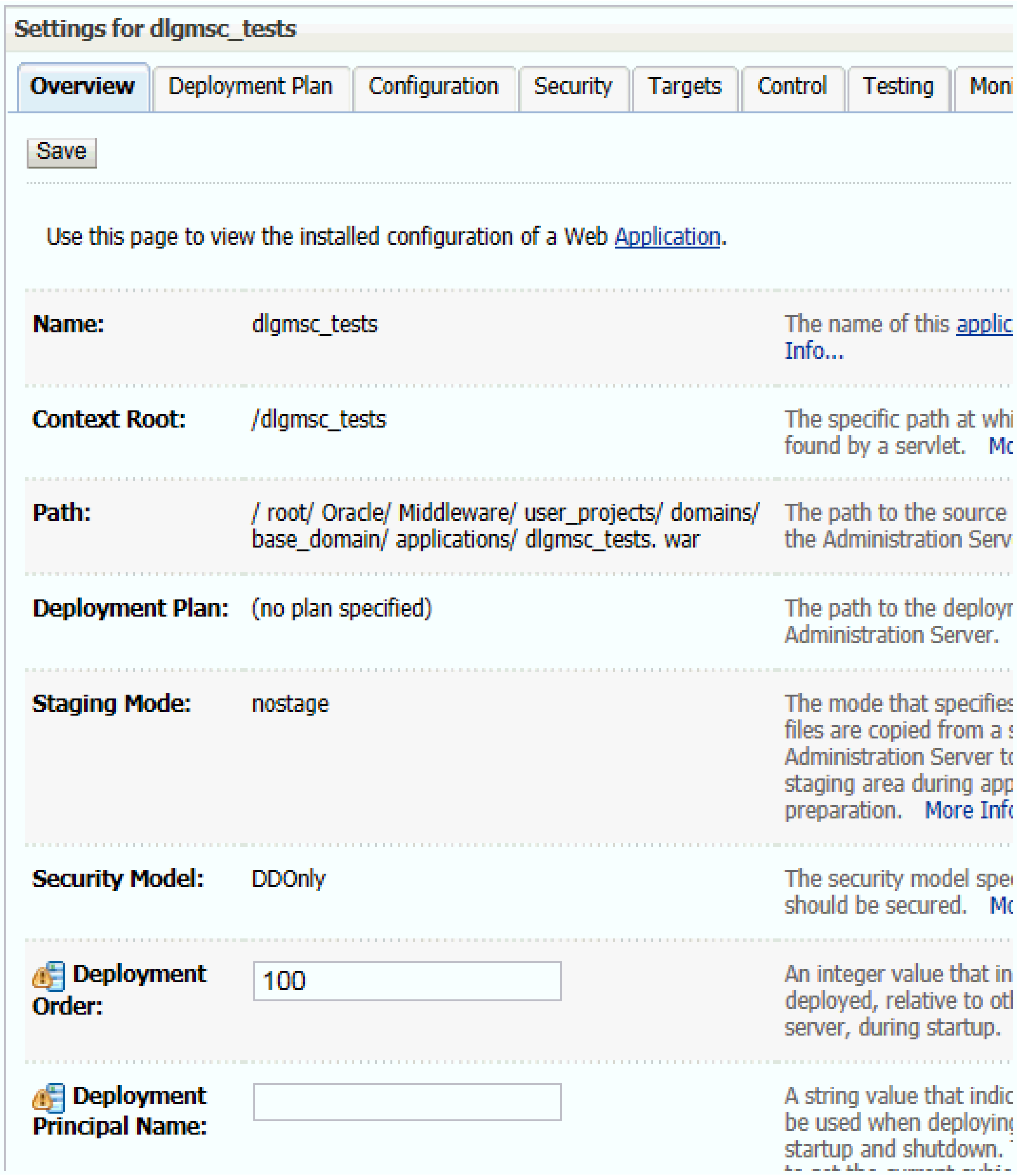
Summary

Deployment:	/root/Oracle/Middleware/user_projects/domains/base_domain/applications/dlgmsc_tests.war
Name:	dlgmsc_tests
Staging mode:	I will make the deployment accessible at /root/Oracle/Middleware/user_projects/domains/base_domain/applications/dlgmsc_tests.war
Security Model:	DDOnly: Use only roles and policies that are defined in the deployment descriptors.

Target Summary

Components	Targets
dlgmsc_tests	AdminServer

Click on **Finish**.


Settings for dlgmsc_tests

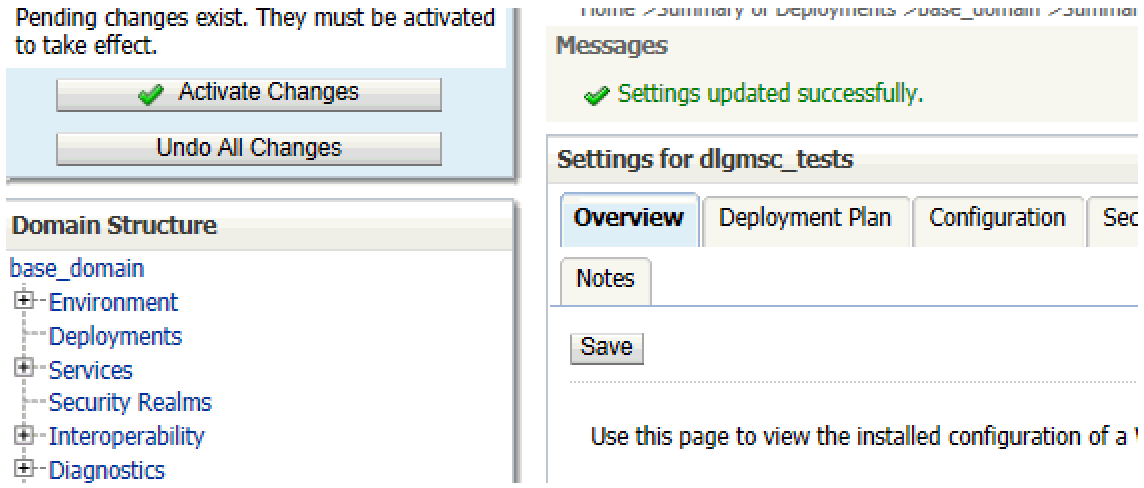
Overview | Deployment Plan | Configuration | Security | Targets | Control | Testing | Moni

Save

Use this page to view the installed configuration of a Web Application.

Name:	dlgmsc_tests	The name of this applic Info...
Context Root:	/dlgmsc_tests	The specific path at whi found by a servlet. Mo
Path:	/ root/ Oracle/ Middleware/ user_projects/ domains/ base_domain/ applications/ dlgmsc_tests. war	The path to the source the Administration Serv
Deployment Plan:	(no plan specified)	The path to the deployr Administration Server.
Staging Mode:	nostage	The mode that specifies files are copied from a s Administration Server to staging area during app preparation. More Info
Security Model:	DDOnly	The security model spe should be secured. Mo
Deployment Order:	100	An integer value that in deployed, relative to ot server, during startup.
Deployment Principal Name:		A string value that indic be used when deploying startup and shutdown.

Click on **Save**.

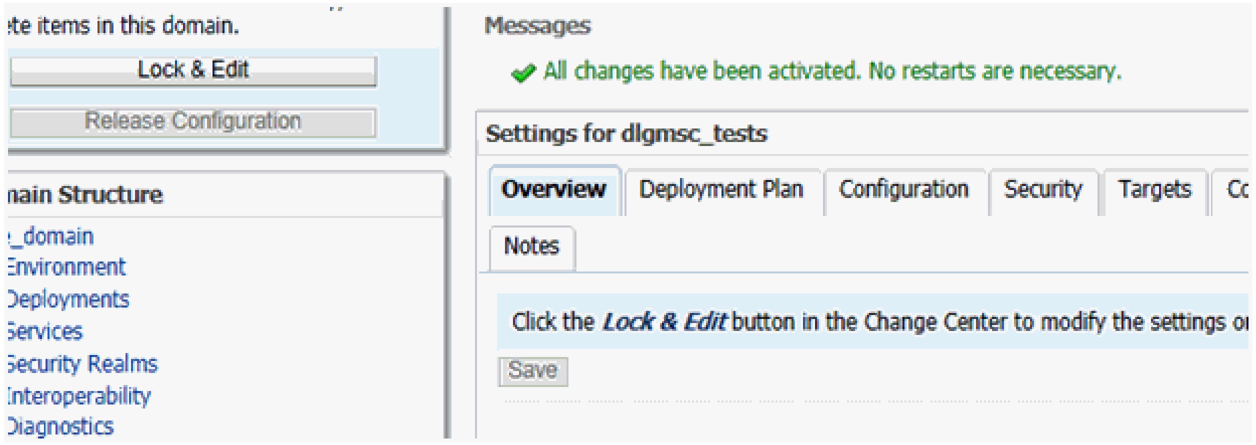

Pending changes exist. They must be activated to take effect.

Settings updated successfully.

Use this page to view the installed configuration of a

Click on **Activate Changes** once the "Settings updated successfully" message appears.

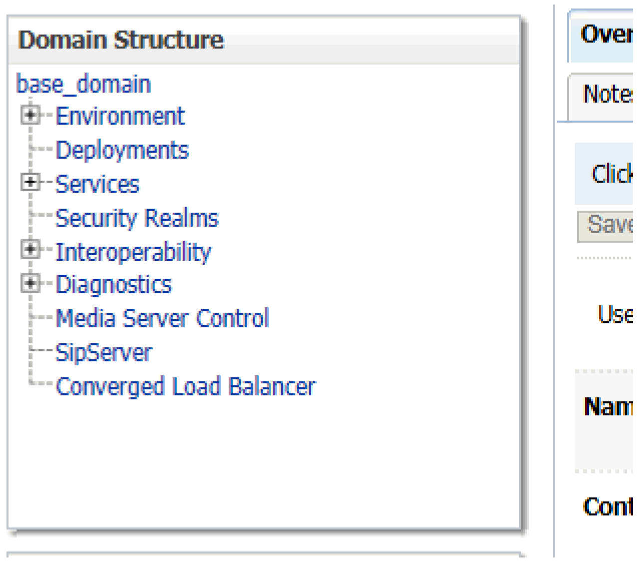
Note: This step is not applicable when OCCAS was installed in **Development Mode**.



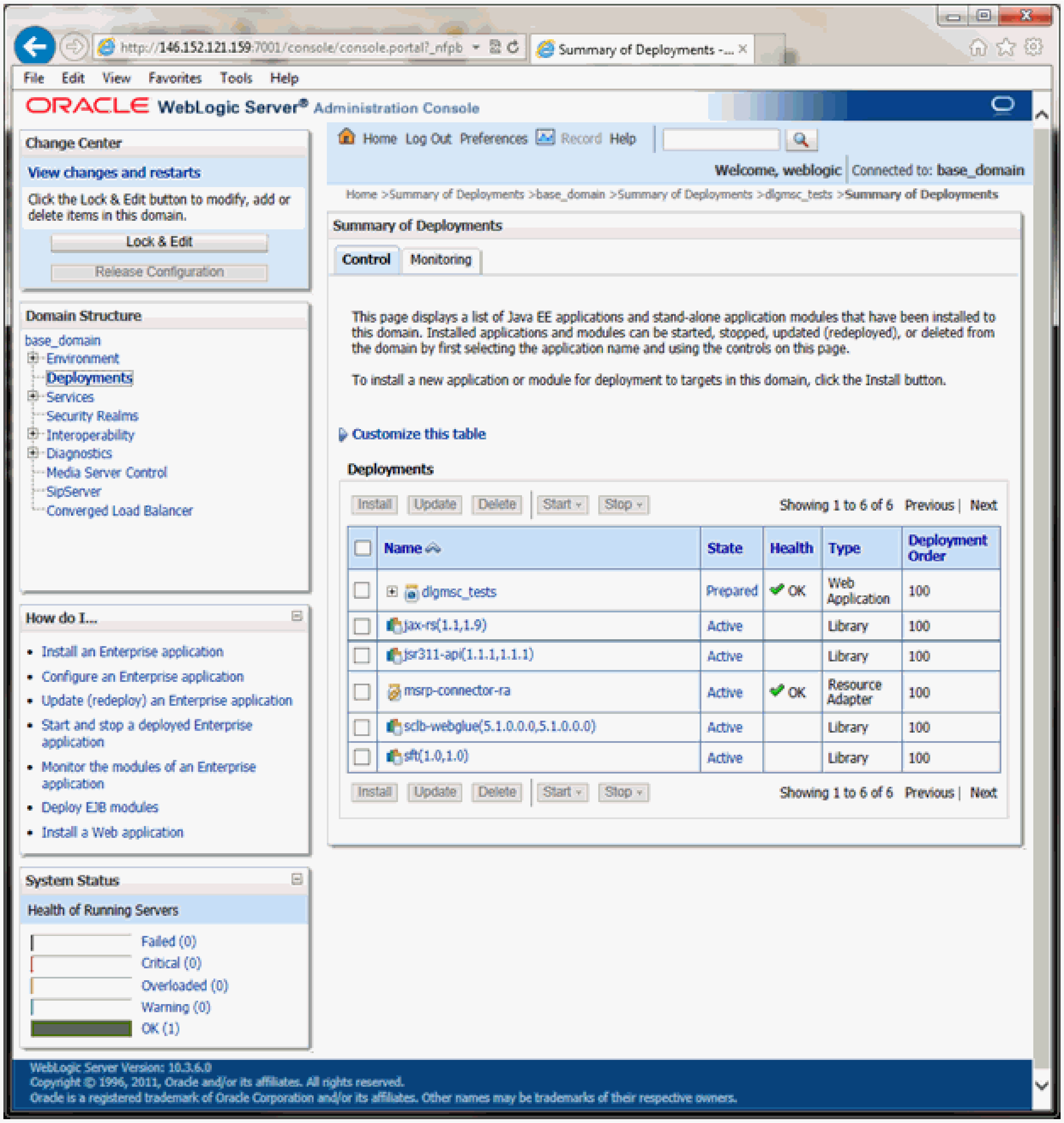
All changes have been activated. No restarts are necessary.

Click the *Lock & Edit* button in the Change Center to modify the settings o

Make sure the "All changes have been activated. No restarts are necessary" message appears.



Go back to **Deployments** under **Domain Structure**.



http://146.152.121.159:7001/console/console.portal?_nfpb

Summary of Deployments -...

File Edit View Favorites Tools Help

ORACLE WebLogic Server® Administration Console

Home Log Out Preferences Record Help

Welcome, weblogic | Connected to: base_domain

Home >Summary of Deployments >base_domain >Summary of Deployments >dlgmsc_tests >**Summary of Deployments**

Change Center

View changes and restarts

Click the Lock & Edit button to modify, add or delete items in this domain.

Lock & Edit

Release Configuration

Domain Structure

base_domain
- Environment
- Deployments
- Services
- Security Realms
- Interoperability
- Diagnostics
- Media Server Control
- SipServer
- Converged Load Balancer

How do I...

- Install an Enterprise application
- Configure an Enterprise application
- Update (redeploy) an Enterprise application
- Start and stop a deployed Enterprise application
- Monitor the modules of an Enterprise application
- Deploy EJB modules
- Install a Web application

System Status

Health of Running Servers

- Failed (0)
- Critical (0)
- Overloaded (0)
- Warning (0)
- OK (1)

Summary of Deployments

Control | Monitoring

This page displays a list of Java EE applications and stand-alone application modules that have been installed to this domain. Installed applications and modules can be started, stopped, updated (redeployed), or deleted from the domain by first selecting the application name and using the controls on this page.

To install a new application or module for deployment to targets in this domain, click the Install button.

Customize this table

Deployments

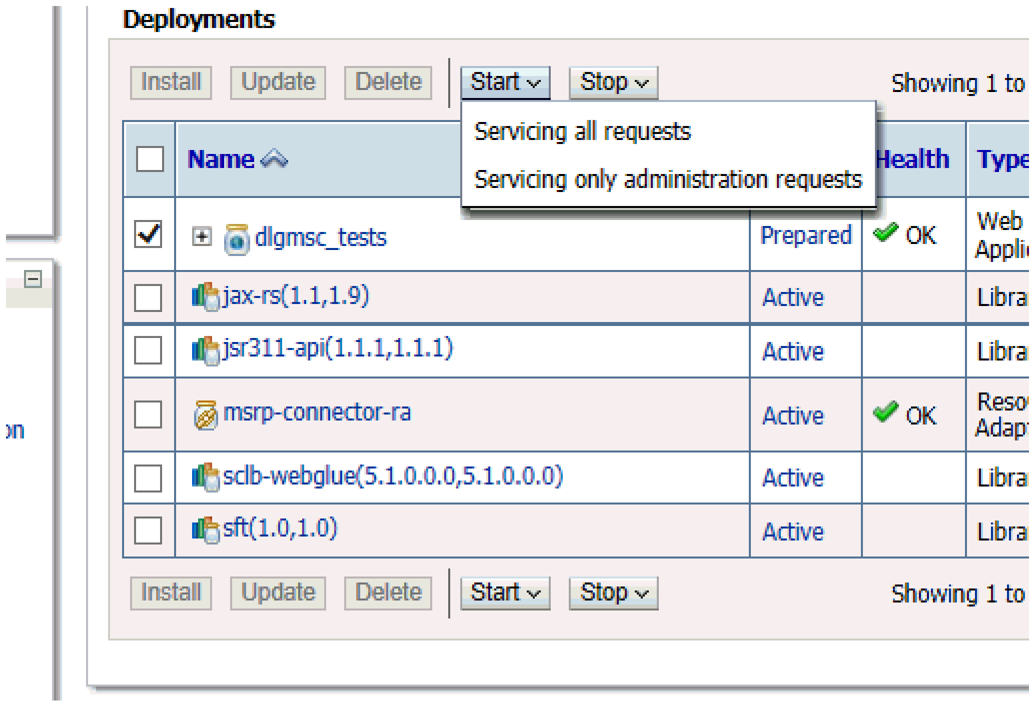
Install | Update | Delete | Start | Stop — Showing 1 to 6 of 6 Previous | Next

	Name	State	Health	Type	Deployment Order
☐	dlgmsc_tests	Prepared	✓ OK	Web Application	100
☐	jax-rs(1.1,1.9)	Active		Library	100
☐	jsr311-api(1.1.1,1.1.1)	Active		Library	100
☐	msrp-connector-ra	Active	✓ OK	Resource Adapter	100
☐	sclb-webglue(5.1.0.0.0,5.1.0.0.0)	Active		Library	100
☐	sft(1.0,1.0)	Active		Library	100

Install | Update | Delete | Start | Stop — Showing 1 to 6 of 6 Previous | Next

WebLogic Server Version: 10.3.6.0
Copyright © 1996, 2011, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

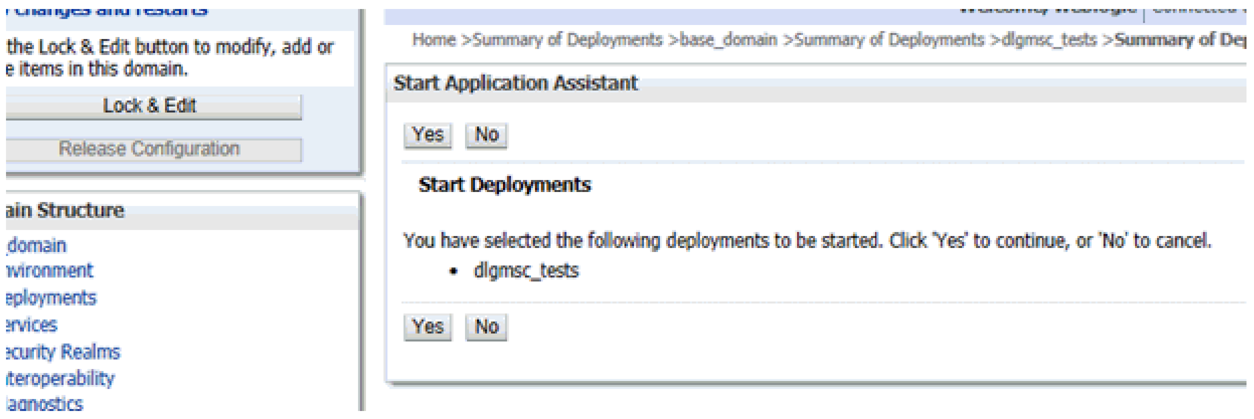
The **Deployments** along with its **State** and **Health** are displayed. Verify that the installed *dlgmsc_tests.war* application is in **Prepared** state.



	Name		Health	Type
✓	dlgmsc_tests	Prepared	✓ OK	Web Appli
	jax-rs(1.1,1.9)	Active		Libra
	jsr311-api(1.1.1,1.1.1)	Active		Libra
	msrp-connector-ra	Active	✓ OK	Reso Adap
	sclb-webglue(5.1.0.0.0,5.1.0.0.0)	Active		Libra
	sft(1.0,1.0)	Active		Libra

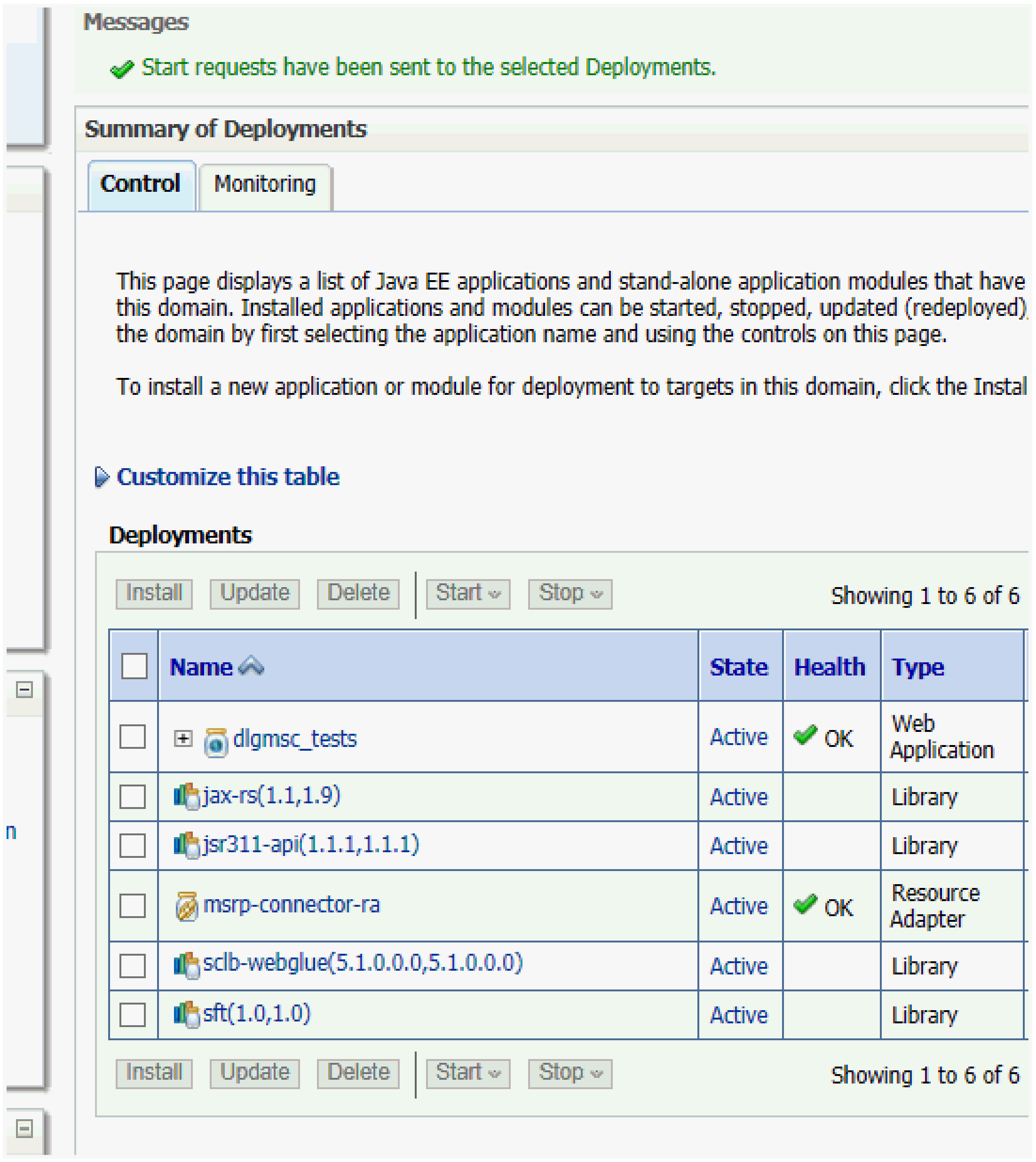
Once in **Prepared** state, select *dlgmsc_tests.war* application and click on **Start** then **Servicing all requests**.

Note: This step is not applicable when OCCAS was installed in **Development Mode**.


the Lock & Edit button to modify, add or
e items in this domain.
Lock & Edit
Release Configuration
ain Structure
domain
wironment
eployments
ervices
ecurity Realms
teroperability
iagnostics
Home >Summary of Deployments >base_domain >Summary of Deployments >dlgmsc_tests >Summary of De
Start Application Assistant
Yes
No
Start Deployments

You have selected the following deployments to be started. Click 'Yes' to continue, or 'No' to cancel.
- dlgmsc_tests

Click on **Yes**.


Messages

✓ Start requests have been sent to the selected Deployments.

Summary of Deployments

This page displays a list of Java EE applications and stand-alone application modules that have
this domain. Installed applications and modules can be started, stopped, updated (redeployed)
the domain by first selecting the application name and using the controls on this page.

To install a new application or module for deployment to targets in this domain, click the Instal

Deployments

Showing 1 to 6 of 6

	Name	State	Health	Type
☐	dlgmsc_tests	Active	✓ OK	Web Application
☐	jax-rs(1.1,1.9)	Active		Library
☐	jsr311-api(1.1.1,1.1.1)	Active		Library
☐	msrp-connector-ra	Active	✓ OK	Resource Adapter
☐	sclb-webglue(5.1.0.0.0,5.1.0.0.0)	Active		Library
☐	sft(1.0,1.0)	Active		Library

Showing 1 to 6 of 6

Make sure the "Start requests have been sent to the selected Deployments" message appears. Verify that deployed *dlgmsc_tests.war* application is in **Active** state.

Running the Test Servlets

Sample test servlets use custom demo prompts which need to be installed on dedicated PowerMedia XMS platform for this configuration. You can locate and install the demo prompts by performing the following:

1. Copy all the *.tar* files inside the `<Release Package>/DlgcJSR309DemoPrompts` directory to the PowerMedia XMS machine under `/var/lib/xms/media/en_US/verification` directory.
2. Untar each of the media files using the command; `tar -xvf <Filename>.tar`.

To verify that your installation is successful, you can dial into OCCAS and run a simple demo included with the JSR 309 Connector which will play a file.

To test the application, dial the following:

```
DlgcPlayerDemo@<OCCAS5.1.0-IP-ADDRESS>
```

If it is successful, you will hear a "Please contact your service provider" prompt.