

openCRX Installation Guide for Sun Application Server 8.2

Version 1.11.0



www.opencrx.org

License

The contents of this file are subject to a BSD license (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at <http://www.opencrx.org/license.htm>

Copyright 2007 © CRIXP Corp. All rights reserved.



Table of Contents

1	About this Book.....	3
1.1	Who this book is for.....	3
1.2	What do you need to understand this book.....	3
1.3	Tips, Warnings, etc.....	3
2	Prerequisites.....	4
3	Installing openCRX for Sun Application Server 8.2.....	5
4	Installing Libraries.....	6
5	Configuring the Java Virtual Machine.....	7
6	Configuring server.policy.....	9
7	Configuring the Datasource.....	10
8	Configuring Security.....	14
9	Deploying openCRX.....	17
10	Final Steps.....	20

List of Figures

Figure 1:	Sun Application Server 8.2 Admin Console.....	5
Figure 2:	Configure the Java Virtual Machine.....	7
Figure 3:	Create a Connection Pool.....	10
Figure 4:	Enter the data source class name.....	11
Figure 5:	Specify the database-specific properties.....	12
Figure 6:	Create a JDBC resource.....	13
Figure 7:	Manage users.....	14
Figure 8:	Add user guest.....	15
Figure 9:	List of users who have access to openCRX.....	16
Figure 10:	Deploy the opencrx-core-CRX-App EAR.....	17
Figure 11:	Accept the default values in deploy enterprise application.....	18
Figure 12:	Deploy the openCRX web application. Select Precompile JSP.....	19
Figure 13:	The openCRX applications must be deployed and enabled.....	20

List of Listings

Listing 1:	Listing of domain1.log.properties.....	8
Listing 2:	File permissions in server.policy before modification.....	9
Listing 3:	File permissions in server.policy before modification.....	9

1 About this Book

This book describes the installation of *openCRX* for *Sun Application Server 8.2*.

1.1 Who this book is for

The intended audience are *openCRX* and application server system administrators.

1.2 What do you need to understand this book

This book describes the installation of *openCRX* for *Sun Application Server 8.2*. The book assumes that you are familiar with *Sun Application Server 8.2* administration and deployment concepts.

1.3 Tips, Warnings, etc.

We make use the following pictograms:



Information provided as a “Tip” might be helpful for various reasons: time savings, risk reduction, etc. - it goes without saying that we advise to follow our guides meticulously

meticulous \muh-TIK-yuh-luhs\, *adjective*:
Extremely or excessively careful about details.



You should carefully read information marked with “Important”. Ignoring such information is typically not a good idea.



Warnings should not be ignored (risk of data loss, etc.)

2 Prerequisites

As a first you must download and install the following software:

- Install *Sun Application Server 8.2*.
- Download **openMDX** from *here*
(http://sourceforge.net/project/showfiles.php?group_id=75132).
- Download **openCRX** from *here*
(http://sourceforge.net/project/showfiles.php?group_id=95219).



Sun Application Server 8.2 comes with *JRE-1.5*. Download the *JRE-1.5* version of *openCRX* and *openMDX* (e.g. *opencrx-1.11.0-core.CRX.jre-1.5.zip*).



Before you can install *openCRX* for *SunAS* you must install the database as described in the *openCRX* database installation guides. If you have successfully installed the *openCRX* database you are ready to continue with the *Sun Application Server 8.2* setup.

3 Installing openCRX for *Sun Application Server 8.2*

After installing *Sun Application Server 8.2* you should be able to start and stop it and launch the *Administrative Console* as shown below:

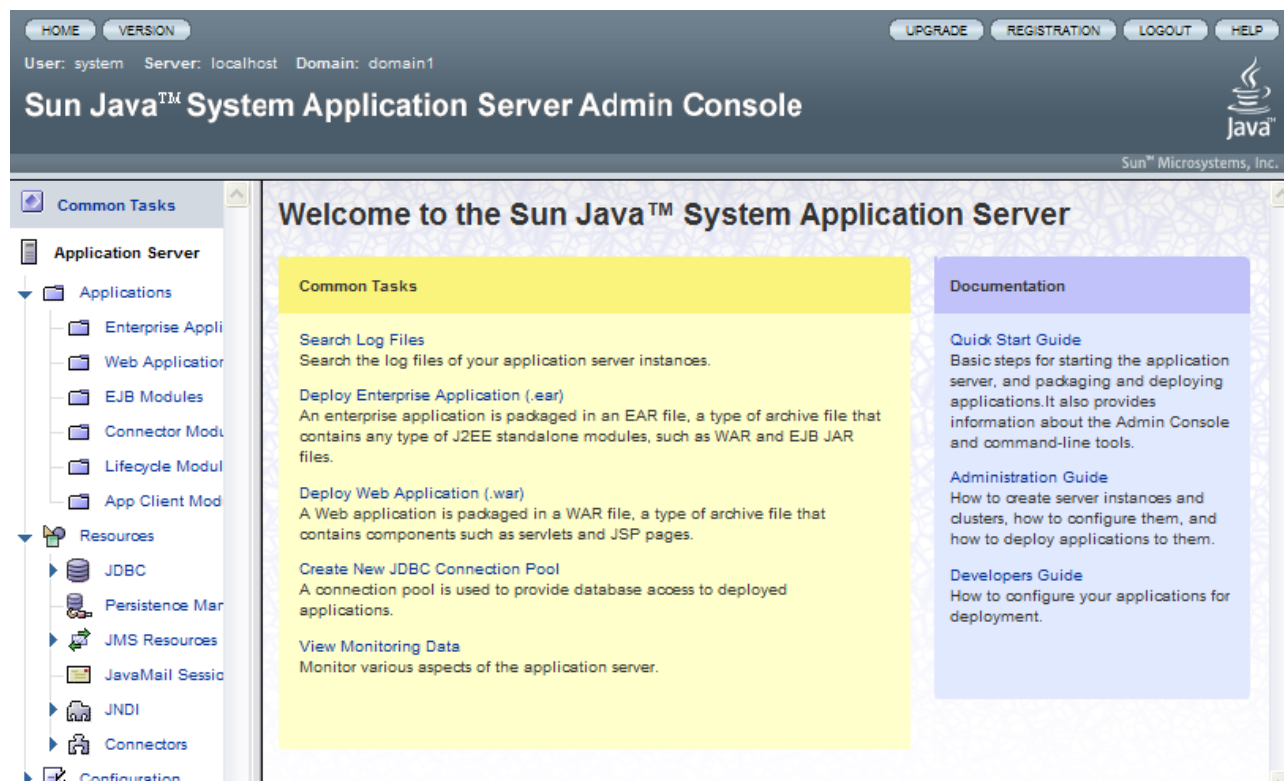


Figure 1: *Sun Application Server 8.2 Admin Console.*

The *openCRX* installation requires the following steps:

- Configure properties for the *JVM* required by *openCRX* and *openMDX*.
- Create and configure the datasource to access the *openCRX* database.
- Configure security.
- Deploy the *openCRX* application enterprise archives.
- Start and test *openCRX*.

4 Installing Libraries

As a first step you must install the *openMDX* and the database libraries and make them available to *SunAS*. You can do this by copying the libraries to the directory *./Sun/AppServer/lib*.

- **openMDX.** Copy *openmdx-kernel.jar* to the directory *./Sun/AppServer/lib*.
- **Database.** If you are using *PostgreSQL* as database copy *postgresql-8.1-407.jdbc3.jar* to *./Sun/AppServer/lib*. For other databases copy the corresponding driver library file(s) to this directory.

5 Configuring the Java Virtual Machine

Now you must add options to the *JVM* configuration. Navigate to *Application Server > JVM Settings Servers > JVM Options* as shown below:

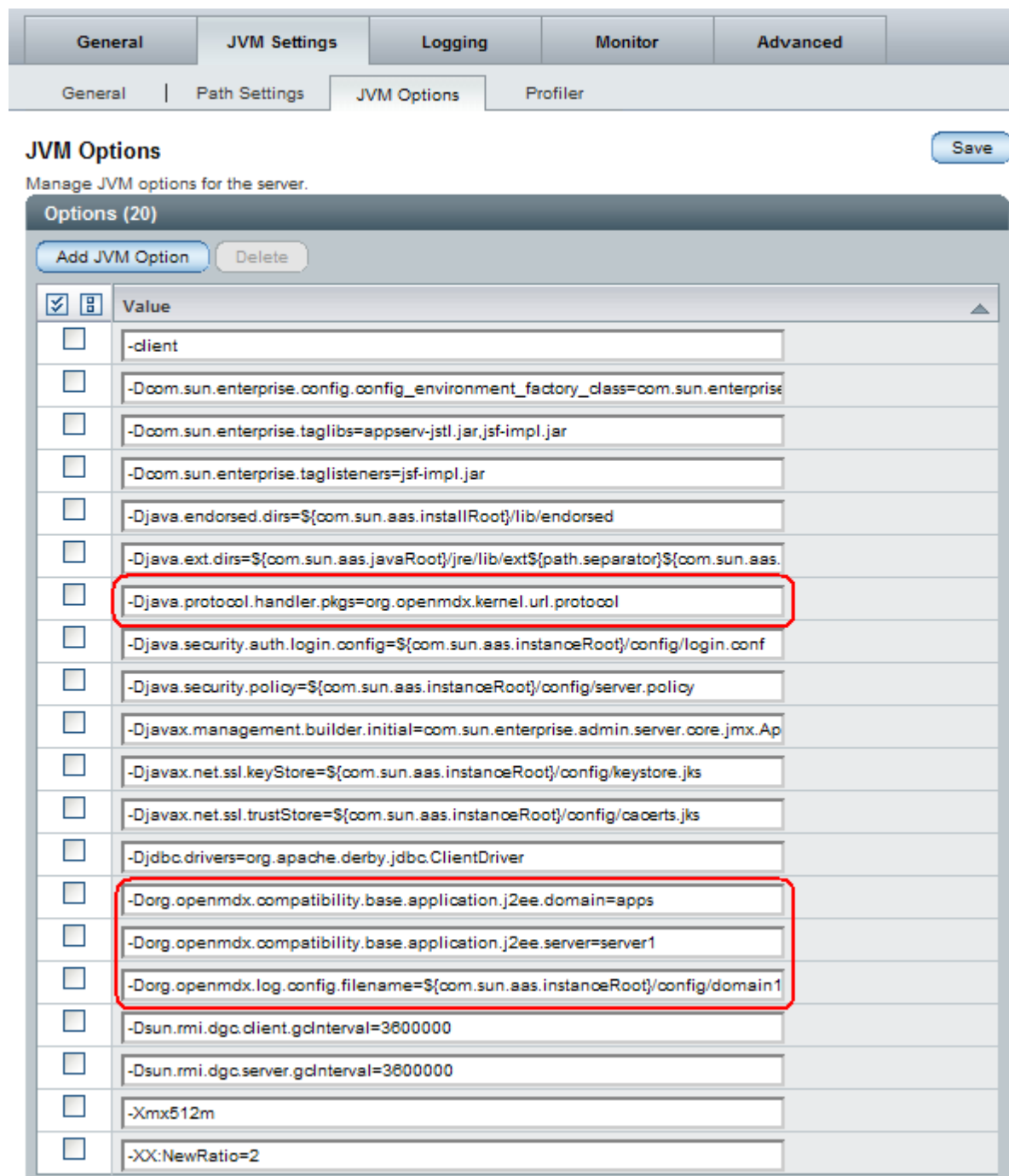


Figure 2: Configure the Java Virtual Machine.

Add the following options:

- `-Dorg.openmdx.compatibility.base.application.j2ee.domain=apps`
- `-Dorg.openmdx.compatibility.base.application.j2ee.server=server1`
- `-Djava.protocol.handler.pkgs=org.openmdx.kernel.url.protocol`
- `-Dorg.openmdx.log.config.filename=${com.sun.aas.instanceRoot}/config/domain1.log.properties`

Create the file `${com.sun.aas.instanceRoot}/config/domain1.log.properties` using a text editor with the following content.

Listing 1: Listing of domain1.log.properties.

```
# Log properties
ApplicationId      = SunAS-domain1
LogLevel          = warning
LogFileExtension  = log
LogFilePath       = C:/pgm/Sun/AppServer/domains/domain1/logs
java.LoggingMechanism = SharedDatedFileLoggingMechanism
```



Adapt `C:/pgm/Sun/AppServer/domains/domain1/logs` to your environment!



Before you continue you must restart (stop and start) *SunAS*. The newly configured libraries and environment variables only become active after restarting *SunAS*.

6 Configuring server.policy

In the file `${com.sun.aas.instanceRoot}/config/server.policy` modify the line shown in *Listing 2* to the line shown in *Listing 3*.

Listing 2: File permissions in server.policy before modification.

```
permission java.io.FilePermission    "<<ALL FILES>>", "read,write";
```

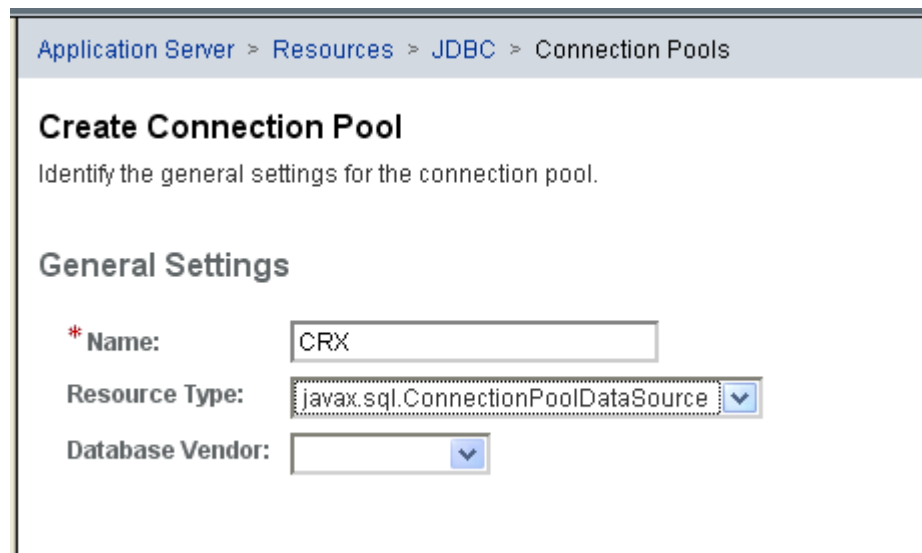
Listing 3: File permissions in server.policy before modification.

```
permission java.io.FilePermission    "<<ALL FILES>>", "read,write,delete";
```

7 Configuring the Datasource

openCRX requires the configuration of a *JDBC* datasource to connect to the *openCRX* database. You can do this as follows:

Navigate to *Resources > JDBC > Connection Pools* and create a new *Connection Pool* as shown below:



Application Server > Resources > JDBC > Connection Pools

Create Connection Pool

Identify the general settings for the connection pool.

General Settings

* Name:

Resource Type:

Database Vendor:

Figure 3: Create a Connection Pool.

For *PostgreSQL* fill out the fields as follows:

- **Name:** *CRX*
- **Resource Type:** *javax.sql.ConnectionPoolDataSource*
- **Database Vendor:** -

On the next page you must enter the

- **Datasource Classname.** For *PostgreSQL* set the value to *org.postgresql.jdbc3.Jdbc3PoolingDataSource* as shown below:

Application Server > Resources > JDBC > Connection Pools

Create Connection Pool

Review the connection pool general settings before proceeding.

General Settings

Name:	CRX
Resource Type:	javax.sql.ConnectionPoolDataSource
Database Vendor:	
* Datasource Classname:	<input type="text" value="org.postgresql.jdbc3.Jdbc3PoolingDataSource"/>

Vendor-specific classname that implements the `DataSource` and/or `XDataSource` APIs

Figure 4: Enter the data source class name.

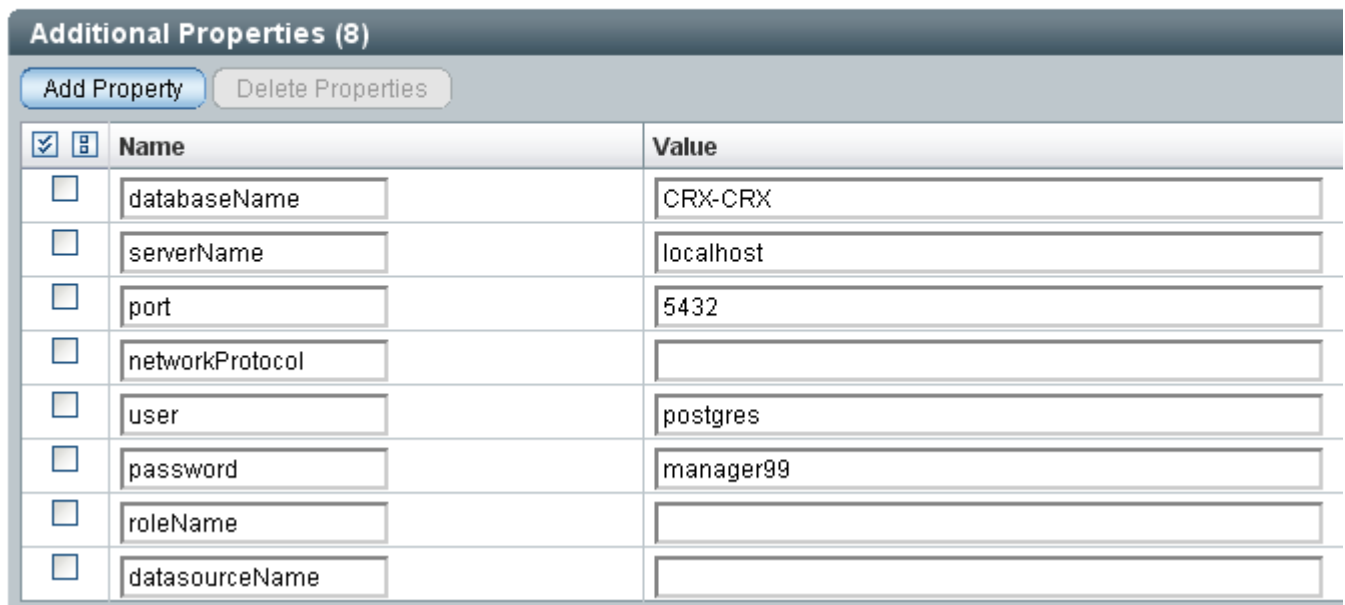
Additional Datasource Classnames:

- MS SQL 2005:
com.microsoft.sqlserver.jdbc.SQLServerConnectionPoolDataSource

Click *Next* and enter the database-specific properties as shown in Figure 5. For *PostgreSQL* fill out the fields as follows:

- **databaseName:** *CRX-CRX*
- **serverName:** *localhost*
- **port:** 5432
- **user:** *postgres* (user which is allowed to access the database *CRX-CRX*).
- **password:** password of the specified database user.

Properties



The screenshot shows a dialog box titled "Additional Properties (8)". At the top, there are two buttons: "Add Property" and "Delete Properties". Below the buttons is a table with two columns: "Name" and "Value". Each row in the table has a checkbox on the left side of the "Name" column. The table contains the following entries:

<input checked="" type="checkbox"/>	Name	Value
<input type="checkbox"/>	databaseName	CRX-CRX
<input type="checkbox"/>	serverName	localhost
<input type="checkbox"/>	port	5432
<input type="checkbox"/>	networkProtocol	
<input type="checkbox"/>	user	postgres
<input type="checkbox"/>	password	manager99
<input type="checkbox"/>	roleName	
<input type="checkbox"/>	datasourceName	

Figure 5: Specify the database-specific properties.

Next you must create a *JDBC* resource. Select *Resources > JDBC > JDBC Resources* and then click *New*. Enter the values as shown below:

- **JNDI Name.** *jdbc/opencrx_CRX*.
- **Pool Name:** Select the previously created connection pool *CRX* in the drop-down.

Application Server > Resources > JDBC > JDBC Resources

Create JDBC Resource

When you create a JDBC resource, you specify a unique JNDI name that identifies the resource

* Indicates required field

* **JNDI Name:**

* **Pool Name:**
Use the [Connection Pools](#) page to create new pools

Description:

Status: Enabled

Figure 6: Create a JDBC resource.

The datasource configuration is now complete.

8 Configuring Security

openCRX requires that each user is properly authenticated. This allows *openCRX* to correlate a session to user-specific application data and to perform access control. *openCRX* does not support non-authenticated sessions. Go to *Configuration > Security > Realms > file > Manage Users* as shown below:

Application Server > Configuration > Security > Realms > file

Edit Realm

Edit an existing security realm. Use the Manage Users button to create or modify user accounts for file realm users.

Realm: file
Select a repository where the server stores user and group information

*** Class Name:**
Specify the class name for the realm you want to create

Additional Properties (2)

<input checked="" type="checkbox"/> <input type="checkbox"/>	Name	Value
<input type="checkbox"/>	<input type="text" value="file"/>	<input type="text" value="\${com.sun.aas.instanceRoot}/config/keyfile"/>
<input type="checkbox"/>	<input type="text" value="jaas-context"/>	<input type="text" value="fileRealm"/>

Figure 7: Manage users.

In the *Manage Users* screen click *New* to add the user *guest* as shown below:

Application Server > Configuration > Security > Realms > file

Create File Realm User

Creates new user accounts for the currently selected security realm.

* Indicates required field

* **User ID:**
Name of a user to be granted access to this realm

* **Password:**

* **Confirm Password:**

Group List:
Separate multiple groups with commas

Figure 8: Add user guest.

Make the user member of the group *OpenCrxUser*. *openCRX* defines the following security roles:

- *OpenCrxUser* and *OpenCrxAdministrator*. Users who are member of *OpenCrxUser* or *OpenCrxAdministrator* are able to login to the web application.
- *OpenCrxRoot*. Root users who are member of *OpenCrxRoot* are able to login to the web application.



Be careful that you add to the group *OpenCrxRoot* only users who should have *openCRX* root privileges. The *openCRX* standard configuration assumes that you add the user *admin-Root* to the group *OpenCrxRoot*, the user *admin-Standard* to the group *OpenCrxAdministrator*, and the user *guest* to the group *OpenCrxUser*.

The roles are defined in the deployment descriptors of the application enterprise archives. In a next step add all the users who should have access to *openCRX*. At least add the following users:

- **guest**: group list = {*OpenCrxUser*}
- **admin-Standard**: group list = {*OpenCrxAdministrator*}
- **admin-Root**: group list = {*OpenCrxRoot*}

Finally, the users list should look similar as shown below:

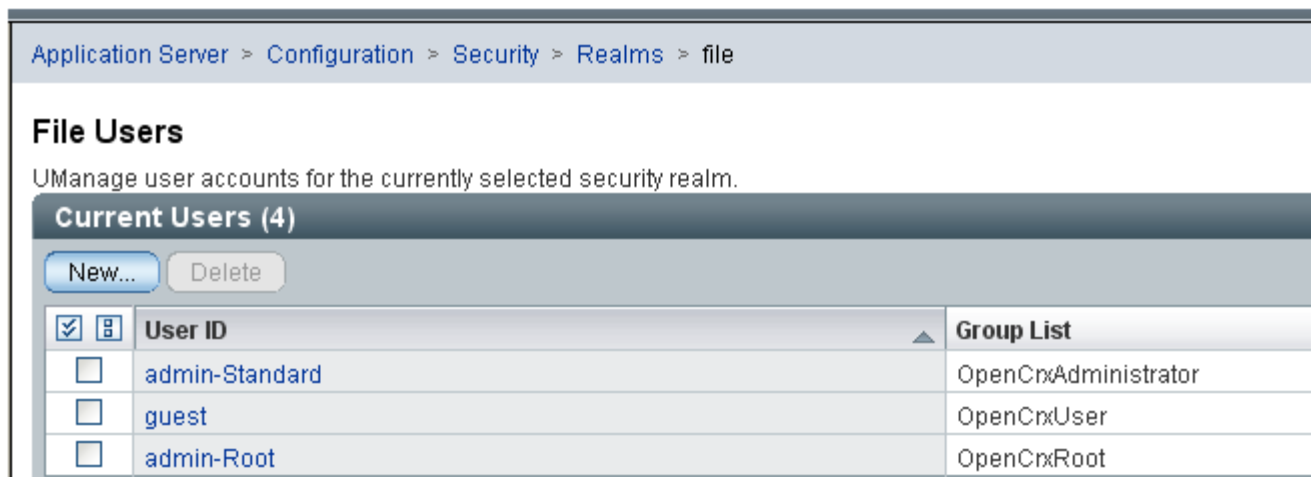


Figure 9: List of users who have access to *openCRX*.

9 Deploying openCRX

openCRX comes with two enterprise application archives (EAR):

- *opencrx-core-CRX-App.ear*. Contains the *openCRX* server components, i.e. Enterprise Java Beans.
- *opencrx-core-CRX-Web.ear*. Contains the web application for *openCRX* users.

In a first step you deploy *opencrx-core-CRX-App.ear*. Select *Enterprise Applications* and then click the *Deploy* button to select the *EAR* file as shown below:

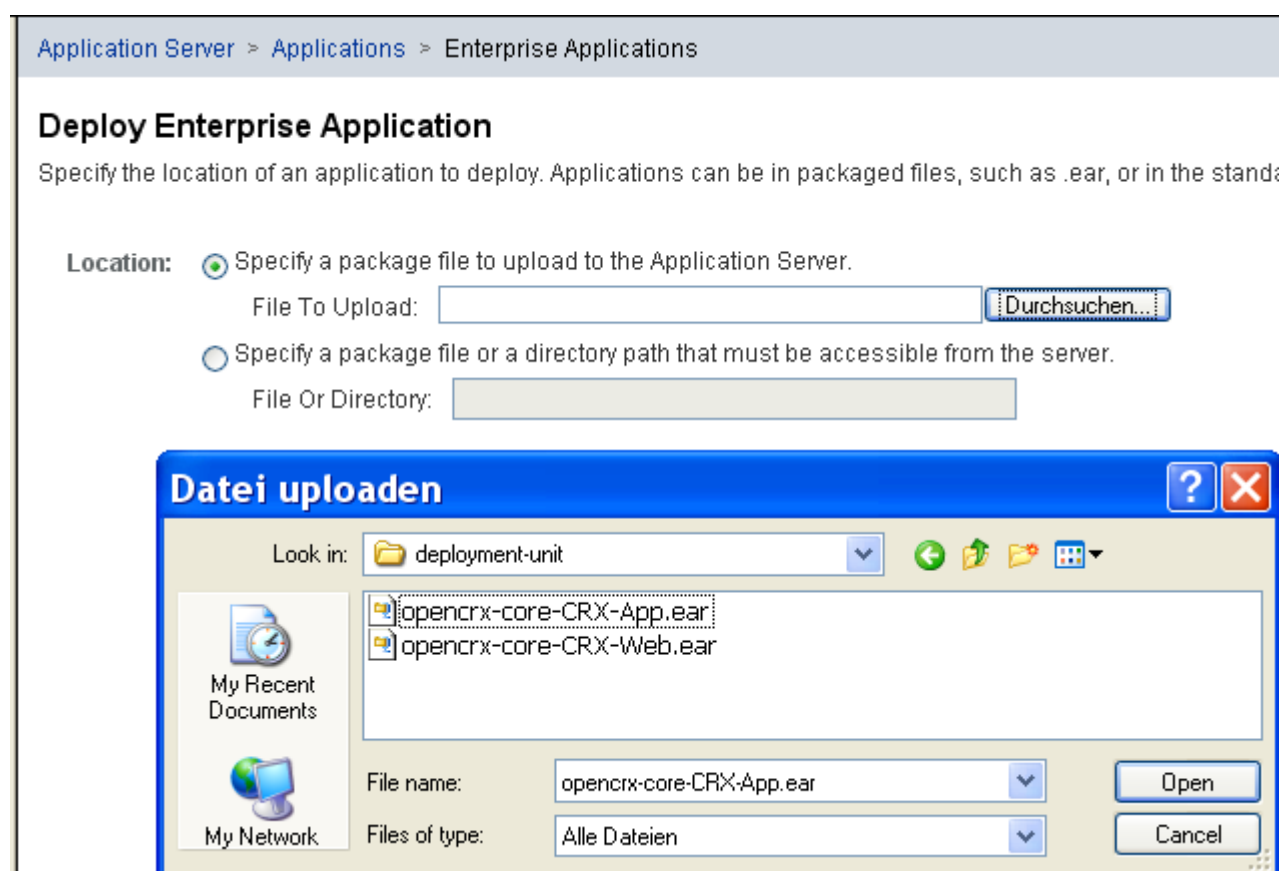


Figure 10: Deploy the *opencrx-core-CRX-App* EAR.

Click *Next* and accept the default values in the next screen as shown below:

Application Server > Applications > Enterprise Applications

Deploy Enterprise Application

General

File Name: opencrx-core-CRX-App.ear

*** Application Name:**

Virtual Servers:
Associates an internet domain name with a physical server

Status: Enabled

If Exists: Redeploy
Redeploy if application with same name exists in deployed list

Run: Verifier
Perform detailed verification before deploying

Precompile: JSPs
Precompile JSPs; deploy only resulting class files

Description:
Add a description of the component

Advanced

Generate: RMIStubs
Generate static RMI stubs and put in client.jar

Figure 11: Accept the default values in deploy enterprise application.

Click *OK* in order to deploy the application.



The deployment process must pass without any errors and warnings. If the deployment process fails *openCRX* will not start.

Next you must install the web application *opencrx-core-CRX-Web.ear*. The process is the same as described previously for the *opencrx-core-CRX-App.ear*. However, in the *Deploy Enterprise Application* screen select the option *Precompile JSP* as shown below. The web application contains Java Server Pages which must be precompiled during deployment.

Application Server > Applications > Enterprise Applications

Deploy Enterprise Application

General

File Name: opencrx-core-CRX-Web.ear

*** Application Name:**

Virtual Servers:
Associates an internet domain name with a physical server

Status: Enabled

If Exists: Redeploy
Redeploy if application with same name exists in deployed list

Run: Verifier
Perform detailed verification before deploying

Precompile: JSPs
Precompile JSPs; deploy only resulting class files

Description:
Add a description of the component

Advanced

Generate: RMISubs
Generate static RMI stubs and put in client.jar

Figure 12: Deploy the openCRX web application. Select Precompile JSP.



The deployment process must pass without any errors and warnings. If the deployment process fails *openCRX* will not start.

10 Final Steps

Before you proceed to the *openCRX QuickStart guide* make sure that you have deployed and enabled the applications as shown below:



Figure 13: The openCRX applications must be deployed and enabled.

The application is initialized the first time a user calls the login page. If the startup fails you should consult the following log files:

- **SunAS logs.** Located in *Sun\AppServer\domains\domain1\logs*.
- **openCRX SunAS-domain1.shared...log.** Located in *Sun\AppServer\domains\domain1\logs* and contains the *openCRX* application log files.