Banner

Employee Self-Service

Installation Guide

Release 9.7

July 2018

Notices

Ellucian
2003 Edmund Halley Drive
Reston, VA 20191
United States of America

Contents

Before you get started

Review these prerequisites for hardware, software, and the supporting documentation you can reference.

Use Ellucian Solution Manager to install your product upgrades

Ellucian recommends that you use Ellucian Solution Manager to perform Banner product upgrades, rather than using a manual installation process.

With Solution Manager, you can:

- Identify dependency information within and between products.
- View the latest version numbers for the Banner products you have installed, along with all other version numbers installed in your environment.
- Use the **Get New Releases** feature to identify available upgrades and download them immediately.
- Identify and install product pre-requisites, along with any upgrades you have selected.

Solution Manager currently supports most Banner 8 and 9 products. For more information on the Banner product versions currently supported by Solution Manager, see the *Banner Upgrades Support Status Guide*. For detailed instructions on how to install and configure Solution Manager, see the latest *Solution Manager User Guide*.

Supporting documentation

You can obtain documentation for any application you download through Ellucian Solution Manager by going to **Products page** > **Latest Available Releases version number** > **documentation icon**.

- *Ellucian Solution Manager Installation and Configuration Guide* (latest version) – found in the Ellucian Solution Manager documentation library in the Ellucian Support Center.
- *Banner Middle Tier Implementation Guide* – found in the Software Downloads section of the Ellucian Support Center. Only users with software download privileges have access to this publication.
- *Application Navigator Installation Guide*
- *Banner 9 Sizing and Configuration Guide*

Hardware requirements

These are the hardware requirements Ellucian recommends.

CPU and memory

Recommended: Quad core CPU with 4 to 8 GB of memory for the application server.

Minimum: Dual core CPU with 6 GB of memory for the application server.

Screen resolution

Minimum: 1024 x768

Tablets

Ellucian supports the following minimum tablet versions. You can implement on one of these versions or higher.

- iPad, iPad Mini, iOS 6x, and iOS 7.x
- Android OS 4.c
- Microsoft Surface 1.0, RT, and PRO

Software requirements

These are the software requirements Ellucian recommends.

Oracle database

Supported versions of the Oracle Database depend on multiple factors, including third-party support time lines. For a complete list of supported Oracle technologies, refer to the Ellucian Oracle Support Calendar. The calendar is available in the Interactive Banner Compatibility Guide, which can be accessed from the Ellucian Download Center.

Web application servers

The Banner Employee Self-Service application is supported by WebLogic and Tomcat web servers.

- Oracle WebLogic: 10.3.3, 10.3.4, 10.3.5, 10.3.6, 12.1, and 12.1.3
- Apache Tomcat: 7 and 8

Oracle Fusion Middleware (OFM) consists of several software products, including WebLogic Server. WebLogic Server is required for an Oracle Banner 9.x application server environment. No other OFM products are required. However, if an SSL-enabled Oracle HTTP Server (OHS) port is used, the Oracle Web Tier should also be installed to use the `mod_wl_ohs`.

Middle tier (application server) platforms

The Employee Self-Service application supports multiple operating systems on both Tomcat and WebLogic servers.

Tomcat (64 bit)	WebLogic (64 bit)
Red Hat Linux 6.0 (minimum version)	Red Hat Linux 6.0 (minimum version)
Windows Server 2008	Windows Server 2008
Solaris 10	Solaris 10
AIX 6.1 (JDK 1.7 SR 10 or later)	AIX 6.1 (JDK 1.7 SR 10 or later)
HP UX	HP UX: 11iV3 (11.31)

Note: Banner 9.x applications are tested on WebLogic using both the Classic Domain template and the Basic Domain template. For WebLogic server environments, JPA 2.0 support must be enabled. WebLogic server does not enable JPA by default.

Developer requirements

Users who pull the source code from GIT need to use specific versions of software when making extensions.

- Grails 2.5.0
- JDK 1.7
- Tomcat 8
- Oracle 11.2.0.4 or higher
- Release GIT Tag, `rel-employeeselfserv-9.7`

Single sign-on (SSO) support

Ellucian recommends that you first establish the single sign-on environment before implementing Employee Self-Service.

Central Authentication Service (CAS) and Security Assertion Markup Language (SAML) 2.0 are the Single Sign-On (SSO) protocols supported by Employee Self-Service. Both are certified to run with Ellucian Ethos Identity. You can download Ellucian Ethos Identity from the Ellucian Support Center.

For more information about how to establish a single sign-on environment, refer to the *Setting Up Ellucian Ethos Identity* guide. Both are available for download from the Documentation Libraries in the Ellucian Support Center.

Warning! If you do not enable Single Sign-On (SSO), a user must authenticate before accessing each Banner 9.x application. If a user logs out of an application, the user is logged out of that current application, but is still logged in to all other applications that are currently open.

Java dependencies

Development for Banner 9.x is currently supported on Java 7 only, and Java 1.7.x (64-bit version) must be installed on the application server before you install the application..

The same version of Java must be used to customize and deploy the WAR file. The JDK bin directory must be defined in the PATH system property.

Note: Java 7 includes security restrictions for Rich Internet Applications. Refer to Article 000030656 on the Ellucian Support Center for details on Java 7 security restrictions with Liveconnect calls to Oracle Forms Applet.

Browsers

For more information about supported browsers, refer to the Interactive Banner Compatibility Guide on the Ellucian Download Center.

- Internet Explorer 11
- Google Chrome
- Mozilla Firefox
- Safari 6 or 7
- Microsoft Edge

Internet Explorer Compatibility view

When using Internet Explorer, you must be in Internet Explorer Standard Mode.

Procedure

1. **Tools** > **Compatibility View Settings**
2. Clear the **Display intranet sites the Compatibility View** check box, and select the **Display all websites** check box.

F5 load balancer configuration

The application was tested using an F5 load balancer.

It was configured with the following settings:

```
Load Balancing type = Round Robin
```

```
Persistence = Cookie
```

Note: Other configurations may be supported depending on Network Load Balancing (NLB).

Configure Application Navigator

Application Navigator is a software component that facilitates seamless navigation between Banner 8.x and Banner 9.x Administrative applications, in addition to Banner 9.x Self-Service applications.

You can configure Application Navigator so that a menu item will display in the Application Navigator menu for each Self-Service Application. Refer to the *Application Navigator Installation Guide* and *Application Navigator Handbook* for instructions on setting up this configuration. These guides are available from the Ellucian Support Center in the Banner General Documentation Library.

Upgrade the database

Employee Self-Service requires a minimum of Banner Database Upgrade 9.15.

For more information about that, see the *Banner Database Upgrade 9.15 Upgrade Guide* , which you can download from the Ellucian Support Center in the Banner General Documentation Library.

Update login.sql

You must edit `login.sql` to update the schema owner's default password and to specify the path to create log files.

Procedure

1. Replace the `#UPDATEME#` string with the value of a particular schema owner's password in your environment. Make this update in your environment for each Banner schema owner.
2. Set the value that gets assigned to `splpref`.

   The value can be set to the ORACLE_SID or to a directory name. Your options depend on the operating system.

   The `splpref` variable defines the file prefix that the installation process uses to generate listings or intermediate SQL routines. This feature allows you to segregate the generated output when the stage must be applied to more than one instance.

Verify that the required products are applied

You have to check and verify that all prerequisite products are applied to the environment.

Procedure

1. Invoke SQL*Plus and run the following procedure:

   ```
   sqlplus /nolog @ruappready
   ```
2. Review the `ruappready` listing.

Verify the banproxy database account

The banproxy account is used for database connections for administrative applications. The database upgrade process grants the BAN_DEFAULT_M role to banproxy. If this role is revoked, the application will not start successfully.

Verify the ban_ss_user database account

The ban_ss_user account is used for database connections for self-service applications. The database upgrade process grants the USR_SS_DEFAULT_M role to ban_ss_user. If this role is revoked, the application will not start successfully.

Verify Oracle user accounts to connect through banproxy

All Internet Native Banner (INB) or Oracle user accounts must connect using the banproxy privilege. Verify that Oracle user accounts can connect through banproxy.

Procedure

1. Access the Security Maintenance (GSASECR) page.
2. Enter a valid user name.
3. Click **Alter**.
4. Select the **Authorize banproxy** check box.
5. Click **Save**.

Set up access for application users with an administrative account

A new security object named SELFSERVICE is created during the installation of the selfservice application. Application users who have an administrative account associated with their login on the Enterprise Access Controls (GOAEACC) page must be assigned this new object with BAN_DEFAULT_M privilege.

Note: The SELFSERVICE object was also added to the BAN_GENERAL_C class. As an alternative, you can associate your administrative users with this class.

Migrate staged files to the permanent directories

This release provides migration scripts for Unix and Windows platforms. These scripts expect your directory structure to match the directory structure created by the Banner installation process. If you choose a different directory structure, you must modify the scripts.

The release does not include migration scripts for other platforms due to their highly customized structures. You can, however, use the file `BWPMIGR.TXT` as a starting point for writing your own migration scripts.

Unix

The file `BWPMIGR.TXT` lists all files that must be deleted from your permanent directories, and all files that should be copied from the staging directory to your permanent directories. The destination is indicated in UNIX format. The format is different on other platforms.

About this task

The file `bwpmigr.shl` does the appropriate removes, copies, and links. The local LN variable at the top of `bwpmigr.shl` determines the type of links that are used in the migration. If you want to use symbolic links, set `LN='ln -s'` so that the command `${LN} file $BANNER_HOME/links` is translated to `ln -s file $BANNER_HOME/links`. If you want to force the removal of any existing targets before linking files, set `LN='ln -f'`.

To run the migration script in background on a Unix platform, perform the following steps:

Procedure

1. Ensure that the directory path names in `bwpmigr.shl` are correct.
2. Ensure that the environment variable *$BANNER_HOME in bwpmigr.shl* is set to the appropriate directory.
3. Sign on to an operating system account that has write permission into the target Banner directories.
4. If you are a cshell user (your operating system prompt is a percent sign), enter `sh` and press Enter to enter the Bourne shell.
5. Navigate to the staging directory for the product.
6. Run the migration script as follows:

   ```
   sh bwpmigr.shl >bwpmigr.log 2>&1 &
   ```

7. If you were a cshell user and want to return to that mode, press CTRL-D or enter `exit`. Then press Enter.
8. Review `bwpmigr.log`. This file contains the results of the migration.

Note: Even if your directory structure matches the baseline perfectly, some link commands will fail (that is, where the link currently exists). Other link errors might indicate that you had two copies of an object when the migration script was executed. This condition must be corrected. The duplication is probably between links and the product subdirectory.

Windows

The file `bwpmigr.pl` does the appropriate deletes and copies. To run the migration script on a Windows platform, perform the following steps.

Procedure

1. Check the value of the `BANENV` environment variable by executing the SET command from the DOS prompt.
   a) If the value of `BANENV` is `REG`, the value used for `BANNER_HOME` will be taken from the registry entry: `HKEY_LOCAL_MACHINE\SOFTWARE\BANNER\BANNER_HOME`.
   b) If the value of `BANENV` is `ENV`, the value used for `BANNER_HOME` will be taken from the environment variable `BANNER_HOME`.
2. Ensure that the directory path names in `bwpmigr.pl` are correct.
3. Sign on to an operating system account that has write permission into the target Banner directories.
4. Navigate to the staging directory for the product.
5. Run the migration script as follows:
   ```
   perl bwpmigr.pl >bwpmigr.log 2>&1
   ```
6. Review `bwpmigr.log`.
   This file contains the results of the migration.

Update the version numbers

To insert the release version numbers into the Web Application (GURWAPP) table, perform the following steps.

Procedure

1. Invoke SQL*Plus and run the following procedure to insert the version number for the self-service application:
   ```
   sqlplus general/password

   start versionupdate
   ```
2. Review the `versionupdate` listing.

Restore original roles for accounts modified

In this part, you will run the `essresroled.sql` to restore the original roles for those accounts modified by the ruappready script run at the beginning of this upgrade.

About this task

Warning! If you are running upgrades in parallel, do not run this part until you have finished all of your upgrades. The reason is that several accounts are used by all upgrades. If they do not have the DBA role as a default role, this script will reset their privileges so DBA is not included. This could have adverse effects on the other upgrades.

Procedure

1. Invoke SQL*Plus and run the following procedure:
   ```
   sqlplus /nolog @essresroled
   ```
2. Review the essresroled listing.

Undeploy existing application

Before you install Banner Employee Self-Service 9.7, you must undeploy the previously deployed Banner Employee Self-Service 9.x application. If you have been using the Employee Profile and Position Description applications, then you must also undeploy Employee Profile 9.x and Position Description 9.x.

The procedure for undeploying the applications varies depending on whether you are using Tomcat or WebLogic servers.

Undeploy applications by using the Tomcat Manager web application

This task allows you to undeploy any previously deployed versions of Banner Employee Self-Service 9.x, Banner Employee Profile 9.x, and Banner Position Description 9.x applications on your systems from the Tomcat server by using the Tomcat Manager web application.

Procedure

1. Access the Tomcat Manager web application at one of the following URLs: `http://server:8080/manager` or `http://server:8080/manager/html`.
2. Access the deployment page by using a valid user name and password.
3. If you have been using Employee Self-Service 9.x, Employee Profile 9.x, or Position Description 9.x, then stop previously deployed versions of these applications.
   a) In the **Commands** area, click **Stop** to stop the existing application.
4. In the confirmation dialog box, click **OK**.
5. In the **Commands** area, click **Undeploy**.
6. In the confirmation dialog box, click **OK** to undeploy the application.

Undeploy applications manually on UNIX

This task allows you to shut down Tomcat and manually undeploy any previously deployed versions of Banner Employee Self-Service 9.x, Banner Employee Profile 9.x, and Banner Position Description 9.x on your UNIX systems.

Procedure

1. Log in to the server on which Tomcat is running by using the credentials that were used to start Tomcat.
2. Shut down Tomcat by running the following shutdown script: `$CATALINA_HOME/bin/shutdown.sh`.

3. Remove the current deployment:
   a) If you have been using Employee Self-Service 9.x, Employee Profile 9.x, or Position Description 9.x, then remove the previously deployed versions of these applications.

```
cd $CATALINA_HOME
rm -rf $CATALINA_HOME/webapps/<Employee Self Service 9x Application>
rm -rf $CATALINA_HOME/webapps/<Employee Profile 9x Application>
rm -rf $CATALINA_HOME/webapps/<Position Description 9x Application>
```

4. Remove the associated WAR file for each of the applications removed in the previous step: `rm -rf $CATALINA_HOME/webapps/`*`war_file_name`*.

Undeploy applications manually on Windows

This task allows you to shut down Tomcat and manually undeploy any previously deployed versions of Banner Employee Self-Service 9.x, Banner Employee Profile 9.x, and Banner Position Description 9.x applications on your Windows systems.

Procedure

1. Shut down Tomcat.

If you have	Then
Installed Tomcat as a service	Use the Service Control panel to stop the application.
Not installed Tomcat as a service	Use the following shutdown script: `%CATALINA_HOME%\bin\shutdown.bat`

2. If you have been using Banner Employee Self-Service 9.x, Banner Employee Profile 9.x, and Banner Position Description 9.x applications, then stop previously deployed versions of these applications.
3. Remove the previously deployed versions of the applications that you have stopped in the previous step.

```
rmdir %CATALINA_HOME%\webapps\<Employee Self Service 9x Application>
rmdir %CATALINA_HOME%\webapps\<Employee Profile 9x Application>
rmdir %CATALINA_HOME%\webapps\<Position Description 9x Application>
```

4. Remove the associated WAR file for each of the applications removed in the previous step: `del %CATALINA_HOME%\webapps\`*`war_file_name`*.

Undeploy applications by using WebLogic

This task allows you to undeploy any previously deployed versions of Banner Employee Self-Service 9.x, Banner Employee Profile 9.x, and Banner Position Description 9.x on your systems by using WebLogic.

Procedure

1. Access the administration server by using the following URL: `http://server:7001/console`.
2. In the **Domain Structure** frame, click **Deployments**.
3. In the **Change Center**, click **Lock and Edit**.
4. If you have used Banner Employee Self-Service 9.x, Banner Employee Profile 9.x, and Banner Position Description 9.x, then select the check box for the versions of the applications that you want to undeploy.
5. Click **Stop**.
6. Click **Force Stop Now**.
7. In the **Force Stop Application Assistant** page, click **Yes**.
8. Select the check box of the application that you have stopped in the previous step.
9. Click **Delete**.
10. In the **Delete Application Assistant** page, click **Yes**.
11. In the **Change Center** frame, click **Activate Changes**.

Customize the WAR file

The release package must be unzipped and the WAR file must be customized for your institution.

The `release-EmployeeSelfService-9.7.zip` release package must be moved to `$BANNER_HOME/payweb/java` subdirectory during the database upgrade.

Note: JDK 1.7 must be installed on your system.

Related Links

Java dependencies on page 10

Unzip the release package

To unzip the release package into a temporary directory, perform the following steps.

Procedure

1. Log in to the application server platform.

   Note: You must have a valid application server account to deploy into the application server container (Tomcat or WebLogic).

2. Create a temporary directory. For example:
   `mkdir $HOME/ban9temp`
3. Locate the release package `release-EmployeeSelfService-9.7.zip`.
4. Transfer this file in binary mode using File Transfer Protocol (FTP) file into the temporary directory. For example:
   `$HOME/ban9temp`
5. Unzip `release-EmployeeSelfService-9.7.zip` into the temporary directory.

Prepare the installer

To prepare the installer, perform the following steps.

Procedure

1. Change the directory to the installer directory:
   `cd installer`
2. Run the `ant` command, which will build the installation tool.

   Note: For Unix, make sure the ant file is executable. For example, `chmod +x ant`.

Example:

```
ban9temp $ cd installer
ban9temp/installer $ ./ant
```

The message `Build successful` confirms a successful build.

Install into the product home directory

The product home directory supports the configuration and creation of a deployable WAR file. Although Banner 9.x web applications are modular and are installed independently, they share a common configuration. The package provides a common installer that creates consistent product home directory structures for all Banner 9.x applications.

About this task

Within a particular environment, you should place the product home directories for Banner 9.x applications in sibling directories. For example, the following directory structure includes four product home directories and a `shared_configuration` directory that support a common test environment.

```
TEST/
|--> Catalog/
|--> Schedule/
|--> StudentOverall/
|--> StudentRegistration/
|--> shared_configuration/
```

A product home directory is created for each deployment. For example, the home directory that is used for the application within a test environment is different than the Application for CAS SSO home directory that is used for the production environment. When you are supporting different environments for multiple home directories for the same solution, this structure provides the necessary configuration, release level, and custom modification flexibility.

The following directory tree illustrates the product home directory that is created for the test environment:

```
TEST/                                   (optional and recommended top-level directory for all homes)

|-->app-name                            (product home for 'app-name' in test environment)
    |--> current
        |--> instance/                  (instance-specific configuration that will not be overwritten)
            |--> config/
               |--> {app-name}_configuration.groovy (module-specific configuration for CAS, logging, etc.)
            |--> i18n                    (new or replacement message bundles that should be added the war)
            |--> css                     (new or replacement css files that should be added the war)
            |--> js                      (new or replacement javascript files that should be added the war)

        |--> lib
            |--> ojdbc6.jar           (the Oracle database driver that must be placed manually into the tomcat/lib directory)
            |--> logging.properties   (logging configuration that may be copied to the WEB-INF/classes directory that is
                                       very useful if the war file cannot be deployed successfully.)
        |--> i18n/                      (contains message bundles that may reflect changes not yet in 'baseline')
        |--> dist/                      (contains the war file, after it is creating using the 'systool')
        |--> installer/                 (contains the installer)
    |--> archived-releases/             (directory for previous releases)
        |-->

|--> shared_configuration/              (home for configuration files shared across modules within an environment)
    |--> banner_configuration.groovy    (a 'shared  configuration file containing datasource)
```

In addition to the application's product home directory, a separate shared_configuration home directory contains cross-application configuration for the test environment. This directory holds the banner_configuration.groovy file, which contains the shared JNDI datasource configuration.

To install the installer into the product home directory, perform the following steps:

Procedure

1. Ensure that the installer is prepared using `ant`.
2. Use the installer to install the release file into the product home directory.

   Note: Your current working directory must be in the installer directory (ban9temp/installer) before executing the following commands.

   On Unix:

   ```
   $ bin/install home
   ```

   On Windows

   ```
   > bin\install home
   ```

3. When prompted, enter the full path of the application home directory.

   The application will be installed within the `current` subdirectory within this home directory and the previous release will be archived.

   On Unix:

   ```
   []: Current_home_directory/banner_test_homes/EmployeeSelfService
   ```

   On Windows:

   ```
   []: c:\banner_test_homes\EmployeeSelfService
   ```

4. Enter the full path of the `shared_configuration` home directory.

   Banner 9.x applications that refer to this home directory share this configuration file.

   On Unix:

   ```
   []: Current_home_directory/banner_test_homes/shared_configuration
   ```

On Windows:

```
[]: c:\banner_test_homes\shared_configuration
```

Note: If an identified home directory or the shared_configuration home directory does not exist, the installer creates it. The name of a product home directory is not restricted. You can name it when prompted by the installer.

Configure shared settings

The `shared_configuration` home directory contains a cross-application configuration file called banner_configuration.groovy. You can change settings in this file.

JNDI datasource

You can optionally change the datasource name in the configuration file to point to the JNDI datasource that is configured in your application server.

For example, `jndiName = "jdbc/bannerSsbDataSource"` is the default configuration. You can change this to match the JNDI datasource name in your environment. If you change the jndiName, you must regenerate the WAR file, even if you are using an external configuration.

Link to Self-Service Banner 8.x

To display existing Banner Self-Service 8.x menus and breadcrumbs, you must edit the `banner8.SS.url` configuration with the URL to your existing Banner Self-Service 8.x application.

```
//replace with the URL pointing to a Self-Service Banner 8.x instance
banner8.SS.url = '<scheme>://<server hosting Self-Service Banner
8.x>:<port>/<context root>/'
```

For example:

```
banner8.SS.url = 'http://localhost:8002/ssb8x/'
```

Note: If the `banner8.SS.url` setting is not in the `banner_configuration.groovy` file, you must add it to the file before you continue with the installation.

To provide single sign on (SSO) to Banner 8.x, Jasig Central Authentication Service (CAS) (http://www.apereo.org/cas) is required as an external authentication provider. You must configure the following components to authenticate using CAS:

- Employee Self-Service application must be configured to authenticate using CAS.
- The SSO Manager must be deployed and configured to enable Self-Service Banner 8.x authentication using CAS. The SSO Manager is a component of Banner Enterprise Identity Services (BEIS).

To allow SSO linking from the Employee Self-Service application to Self-Service Banner 8.x, add the following URL in the `banner_configuration.groovy` file:

```
banner8.SS.url = 'http://beissmpl.university.com:7777/ssomanager/c/SSB?
pkg='
```

The `banner8.SS.url` setting references the SSO Manager URL `('http://beissmpl.university.com:7777/ssomanager/c/SSB')` and appends the `'?pkg='` suffix to support deep linking to a specific Self-Service Banner 8.x page.

The following is an example of a Banner 8.x SSO URL through the SSO Manager:

```
http://beissmpl.greatvalleyu.com:7777/ssomanager/c/SSB?
pgk=bwgkogad.P_SelectAtypView
```

Banner Self-Service now supports locale specific redirect to Banner 8x from Employee Self-Service. The following is an example of a Banner 8 SS Locale URL:

```
banner8.SS.locale.url =[
default : 'http://localhost:8002/DEFAULT/',
en : 'http://localhost:8002/EN/',
en_US : 'http://localhost:8002/enUS/',
en_AU : 'http://localhost:8002/enAU/',
en_GB : 'http://localhost:8002/enGB/',
en_IE : 'http://localhost:8002/enIE/',
en_IN : 'http://localhost:8002/enIN/',
fr : 'http://localhost:8002/FR/',
fr_CA : 'http://localhost:8002/frCA/',
pt : 'http://localhost:8002/PT/',
es : 'http://localhost:8002/ES/',
ar : 'http://localhost:8002/AR/',
]
```

The following conditions affect how Banner Employee Self-Service redirects to Banner 8.x.

- If a locale specific URL does not exist, then it will use the default URL defined in the `banner8.SS.locale.url` map.
- For SPRIDEN users, if you do not define the `banner8.SS.url` setting with a valid URL, the Employee Self-Service application does not display Banner 8.x menus and breadcrumbs.
- Without CAS and the SSO Manager, navigation to Banner 8.x is not seamless. A user must log in to Banner Self-Service 8.x using a valid user name and password. Navigation terminates at the Banner Self-Service 8.x main menu page.

Navigational MEP support from Employee Self-Service to Banner Self-Service 8.x

Employee Self-Service application supports the ability to call a MEP context-sensitive URL that maps to the appropriate Banner Self-Service 8.x URL. The URL configurations enable the end user SSO access from Employee Self-Service to Banner Self-Service 8.x applications along with preserving the MEP context for that user session.

If your institution uses MEP, you must update key configurations and URL contexts. The key for the string must be in the following format: `mep.banner8.SS.url` and the key must be

configured with the following map list: `mep.banner8.SS.url = ['GVU': 'http://<host_name>:<port_number>/<banner8>/SMPL_GVU', 'BANNER': 'http://<host_name>:<port_number>/<banner8>/SMPL_BANNER' ]`

The following is an example of a Banner 8.x SSO URL through the SSO Manager in a MEP environment: `mep.banner8.SS.url = ['GVU': 'http://e009070.ellucian.com:8888/ssomanager/c/SSB?pkg=http://e009070.ellucian.com:9020/SMPL_GVU/', 'BANNER': 'http://e009070.ellucian.com:8888/ssomanager/c/SSB?pkg=http://e009070.ellucian.com:9020/SMPL_BANNER/' ].`

Use the following structure to support a locale specific redirect to Banner 8:

```
mep.banner8.SS.locale.url=[
      GVU:
                [
                     “default” : 'sampleUrlDefault ',
                     "ar": 'sampleUrlAR ',
                     "fr": 'sampleUrlfr',
                     "fr_CA": 'sampleUrlfrca'
                ],
      BANNER :
                [
                      “default” : 'sampleUrlDefault',
                      "ar": 'sampleUrlAR',
                      "fr": 'sampleUrlfr ',
                      "fr_CA": 'sampleUrlfrca'
                ]
]
```

Note: If a locale specific URL does not exist, then it will use the default URL defined in the `mep.banner8.SS.locale.url` map.

Session timeouts

When you determine the timeout settings for CAS and Banner 9.x applications, consider the time limits imposed for each.

For example, if the Banner 9.x timeout setting is less than the CAS timeout setting, SSO authentication might still be valid even though the user is exited from the Banner 9.x application. As long as the CAS authentication remains valid, the user can re-enter the Banner 9.x application without re-authenticating.

The following timeouts are used in the self-service application.

banner.transactionTimeout

The `banner.transactionTimeout` setting is used to prevent excessive delays due to long database transactions. The setting is configured in the `banner_configuration.groovy` file. To use this timeout, set `the banner.transactionTimeout` setting to `300` seconds.

Ensure that either the configuration file is deployed with the application, or the application is using the configuration file where it is currently located.

If a database transaction takes longer than `banner.transactionTimeout` seconds, the transaction is aborted and any change is rolled back.

Note: In some cases, the web user interface might ignore the database timeout/error notification and not remove the loading spinner. If this occurs, refresh the page to continue using the application.

AJAX timeout

The AJAX timeout terminates HTTP requests that exceed the specified time limit. The self-service application sets the timeout value based on the `banner.transactionTimeout setting` plus an increment to allow for communication and processing of the request.

You do not need to override the AJAX timeout, but the timeout value can be overridden in JavaScript by calling `$.ajaxSetup( { timeout: timeoutValue } );`

Note: The `timeoutValue` must be in milliseconds.

Note: Although the web user interface continues after an AJAX timeout, the server might continue processing the request until it completes or reaches a database `transactionTimeout`.

defaultWebSessionTimeout

The `defaultWebSessionTimeout` property is used to terminate user sessions that are left idle or abandoned without logging out. This setting is used to set the overall session inactivity time. The `defaultWebSessionTimeout` setting can be configured in the `banner_configuration.groovy file`.

Note: If `defaultWebSessionTimeout` is not specified, a default value of `1500` seconds is used.

Role-based web session timeouts are configurable in Banner Web Tailor where `TWTVROLE` is the logged in user role and `TWTVROLE_TIME_OUT` is the timeout for users with that role.

Note: An individual's session timeout is the longer of the `defaultWebSessionTimeout` and the role-based timeout from `TWTVROLE`.

Location of photographs

The `banner.picturesPath` setting in the `banner_configuration.groovy` file specifies the location of student, faculty, and employee photographs.

In `banner { picturesPath = System.getProperty('base.dir') + '/test/images' }`, `banner.picturesPath` setting is the application base directory appended with `/test/images`.

The `base.dir` is a system property that can be set through `JAVA_OPTS`.

The value of `banner.picturesPath` must be a fully qualified path to the directory that contains the image files. For example, `banner { picturesPath = '/home/banner/test/images' }`.

Configure application-specific settings by modifying the groovy file

Banner XE applications read settings from two configuration files:
the general `banner_configuration.groovy` file and the `EmployeeSelfService_configuration.groovy` file. The EmployeeSelfService file provides application-specific settings and can override settings in the general configuration file.

About this task

Applications often override general settings for logging and keeping application logs in a file separate from other applications. Some applications also define custom settings. For example, the Banner Student Advisor application allows configuration of roles that may access specific views within the application.

Procedure

1. Copy `EmployeeSelfService_configuration.example` and change the name to `EmployeeSelfService_configuration.groovy`.

   The installer creates the `EmployeeSelfService_configuration.example` file.
2. Place the `EmployeeSelfService_configuration.groovy` file in the `EmployeeSelfService\current\instance\config` directory.

   This application-specific configuration file contains settings that you can customize for your specific environment.

JMX MBean name

The name that is used to register MBeans must be unique for each application that is deployed into the JVM. This configuration should be updated for each instance of each application to ensure uniqueness.

For example:

```
jmx {
exported {
log4j = "EmployeeSelfService-log4j"
} }
```

In the example, the user needs to update `EmployeeSelfService-log4j` identifier for each installation of each application. This allows the JMX management to distinguish between different installations of the application.

Location of the logging file

Log4j is the common logging framework used with applications that run on the Java Virtual Machine. You can configure the location at which the log file is saved.

For more information about the Log4j settings, see http://docs.grails.org/2.5.0/guide/conf.html.

The configuration file includes documentation on various elements that can be modified depending on your environment.

The following example shows how to configure the location where the log file is saved:

```
loggingFileDir = "System.properties['logFileDir'] ?
"${System.properties['logFileDir']}" : "target/logs"
logAppName = "EmployeeSelfService"
loggingFileName = "${loggingFileDir}/
${logAppName}.log".toString()
```

`logAppName = "EmployeeSelfService"` must be set by the user.

The following example shows how to override the log file directory properties:

```
export JAVA_OPTS = "-DlogFileDir=/PRODUCT_HOME /"
```

The output log file location is relative to the application server to which you are deploying.

Logging level

The root logging level is pre-configured to the `ERROR` level. Multiple class or package level configurations, by default, are set to a status of `off`. You can set a different logging level for any package or class. However ,configuration changes for logging take effect when the application is restarted.

For example:

```
case 'production':
root {
error 'appLog' //change the log level here with the
appropriate log level value.
additivity = true
}
```

Note: Changing the logging level to `DEBUG` or `INFO` produces very large log files.

Changes to the `EmployeeSelfService_configuration.groovy` file take effect after the application is restarted.

Alternatively, you can use JMX to modify logging levels for any specified package or class, or even at the root level. When using JMX, the logging level changes only affect the running application. When you restart the application, changes that you made using JMX are lost.

Related Links

Configure Java management extension on page 51

Institutional home page redirection support

The institutional home page redirection support configuration allows Employee Self-Service to provide an institutional home page to which users can navigate back to if they have access issues specific to insufficient privileges when accessing the Employee Self-Service application.

```
/*****************************************************************
* Home Page link when error happens during authentication. *
*****************************************************************/
grails.plugin.springsecurity.homePageUrl='http://URL:PORT/'
```

Proxied Oracle users

When connecting to self-service applications, the `ssbOracleUsersProxied` setting specifies whether the Oracle account is used for the database connections. The `ssbOracleUsersProxied` setting also controls whether Value Based Security (VBS) is enabled in the self-service application.

The following values can be used for the `ssbOracleUsersProxied` setting:

- `False` - The Oracle account is not used for the connection and VBS is not enabled in the self-service application. If the Oracle account is locked, the user can still log in to the self-service application.
- `True` - The Oracle account is used for the connection and VBS is enabled in the self-service application. If the Oracle account is locked, the user cannot log in to the self-service application.

Logout URL

You can specify where a user should be directed after logging out of the application by updating the `EmployeeSelfService_configuration.groovy` file.

There are two ways the application can handle logouts:

- Logouts can display the CAS logout page with a redirect URL.
- Logouts can automatically go to a redirect URL (without displaying the CAS logout page).

The redirect URL can be different for each Banner application, depending on where you choose to send the user. If the redirect URL is the same for all Banner applications, it can be defined in the global `banner_configuration.groovy` file.

Provide the CAS logout page with a redirect URL

With this method of handling logouts, users see the CAS logout page when they log out of the application. The CAS logout page displays a URL that users must click to continue.

Procedure

1. Use the `logout.afterLogoutUrl` setting to configure the logout URL.
2. Define the `logout.afterLogoutUrl` as follows: `logout.afterLogoutUrl='https://<CAS_HOST>:<PORT>/<cas>/ logout?url=http://myportal/main_page.html'`

Example

The logout URL in the following example instructs the CAS server to redirect the browser to `http://myportal/main_page.html` after performing a CAS single logout. Depending on your needs, you can customize the serverUrlPrefix, serviceUrl, and serverName entries.

```
// +++ CAS CONFIGURATION +++
*set active = true if the application is configured with
 CAS SSO
************************************************************
banner {
sso {
authenticationProvider = 'cas' // Valid values are:
 'default', 'cas',
'saml'
authenticationAssertionAttribute = 'UDC_IDENTIFIER'
if(authenticationProvider != 'default') {
```

```
grails.plugin.springsecurity.failureHandler.defaultFailureUrl
 = '/
login/error'
}
}
}
grails {
plugin {
springsecurity {
cas {
active = true
serverUrlPrefix = 'http://CAS_HOST:PORT/cas'
serviceUrl = 'http://BANNER9_HOST:PORT/APP_NAME/
j_spring_cas_security_check'
serverName = 'http://BANNER9_HOST:PORT'
proxyCallbackUrl = 'http://BANNER9_HOST:PORT/APP_NAME/
secure/
receptor'
loginUri = '/login'
sendRenew = false
proxyReceptorUrl = '/secure/receptor'
useSingleSignout = true
key = 'grails-spring-security-cas'
artifactParameter = 'SAMLart'
serviceParameter = 'TARGET'
serverUrlEncoding = 'UTF-8'
filterProcessesUrl = '/j_spring_cas_security_check'
}
logout {
afterLogoutUrl = 'https://cas-server/logout?url=http://
myportal/main_page.html'
mepErrorLogoutUrl = '/logout/logoutPage'
}
}
}
}
}
```

Provide only a redirect URL

With this method of handling logouts, users automatically go to a redirect URL.

Procedure

1. Configure logout information, replacing `url` with `service` as follows: `grails.plugin.springsecurity.logout.afterLogoutUrl = 'https://<CAS_HOST>:<PORT>/cas/logout?service=http://myportal/main_page.html'`.
2. Set the property `followServiceRedirects` to `true` on the LogoutController that is defined in `cas-servlet.xml` as follows: `<bean id="logoutController" class="org.jasig.cas.web.LogoutController" p:centralAuthenticationService-ref="centralAuthenticationService" p:logoutView="casLogoutView" p:warnCookieGenerator-`

```
ref="warnCookieGenerator" p:ticketGrantingTicketCookieGeneratorref="
ticketGrantingTicketCookieGenerator" p:followServiceRedirects="true" /
>.
```

Password reset

You can specify whether users have the ability to reset their passwords by updating the `ssbPassword.reset.enabled` setting.

If the value of the setting is `true`, users can reset their passwords. If the value of the setting is `false`, a `disabled` error message is displayed.

Redirect pages in a MEP environment

If your environment is enabled for Multi-Entity Processing (MEP), two settings (`grails.plugin.springsecurity.logout`) must be configured in the `EmployeeSelfService_configuration.groovy` configuration file for your environment. You do not need to configure these settings if your environment is not enabled for MEP.

If a user tries to access a self-service URL that does not include a valid MEP code, an error prompt is displayed on the **Error** page of the application and the user can use the **Logout** or **Return Home** button.

The following configuration settings determine where each button redirects the user:

- `grails.plugin.springsecurity.logout.afterLogoutUrl` - This setting contains the URL of the institution-specific page where the user is redirected if the **Logout** button is clicked.

  For example, a portal page has the following settings:

  - `https://cas-server/logout?url=http://myportal/ main_page.html` (CAS environment)
  - `http://myportal/main_page.html` (non-CAS environment)
- `grails.plugin.springsecurity.logout.mepErrorLogoutUrl` - This setting contains the URL of the institution-specific page where the user is redirected if the **Return Home** button is clicked.

Each URL can be any page that does not require a MEP-enabled database connection or any page outside the self-service application.

Theme Editor tool

Use the Theme Editor tool to apply color theme and logo changes to Banner applications.

Access the Banner Applications Configurations (GUACONF) page to set up, change, and enable the Theme Editor.

Setting	Description
`banner.theme.url="<UPDATEME>"`	The URL of the application hosting the Theme Server.
`banner.theme.name="<UPDATEME>"`	The name for the theme. **Note:** In a Multi-Entity Processing (MEP) environment, the application uses the MEP code as the theme name instead of `banner.theme.name`. You must create a theme with the same name in the Theme Editor on the server specified by `banner.theme.url`.
`banner.theme.template= "EmployeeSelfService"`	The scss file containing the theme settings in the WAR file.
`banner.theme.cacheTimeOut=120`	Indicates how long to cache the CSS file that was generated using the template and theme. This is an advanced setting that you can use when the application is configured to be the host of the theme server.

After you enable the Theme Editor, the Preview section displays an example of how the theme settings might look. Your theme choices could display differently in some applications. You can configure the following theme parameters in the Theme Editor section.

Parameter	Description
Theme name (Required)	The name is the theme's primary identifier. Create a recognizable name for the theme, for example, an institution name or abbreviation. If your institution is MEP-enabled, you may want to use a MEP code as a theme name. This gives you the ability to create code themes based on MEP code. If you change the name of an existing theme and click **Save**, this creates a new theme.
Primary color	Defines the color of the main title bar. Use the drop down or enter the hex code of the color. Theme Editor generates several related shades for borders, backgrounds, and text based on your color choice. View the Preview section to see your choices.

Parameter	Description
	It is recommended that you choose colors that coordinate with your institution logo.
Secondary color	Defines color of other page elements related to the Primary color.
Accent color	Defines additional color for various page elements. As you modify the primary and secondary colors, the Theme Editor automatically determines an accent color that contrasts with the primary and secondary colors. If you prefer, you can specify the Accent color.
Logo URL	The URL of the to include in pages. Create the logo as an SVG file so that it can scale to fit the page The URL must be accessible to application clients because users' browsers will load the logo from this URL.
Save Theme	Saves the theme based on the current Theme name. The theme is saved in the file system of the application server hosting the Theme Editor and is available to all applications. When you click **Save Theme**, the theme is applied to the current page.
New Theme	Clears the theme settings. You can also create a new theme by copying an existing one, changing the Theme Name, and clicking **Save Theme**.
As you create themes, they display in the **Saved Themes** section where you can perform the following functions.	
Load	Loads the theme into the Theme Editor and applies the theme to the current page.
Delete	Deletes the theme from Theme Server. This action cannot be undone.
JSON	Links to the JSON values that encode the theme. This allows you to save the JSON to another location.
CSS	Links to the CSS file generated for the theme. This lets you save the CSS to another location and optimize performance by configuring applications to use the static theme. However, if an application uses a statically saved theme file, you must manually update the static file with any theme changes.

Google Analytics

Use Google Analytics for the Employee Self-Service application.

Access the Banner Applications Configurations (GUACONF) page to set up and change the following configurations and enable Google Analytics for the Employee Self-Service application.

Configuration	Description
`banner.analytics.trackerId=`	Set this value to your institution's Google Analytics Tracker ID. Default is blank
`banner.analytics.allowEllucianTracker=`	Specify whether to allow Ellucian Tracker. Default is true

Configuration example

```
banner.analytics.trackerId = 'UA-83915850-1'
banner.analytics.allowEllucianTracker = false
```

Following are the possible combination of configuration values and their outcome.

- If there is no configuration in the configuration file, then by default the Ellucian tracking ID is enabled and Ellucian tracks analytics.
- If `allowEllucianTracker=true`, Ellucian tracks the analytics data.
- If `allowEllucianTracker=false`, the tracking script will not be added in the gsp page.
- If only the client tracker ID exists in the configuration, then both Ellucian and the client track the Google Analytics data.
- If `allowEllucianTracker=true` and the client tracker ID exists in the configuration, then both the client and Ellucian track the analytics data.
- If `allowEllucianTracker=false` and the client tracker ID both exist in the configuration, then the client tracks the analytics data, but Ellucian does not.

Employee Self-Service Web app extensibility

This feature gives a user defined in SPRIDEN whose Oracle user account has the WTAILORADMIN role the ability to show and hide fields on the user interface.

Note: You must assign the WTAILORADMIN role to any Oracle user who will work with Extensibility.

Web app extensibility for UNIX

Complete the following steps to grant web app extensibility for UNIX.

Procedure

1. Verify that the following directory structure is available. This directory structure is required for extensibility.

   ```
   /u02/BanXE/Extensions/ss_ext/extensions
   /u02/BanXE/Extensions/ss_ext/i18n
   ```

   where `/u02/` is a sample path set up during installation.

   If the previous directory structure does not exist, use the following command to create the directory:

   ```
   mkdir /u02/BanXE/Extensions/ss_ext/extensions
   mkdir /u02/BanXE/Extensions/ss_ext/i18n
   ```

2. Go to the two new directories: `extensions` and `i18n` and provide read and write access configuration to the application server user.

   ```
   webAppExtensibility {
     locations {
       extensions = "/u02/BanXE/Extensions/ss_ext/extensions/"
       resources = "/u02/BanXE/Extensions/ss_ext/i18n/"   }
     adminRoles = "ROLE_SELFSERVICE-WTAILORADMIN_BAN_DEFAULT_M"   }
   ```

Web app extensibility for Windows

Complete the following steps to grant web app extensibility for Windows.

Procedure

1. Verify that the following directory structure is available. This directory structure is required for extensibility.

   ```
   C:/BanXE/Extensions/ss_ext/extensions
   C:/BanXE/Extensions/ss_ext/i18n
   ```

   where `C:/` is a sample path set up during installation.

   If the previous directory structure does not exist, use the following commands to create the structure.

   ```
   mkdir C:/BanXE/Extensions/ss_ext/extensions
   mkdir C:/u02/BanXE/Extensions/ss_ext/i18n
   ```

2. Go to the two new directories: `extensions` and `i18n` and provide read and write access configuration to the application server user.

```
webAppExtensibility {
  locations {
    extensions = "C:/BanXE/Extensions/ss_ext/extensions/"
    resources = "C:/BanXE/Extensions/ss_ext/i18n/"   }
  adminRoles = "ROLE_SELFSERVICE-WTAILORADMIN_BAN_DEFAULT_M"   }
```

Hibernate Secondary Level Caching in MEP environment

Employee Self-Service uses Hibernate Secondary Level Caching, which improves the application performance.

The configuration for this caching is enabled (set to true) by default. If your institution has either the Human Resources or Position Control module enabled for Multi-Entity Processing (MEP), then you must disable (set to false) the following settings as shown.

```
hibernate.cache.use_second_level_cache = false
hibernate.cache.use_query_cache = false
```

Migrate messages.properties file to the database

Migrate the `messages.properties` file to the database which allows users to use the Text Manager function for modifications.

See the *Banner General Text Manager Handbook* for instructions on using Text Manager.

If your institution does not migrate the information to the database, you should review and follow the instructions in the Configure application-specific settings by modifying messages.properties file on page 37 section.

Configure application-specific settings by modifying messages.properties file

To configure application-specific settings, you have the option to modify the groovy file or directly modify the `messages.properties` file, CSS, or JavaScript files within the WAR file. If you do not plan to upload properties from a Banner Self-Service application, then follow the steps in this section.

Some of the application-specific settings that you can configure are the name format, date format, time format, multiple calendars, institution name, and java script.

Customize the messages.properties file

Procedure

1. Copy the war file of the consolidated application `EmployeeSelfService-9.7.war` to a working folder.
2. At a command prompt, run the following JAR command to inflate the contents of the WAR to the current folder.

   ```
   jar -xvf EmployeeSelfService-9.7.war
   ```
3. Delete the existing war file.
4. Locate the UI plugins `banner-hr-employee-profile-ui`, `banner-hr-position-description-ui` `banner-hr-labor-redescription-ui`, and `banner-hr-effort-reporting-ui` in the `WEB-INF` folder.
5. Modify the application specific `messsages.properties` within the appropriate UI plugin as needed.

   In the war file, messages are located local to the plugin as follows:

   ```
   WEB-INF/plugins/<Module>/grails-app/i18n/


   messages.properties
   messages_ar.properties
   messages_en_AU.properties
   messages_en_GB.properties
   messages_en_IE.properties
   messages_en_IN.properties
   messages_es.properties
   messages_fr.properties
   messages_fr_CA.properties
   ```
6. At a command prompt, run the following JAR command. This command will bundle all the files and folders in the current folder.

   ```
   jar –cvf EmployeeSelfService-9.7.war
   ```

Results

The new war file is generated and ready for deployment.

Display name support

You can configure the login user name to display a person's preferred name.

Before you begin

This functionality requires that you have General 8.8.5 installed. You also must execute the seed data script delivered as part of the Banner Database Upgrade 9.11 release package. If the

seed data is not available, refer to the values in the following tables and use the Name Display (GUANDSP) page to enter the information manually.

Procedure

1. Access the Name Display (GUANDSP) page.

   Employee Self-Service provides the following Usage patterns as part of the Common Database Upgrade.

Usage Name	Format
ELFMI	`$Surname Prefix $LastName (,) $FirstName $MiddleNameInitial (.)`
ELPFMI	`$Surname Prefix $LastName (,) $PreferredFirstName $MiddleNameInitial (.)`

   It also provides the following Hierarchy records associated with each Usage pattern.

Usage Name	Product Name	Application Name	Page
ELFMI	EmployeeSelfService	EmployeeSelfService	
ELPFMI	EmployeeSelfService	EmployeeSelfService	EmployeeProfile

2. If the seed data shown in the preceding tables doesn't exist, create these Usage Pattern and Hierarchy records.
3. If you need to apply a different pattern across the Employee Self-Service application, change the corresponding Usage pattern associated with the Employee Self-Service hierarchy.
4. If you need to apply a different pattern across the Employee Profile module while using Preferred First Name, change the corresponding Usage pattern associated with the Employee Profile hierarchy.

   Note: *Do not change the Product Name, Application Name, or Page*; they are used internally.

Format employee addresses

You can configure up to eight lines of the addresses, but you must include at least the first four lines that are delivered as the default.

Before you begin

If you migrated the `messages.properties` file to the Text Manager, you need to edit this format on the Text Manager Interactive Translator (GMATRAN) page or refer to the Customize the messages.properties file on page 38.

About this task

You cannot use separator values such as comma (,) and period (.) if the preceding address attribute can be null.

Procedure

1. If you need to format the address for a different locale, add the appropriate locale file in the directory.
2. Open the file in a text editor and edit the `default.personAddress.linex.format` properties using the elements in the following table to define how employee addresses are displayed.

   If data is not required on the fourth line of the address, then you must use the following format:

   ```
   default.personAddress.line4.format=
   ```

Component	Element
House number	$houseNumber
Street line 1	$streetLine1
Street line 2	$streetLine2
Street line 3	$streetLine3
Street line 4	$streetLine4
City	$city
State or province	$state
ZIP or postal code	$zip
County	$county
Nation	$country

   For example, you can format the employee addresses as follows:

   ```
   default.personAddress.line1.format=$streetLine1
   default.personAddress.line2.format=$streetLine2
   default.personAddress.line3.format=$city
   default.personAddress.line4.format=$state $zip
   ```

3. Rebuild the application WAR file to include your customizations.
4. Redeploy the WAR file to your application server.

Change the phone format

The default phone number format is locale specific. The default phone format for U.S. English is `default.personTelephone.format=$phoneCountry $phoneArea $phoneNumber $phoneExtension`.

Before you begin

If you migrated the `messages.properties` file to the Text Manager, you need to edit this format on the Text Manager Interactive Translator (GMATRAN) page or refer to the Customize the messages.properties file on page 38.

About this task

The phone area and phone number are two different pieces of data in the database. The items are concatenated as specified in the format either with or without a hyphen. You cannot format the phone number, therefore how the number is entered is how it will display.

Procedure

1. To add a new phone locale, add the appropriate locale file in the directory.
2. Add an entry for the `default.personTelephone.format` setting, using the following elements: Area code: `$phoneArea`, Phone number: `$phoneNumber`, Extension: `$phoneExtension`, Telephone country code: `$phoneCountry`, and International access code: `$phoneInternational`.

   For example, `default.personTelephone.format=$phoneCountry $phoneArea $phoneNumber $phoneExtension`.

Change the date format

The default date format is locale specific. The default format for U.S. English is `MM/dd/yyyy` and the `js.datepicker.dateFormat` is `mm/dd/yyyy`. Date formats are case sensitive.

Procedure

1. Copy the war file of the consolidated application `EmployeeSelfService-9.7.war` to a working folder.
2. At a command prompt, run the following JAR command to inflate the contents of the WAR to the current folder.

   ```
   jar -xvf EmployeeSelfService-9.7.war
   ```
3. Delete the existing war file.
4. Locate the plugins `i18n-core` in the WEB-INF folder.

   The `messages.properties` file is located at `grails-app/i18n/messages.properties` within the war file.

   The default for American English is the `messages.properties` file.
5. Open the file using a text editor.

6. Customize the `default.date.format` and `js.datepicker.dateFormat` keys in the `messages_<ISO_language_code>_<ISO_country_code>.properties` file, located in the `EmployeeSelfService\current\i18n` directory.

Related Links

Date format keys on page 42

Date format keys

The `default.date.format` and `js.datepicker.dateFormat` are date format keys that use different specifications to represent the date. The `default.date.format` is associated with the ICU format, which is `dd/MM/yyyy`. The `js.datepicker.dateFormat` is associated with the Java Script or Keith Wood format, which is `dd/mm/yyyy`.

For dates to be displayed properly, the two formats must match. For example, for 1 June, 2012, the calendar parses 01/06/2012 using `js.datepicker.dateFormat`, and when the dates are saved, the date value on the server side is parsed by groovy using `default.date.format` to display 01/06/2012 in the application.

default.date.format

This key determines the date format for display and data entry in the user interface. It must match the ICU specification and can be changed based on locale. For more information about the ICU specification, see http://userguide.icu-project.org/formatparse/datetime.

For the `default.date.format` for June 1, 2014, use one of the following variables for the year:

Year format	Interpretation	Comment
yy	14	Two digit year
yyyy	2014	Four digit year

For the `default.date.format` for June 1, 2014, use one of the following variables for the month:

Month format	Interpretation	Comment
M	6	Single digit month (no leading zero)
MM	06	Double digit month
MMM	Jun	Short month name
MMMM	June	Long month name

For the `default.date.format` for June 1, 2014, use one of the following variables for the day:

Day format	Interpretation	Comment
d	1	Single digit day in a month (no leading zero)
dd	01	Double digit day in a month

js.datepicker.dateFormat

This key determines the date format for the interactive date selection control. It must match the jQuery Keith Wood datepicker format specification. For more information on the Keith Wood datepicker format, see http://keith-wood.name/datepick.html.

For the `js.datepicker.dateFormat` for June 1, 2014, use one of the following variables for the year:

Year format	Interpretation	Comment
yy	14	Two digit year
yyyy	2014	Four digit year

For the `js.datepicker.dateFormat` for June 1, 2014, use one of the following variables for the month:

Month format	Interpretation	Comment
m	6	Single digit month (no leading zero)
mm	06	Double digit month
M	Jun	Short month name
MM	June	Long month name

For the `js.datepicker.dateFormat` for June 1, 2014, use one of the following variables for the day:

Day format	Interpretation	Comment
d	1	Single digit day in a month (no leading zero)
dd	01	Double digit day in a month

Display multiple calendars

Customization of multiple calendars is implemented for the Arabic language (AR). You can customize and display multiple calendars by using a set of `calendar` keys.

Procedure

1. Copy the war file of the consolidated application `EmployeeSelfService-9.7.war` to a working folder.
2. At a command prompt, run the following JAR command to inflate the contents of the WAR to the current folder.

   ```
   jar -xvf EmployeeSelfService-9.7.war
   ```
3. Delete the existing war file.
4. Locate the plugins `i18n-core` in the WEB-INF folder.

   The `messages.properties` file is located at `grails-app/i18n/messages.properties` within the war file.

   The default for American English is the `messages.properties` file.
5. Open the file using a text editor.
6. To display multiple calendars in the application, use the following keys in the `messages_ar.properties` file: `default.calendar`, `default.calendar1`, and `default.calendar2`.

   The `default.calendar2` is optional.

   For example:

   By using the following keys, you can set your default calendar format as Islamic, the first alternate calendar displayed as Gregorian, and the second alternate calendar as Arabic:

   - `default.calendar=islamic`
   - `default.calendar1=gregorian`
   - `default.calendar2=arabic`

Customize CSS

To customize the appearance of the self-service application, you can provide custom CSS and image files.

Procedure

1. Create a CSS file `EmployeeSelfService/current/instance/css/bannerSelfService-custom.css` containing the custom CSS directives.
2. If the `EmployeeSelfService/current/instance/css/images` directory structure does not exist, create the directory structure.

3. If you want to provide custom images, save the images in the `EmployeeSelfService/current/instance/css/images` directory, and in the CSS, specify their paths as a relative URL from the CSS directory.

   For example:

   An image in `css/images/institution-logo.svg` can have the following CSS rule: `background-image: url(images/institution-logo.svg);`.

Change the institution logo

The default layout includes an institution branding area that displays the institution logo. To customize the system or university name, you must provide a custom CSS file to override the default styling with a replacement logo image.

Before you begin

There is not a recommended size for the logo. The system uses Apache FOP to render the PDF using XSLT (styling is *not* through CSS.) In the exploded war structure, there is a `payStub-styles-custom.xsl` file in the `./WEB-INF/fop/payStub` directory. In this file you can define stylesheet attributes that override the defaults. Enter the following lines in the file to shrink the logo to fit the area:

```
<xsl:attribute-set name="employer-logo">
<xsl:attribute name="content-width">scale-to-fit</xsl:attribute>
<xsl:attribute name="content-height">100%</xsl:attribute>
<xsl:attribute name="width">100%</xsl:attribute>
<xsl:attribute name="scaling">uniform</xsl:attribute>
</xsl:attribute-set>
```

About this task

The default layout styles the institutional branding area as follows:

```
.institutionalBranding {
position:relative;
float:left;
left:10px;
top:11px;
height:78px;
width:54px;
background:url("images/ellucian-university-logo-sm.png") no-repeat;}
```

Procedure

1. To override the default image, create a CSS file named `EmployeeSelfService/current/instance/css/ bannerSelfService-custom.css`.
2. In the CSS file you have created, enter `.institutionalBranding {background-image: url("./images/ institutionLogo.png");}`.

3. Replace the logo image in the following directory: `EmployeeSelfService/current/instance/css/images/institutionLogo.png`.
4. To deploy your updates, you must rebuild and redeploy the WAR file.

Related Links

Regenerate the WAR file on page 47

Customize JavaScript

You can customize the java script to modify the behavior of the web pages of your application by placing your JavaScript file named bannerSelfService-custom.js in the following location: EmployeeSelfService/current/instance/js/bannerSelfService-custom.js.

Procedure

1. Create a JavaScript file, `current/instance/css/bannerSelfService-custom.js`.
2. In the JavaScript file you have created, enter the custom JavaScript code.
3. To deploy your updates, you must rebuild and redeploy the WAR file.

Regenerate the WAR file

After the shared and application-specific configurations are complete, the application WAR file can be regenerated to include your customizations and application-specific settings. The WAR file can then be deployed into your specific application server. The `systool` is used to create the WAR file.

About this task

Application uses the configuration files in the WAR file unless you override them by specifying the environment variable. For example, you can override the location of the `banner_congifuration.groovy` file by setting the environment variable as follows: `BANNER_APP_CONFIG=/path/to/banner_configuration.groovy`

Procedure

1. Change your current working directory to the product home directory: `EmployeeSelfService/current/installer`.
2. Run the `ant` command which builds the `systool` module.
   For UNIX systems, ensure that the `ant` file is executable. For example,

   ```
   chmod +x ant
   ```

   For example,

   ```
   $ cd EmployeeSelfService/current/installer

   $ ./ant
   ```

3. Use the `systool` module to create the WAR file.

Operating system	Command
UNIX	`$ bin/systool war`
Windows	`> bin\systool war`

   The WAR file is created in the `EmployeeSelfService/current/dist` directory.

Configure the web application server and deploy the WAR file

You can configure a web application server such as Tomcat or WebLogic and deploy the WAR file on the web application server. It is recommended that you can secure the web application traffic by using standard TLS encryption, which is supported by the application server software.

For more information about enabling HTTPS support for web applications, see your application server documentation.

Configure the Tomcat server on page 48

Configure the WebLogic server on page 55

Configure the Tomcat server

You can configure the Tomcat server and then deploy the WAR file to the Tomcat server.

Before you begin

Ensure that you download and install either Tomcat 7 or Tomcat 8 versions. For more information about downloading and installing the Tomcat server, see http://tomcat.apache.org.

About this task

If you choose to install the application on a Tomcat server, you do not need to install it on WebLogic.

Procedure

1. Locate the Oracle JDBC jar files, `ojdbc6.jar` and `xdb6.jar` in the `EmployeeSelfService\current\lib` directory.

   The account that runs the Tomcat application server must configure environment settings to support the application.
2. On a Linux system, ensure that `CATALINA_HOME` is defined to reference your Tomcat software installation location.

   Note: This step must be executed on a Linux system only.

   For example, in the `CATALINA_HOME=/opt/apache-tomcat-.xx`, `xx` indicates the point version of Tomcat you have installed.
3. Define `CATALINA_OPTS` to configure the following JVM settings: `CATALINA_OPTS=-server -Xms2048m -Xmx4g -XX:MaxPermSize=512m`

Note: If you are deploying multiple Banner 9.x applications to the same Tomcat server, `-Xmx` must be increased by `2g` and `-XX:MaxPermSize` must be increased by `128m`. You should deploy Banner 9.x administrative applications to one Tomcat server instance and Employee Self-Service application to a separate Tomcat server instance.

You can define this variable in the account's profile startup script or you can add this definition in `$CATALINA_HOME/bin/catalina.sh` for Linux or `catalina.bat` for Windows.

4. **Optional:** If you install Tomcat as a Windows service, few JVM arguments must be specified.
   a) From the Windows **Start** menu, select **Configure Tomcat** application.
   b) Select the **Java** tab.
   c) In the **Java Options** field, add `-XX:MaxPermSize=384m`.
   d) Set the initial memory pool to 2048.
   e) Set the maximum memory pool to 4096.
   f) Save the settings.
   g) Restart the Tomcat Windows service.
   h) Redeploy Employee Self-Service to reconnect it with the Oracle Advance Queue.
5. Define the JNDI datasource resource name for the application.
   a) Edit `$CATALINA_HOME/conf/context.xml`.
   b) Uncomment `<Manager pathname="" />` to disable Tomcat session persistence.
   For example: change the following:

```
<!-- Uncomment this to disable session persistence
across Tomcat restarts -->
<!--<Manager pathname="" />-->
```

   to

```
<!-- Uncomment this to disable session persistence
across Tomcat restarts -->
<Manager pathname="" />
```

   c) Add the ResourceLink definitions inside the `<Context>` element.

```
<ResourceLink global="jdbc/bannerDataSource"
name="jdbc/bannerDataSource"
type="javax.sql.DataSource"/>

<ResourceLink global="jdbc/bannerSsbDataSource"
name="jdbc/bannerSsbDataSource"
type="javax.sql.DataSource"/>
```

   d) Save your changes in `context.xml`.

e) Edit `$CATALINA_HOME/conf/server.xml` to configure the database JNDI resource name and connection pool configuration.

f) Add the Resource definitions inside the `<GlobalNamingResources>` element.

```
<Resource name="jdbc/bannerDataSource" auth="Container"
type="javax.sql.DataSource"
driverClassName="oracle.jdbc.OracleDriver"
url="jdbc:oracle:thin:@//hostname:port/service_name"
username="banproxy" password="the_banproxy_password"
initialSize="5" maxActive="100" maxIdle="-1" maxWait="30"
validationQuery="select 1 from dual"
testOnBorrow="true"/>

<Resource name="jdbc/bannerSsbDataSource" auth="Container"
type="javax.sql.DataSource"
driverClassName="oracle.jdbc.OracleDriver"
url="jdbc:oracle:thin:@//hostname:port/service_name"
username="ban_ss_user" password="ban_ss_user_pasword"
initialSize="5" maxActive="100" maxIdle="-1" maxWait="30"
validationQuery="select 1 from dual"
testOnBorrow="true"/>
```

If you are using Tomcat 8, then you must include the following line in both of these blocks:

```
accessToUnderlyingConnectionAllowed="true"
```

For example: if your database server's name is `myserver.university.edu` and the Oracle TNS Listener is accepting connections on port 1521 and your database service's name is `SEED`, then the URL is `jdbc:oracle:thin:@//myserver.university.edu:1521/SEED`.

g) Save your changes in `server.xml`.

h) Copy the Oracle JDBC jar file `(ojdbc6.jar and xdb6.jar)` from the `EmployeeSelfService/current/lib` directory to the `$CATALINA_HOME/lib` directory.

i) Start the application server, `$CATALINA_HOME/bin/startup` to validate the configuration of the Tomcat server.

For example: for Linux,

```
cd $CATALINA_HOME
$ bin/startup.sh
```

For Windows,

```
cd %CATALINA_HOME%
> bin\startup.bat
```

j) Browse `http://servername:<port>`.

Example

To override the configuration that was added into the WAR file, you must set system properties to point to external configuration files. For example, to point to a configuration file residing in the `PRODUCT_HOME` directory, export `JAVA_OPTS="-DBANNER_APP_CONFIG=/PRODUCT_HOME/shared_configuration/banner_configuration.groovy -DBANNER_EMPLOYEE_SSB_CONFIG=/PRODUCT_HOME/EmployeeSelfService/current/instance/config/EmployeeSelfService_configuration.groovy"`.

Configure Java management extension

This is an optional step that enables you to monitor or debug the application.

About this task

Java Management Extensions (JMX) is a Java technology that supplies tools for managing and monitoring applications, system objects, devices, and service oriented networks. Enabling JMX connections allows you to remotely monitor and debug the application server.

Procedure

1. Add the following options to the `catalina.sh` file.

   ```
   set CATALINA_OPTS=-Dcom.sun.management.jmxremote
   -Dcom.sun.management.jmxremote.port=8999
   -Dcom.sun.management.jmxremote.ssl=false
   -Dcom.sun.management.jmxremote.authenticate=false
   -Djava.rmi.server.hostname=your.hostname.com
   ```

2. Restart the Tomcat server.
3. Redeploy Employee Self-Service to reconnect it with the Oracle Advance Queue.
4. Change the `java.rmi.server.hostname` value to the hostname or IP address of the machine where Tomcat is installed.

   For example:

   ```
   -Djava.rmi.server.hostname=prod.appserver1.com
   or
   -Djava.rmi.server.hostname=149.24.3.178
   ```

5. JMX does not define a default port number to use. If necessary, change `com.sun.management.jmxremote.port` to use port 8999.

   Ellucian recommends that you connect remotely to the Tomcat server using JMX.

   Warning! Ensure that the **jmxremote.authenticate** parameter is not set to *False* in a production environment. If it is set to *False*, it does not require connections to be authenticated and will create a security threat. For more information, see http://tomcat.apache.org/tomcat-6.0-doc/monitoring.html#Enabling_JMX_Remote.

Deploy the WAR file to the Tomcat server

The systool that is used to create the WAR file can also be used to deploy the WAR file to a Tomcat container. You should deploy 9.x administrative applications and Employee Self-Service application to separate Tomcat servers to increase performance.

About this task

Note: The `systool` does not provide the capability to undeploy or redeploy an application. If you are redeploying the application, you must use the Tomcat Manager web application to undeploy the existing application.

The `systool` supports deploying the `dist/WAR` file using the Tomcat Manager web application. Because environments vary significantly with respect to user privileges, clustering approach, web container version, operating system, and more, the target may or may not be suitable for your use.

Note: You can also deploy the WAR file to the Tomcat server by copying the WAR file to the Tomcat `webapps/` directory.

To use the target, you must provide the following information:

- URL- This is the URL of the manager application in the Tomcat server. For example: `http://localhost:8080/manager`.
- Username- This Tomcat server username must have privileges to deploy WAR files.
- Password- This is the password of the Tomcat server user.

Username/password combinations are configured in your Tomcat user database `<TOMCAT_HOME>\conf\tomcat-users.xml`. For Tomcat 8.x, you must configure at least one username/password combination with the manager role. For example: `<user username="tomcat" password="tomcat" (your password) roles="manager-gui, manager"/>`.

Note: The roles in Tomcat server changed between point releases in version 8.x. Refer to the Tomcat documentation specific to your release for information on enabling access to provide the appropriate role to a user account for deployment.

Procedure

1. Navigate to the `EmployeeSelfService\current\installer` directory.
2. Deploy Tomcat.

Operating system	Instruction
Unix	`$ bin/systool deploy-tomcat`
Windows	`> bin\systool deploy-tomcat`

3. Enter the URL, `[]: http://localhost:8080/manager` for the Tomcat Manager.
   This URL will be accessed to deploy the WAR file into the container.
4. Enter a valid Tomcat username to deploy the WAR file.

This user must have the `manager-gui` role.

For example: `[]: tomcat`

5. Enter the Tomcat password for the user:
   This password will not be persisted.

   For example: `[]: password`
6. Access the web application `http://servername:<port>/EmployeeSelfService`.

WebLogic server

To configure the web application and deploy the WAR file to the WebLogic server, you must perform certain tasks such as verify certain WebLogic prerequisites, set up the cookie path for WebLogic installation, create a WebLogic machine, and create a WebLogic server.

If you choose to install the application on a WebLogic server, you need not install it on Tomcat.

Verify WebLogic prerequisites

Ensure that the WebLogic prerequisites are verified before configuring your WebLogic server.

Procedure

1. Ensure that WebLogic is installed.
   WebLogic can be downloaded and installed from the Oracle web site.
2. Ensure that the minimum requirement for OFM is 11.1.1.7 or 11.1.1.9 with WebLogic 10.3.6.
   Banner 9 web applications are also supported on OFM WebLogic 12c (12cR1, version 12.1.x).
3. Ensure that both the WebLogic node manager and the administration server are started.
   The administration server can be accessed using the following URL, `http://server:7001/console`.

Set up the cookie path

For a WebLogic installation, the cookie path needs to match the location where the application is deployed. Otherwise, the cookies will not be found by the application. If this change is not made, users will be prompted to log in each time they switch between applications.

About this task

Add the following in the `weblogic.xml`:

```
<wls:session-descriptor>
<wls:cookie-path>/<WebApp_Root_Context></wls:cookie-path>
</wls:session-descriptor>
```


Procedure

1. On the Shell command line, in a directory containing the WAR file, enter `jar xvf application_context.war WEB-INF/weblogic.xml`.
2. Enter `vi WEB-INF/weblogic.xml`
3. Add the code listed above for the cookie path to the appropriate place in the file.
4. Enter `jar uvf application_context.war WEB-INF/weblogic.xml`.
5. Redeploy the application.

Example

WebLogic file

```
<?xml version="1.0" encoding="UTF-8"?>
<wls:weblogic-web-app
xmlns:wls="http://www.bea.com/ns/weblogic/weblogic-web-app"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
 http://
java.sun.com/xml/ns/javaee/web-app_2_5.xsd http://
www.bea.com/ns/
weblogic/weblogic-web-app http://www.bea.com/ns/weblogic/
weblogic-webapp/
1.0/weblogic-web-app.xsd">
<wls:weblogic-version>10.3.0</wls:weblogic-version>
<wls:container-descriptor>
<wls:prefer-web-inf-classes>true</wls:prefer-web-inf-
classes>
<wls:show-archived-real-path-enabled>true</wls:show-
archived-real-pathenabled>
</wls:container-descriptor>
<wls:session-descriptor>
<wls:cookie-path>/application_context</wls:cookie-path>
</wls:session-descriptor>
</wls:weblogic-web-app>
```

Create a WebLogic machine

For configuring the WebLogic server, a WebLogic machine must be created. You need not create a WebLogic machine definition again if you have already created a machine earlier.

Procedure

1. In the **Change Center frame**, click **Lock & Edit**.
2. In the **Domain Structure frame**, click **(+)** to expand and view the list of environments.
3. Click the **Machines** link.
4. Click **New**.

5. Enter a machine name and click **Next**.
6. Accept the defaults and click **Finish**.
7. In the **Change Center frame**, click **Activate Changes**.

Configure the WebLogic server

You can configure the WebLogic server and then deploy the WAR file to the WebLogic server. If you previously created a WebLogic server for the application, you can use the same server.

Procedure

1. In the **Change Center frame**, click **Lock & Edit**.
2. In the **Domain Structure frame**, click **(+)** to expand and view the list of environments.
3. Click the **Servers** link.
4. Click **New**.
5. Enter a server name and server listen port.
   You can have the server name as Banner9-SS and server listen port as 8180.
6. Click **Finish**.
7. Click the newly created server link.
8. In the **General** tab, assign the machine to this server.
9. Click **Save**.
10. Select the **Server Start** tab.
11. Add the following to the **Arguments** text area: `-server -Xms2048m -Xmx4g -XX:MaxPermSize=512m`.

    Note: If you are deploying multiple Banner 9.x applications to the same WebLogic server, increase `-Xmx` (max heap) by 2g and `-XX:MaxPermSize` by 128m. You should deploy Banner 9.x administrative applications to one WebLogic server instance and Employee Self-Service applications to a separate WebLogic server instance.

    a) To override the configuration that was added into the WAR file, you can set system properties to point to external configuration files by appending the following to the **Arguments** text area: `-DBANNER_APP_CONFIG=<full file path to banner_configuration.groovy> -DBANNER_EMPLOYEE_SSB_CONFIG=<full file path to EmployeeSelfService_configuration.groovy>`.
12. Click **Save**.
13. In the **Change Center** frame, click **Activate Changes**.
14. In the **Domain Structure** frame, click the **Servers** link.
15. Select the **Control** tab.
16. Select the check box next to your new server definition.
17. Click **Start**.

Configure weblogic.xml file to make Banner 9.x JSession cookie secure

To make the Banner 9.x JSession cookie secure, certain configuration information must be added to the `weblogic.xml` file. This configuration is only specific to WebLogic server.

About this task

The configuration changes should be added based on specifications at your institution. After the cookie has been secured, the same application cannot be accessed through a non-SSL port. These configuration changes should only be applied if the SSL is in use.

Procedure

Add the following configuration information to the `weblogic.xml` file to make the Banner 9.x JSession cookie secure: `<wls:session-descriptor> <wls:cookie-secure>true</wls:cookie-secure> <wls:url-rewriting-enabled>false</wls:url-rewriting-enabled> </wls:session-descriptor>`.

Update Oracle JDBC JAR files on the WebLogic server

WebLogic releases include specific versions of the JDBC JAR files, but the versions used are those that ship with the Banner application release.

About this task

For more information about JDBC JAR files, see https://docs.oracle.com/middleware/1212/wls/JDBCA/third_party_drivers.htm#JDBCA231.

Procedure

1. Copy the Oracle JAR files (`ojdbc6.jar` and `xdb6.jar`) from the `$PRODUCT_HOME/current/lib` directory to the `$MIDDLEWARE_HOME/modules` directory.
   - `$PRODUCT_HOME` is where the application's release zip file is unpacked and installed.
   - `$MIDDLEWARE_HOME` is the location where Oracle WebLogic is installed.
2. For Linux or Unix servers, edit the `setDomainEnv.sh` file under the `$MIDDLEWARE_HOME/user_projects/domains/<CUSTOM_DOMAIN>/bin` folder and update the `ADD EXTENSIONS` comment with additional information.

   For example:

```
#ADD EXTENSIONS TO CLASSPATH
export MIDDLEWARE_HOME="/u01/app/oracle/Middleware"
export WLS_MODULES="${MIDDLEWARE_HOME}/modules"
export EXT_PRE_CLASSPATH="${WLS_MODULES}/
xdb6.jar:${WLS_MODULES}/ojdbc6.jar"
```

If you plan to copy and paste the configuration settings into the `setDomainEnv.sh` file, ensure that there is no typo or special characters that get carried over (especially with double quotes on the variable declarations). If you see `Class NotFoundException` in your log files, there might have been a typo when you edited the `setDomainEnv.sh` file and the `xdb6.jar` or `ojdbc6.jar` file cannot be found during Application startup.

3. For MS Windows servers, edit the `setDomainEnv.cmd` under the `$MIDDLEWARE_HOME/user_projects/domains/<CUSTOM_DOMAIN>/bin` folder and update the `ADD EXTENSIONS` comment with additional information.

   For example:

   ```
   @REM ADD EXTENSIONS TO CLASSPATH
   set MIDDLEWARE_HOME=D:\Oracle\Middleware
   set WLS_MODULES=%MIDDLEWARE_HOME%\modules set
   EXT_PRE_CLASSPATH=%WLS_MODULES%\xdb6.jar;%WLS_MODULES%\ojdbc6
   .jar
   ```

   If you plan to copy and paste the configuration settings into the `setDomainEnv.cmd` file, ensure that there is no typo or special characters that get carried over (especially with double quotes on the variable declarations). If you see `Class NotFoundException` in your logs, there might have been a typo when you edited the `setDomainEnv.cmd` file and the `xdb6.jar` or `ojdbc6.jar` file cannot be found during Application startup.

4. Restart the WebLogic Managed Server.
5. Redeploy Employee Self-Service to reconnect it with the Oracle Advance Queue.

Create an administrative datasource and connection pool

You can use this task to configure an application's connection to the Oracle database for administrative users. If you have already created an administrative datasource and connection pool, you need not create this again.

Procedure

1. In the **Change Center** frame, click **Lock & Edit**.
2. In the **Domain Structure** frame, click **(+)** to expand **Services** and then select **Data Sources**.
3. Click **New**.
4. Select **Generic DataSource**.
5. Specify a datasource name.

   For example, `Banner9DS`.
6. Specify the JNDI name.

   For example: a JNDI name can be `jdbc/bannerDataSource`.
7. Specify `Oracle` for Database Type and then click **Next**.
8. Select **Oracle Driver (Thin) for Service Connections** and then click **Next**.
9. Clear the **Supports Global Transactions** check box and then click **Next**.

10. Enter the database name, host name, port, user name, password, and password confirmation, and then click **Next**.

    For example:

Database name	BAN9
Host name	yourhostname.yourdomain.com
Port	1521
UserName	banproxy
Password	your_password

11. Click **Test Configuration**.
12. Click **Next** for the connection test to be successful.
13. Select the server that you previously created to allow the datasource to be deployed and used by this server.
14. Click **Finish**.
15. Select the datasource link that you created.
16. Select the **Connection Pool** tab.
    a) Set the Initial Capacity parameter to specify the minimum number of database connections to be created when the server starts up.

       For example: Initial Capacity = 5

    b) Set the Maximum Capacity parameter to specify the maximum number of database connections that can be created.

       For example: Maximum Capacity = 100
17. Change `Statement Cache Type = Fixed`.
18. Change `Statement Cache Size = 0`.
19. Click **Save**.
20. In the **Change Center** frame, click **Activate Changes**.

Create a self-service datasource and connection pool

You can use this task to configure an application's connection to the Oracle database for self-service users. If you have already created a self-service datasource and connection pool, you need not create this again.

Procedure

1. In the **Change Center** frame, click **Lock & Edit**.
2. In the **Domain Structure** frame, click **(+)** to expand **Services** and then select **Data Sources**.
3. Click **New**.
4. Select **Generic DataSource**.

5. Specify a datasource name.
   For example, `Banner9DS`.
6. Specify the JNDI name.
   A JNDI name can be `jdbc/bannerSsbDataSource`.
7. Specify `Oracle` for Database Type and then click **Next**.
8. Select **Oracle Driver (Thin) for Service Connections** and then click **Next**.
9. On the **Transaction Options** page, clear the **Supports Global Transactions** check box and then click **Next**.
10. Enter the database name, host name, port, user name, password, and password confirmation, and then click **Next**.

Database name	BAN9
Host name	yourhostname.yourdomain.com
Port	1521
UserName	ban_ss_user
Password	your_password

11. Click **Test Configuration**.
12. Click **Next** for the connection test to be successful.
13. Select the server that you previously created to allow the datasource to be deployed and used by this server.
14. Click **Finish**.
15. Select the datasource link that you created.
16. Select the **Connection Pool** tab.
    a) Set the Initial Capacity parameter to specify the minimum number of database connections to be created when the server starts up.
       Initial Capacity = 5
    b) Set the Maximum Capacity parameter to specify the maximum number of database connections that can be created.
       Maximum Capacity = 100
17. Change `Statement Cache Type = LRU`.
18. Change `Statement Cache Size = 20`.
19. Click **Save**.
20. In the **Change Center** frame, click **Activate Changes**.

Configure server communication

Configure the application server to use the administrative and self-service data sources.

Procedure

1. In the **Change Center** frame, click **Lock & Edit**.
2. In the **Domain Structure** frame, click **(+)** to expand **Services** and then select **DataSources**.
3. Select the datasource for the administrative application.
   For example: `Banner9DS`
4. Select the **Targets** tab.
5. Select the check box for the self-service server.
   For example: `Banner9-SS`
6. Click **Save**.
7. In the **Change Center** frame, click **Activate Changes**.

Deploy and start the application in the WebLogic server

The WebLogic server uses the deployed WAR file and implements the code in the WAR file by starting the application.

Procedure

1. Change the name of the WAR file to remove the version number.
   For example: Change

   ```
   EmployeeSelfService/current/dist/
   EmployeeSelfService-9.7.war
   ```

   To

   ```
   EmployeeSelfService/current/dist/
   EmployeeSelfService.war
   ```

2. Access the administration server at `http://server:7001/console`.
3. In the **Domain Structure** frame, select the **Deployments** link.
4. In the **Change Center** frame, select **Lock and Edit**.
5. Click **Install**.
6. Select the WAR file to be deployed and then click **Next**.
   The file is located at `EmployeeSelfService/current/dist`.
7. Select **Install this deployment** as an application and then click **Next**.
8. Select the target server on which to deploy this application and then click **Next**.
   For example, `Banner9-SS` is a target server.

9. Click **Finish**.
10. In the **Change Center** frame, click **Activate Changes**.
11. Select the deployed application and then click **Start**.
12. Select **Servicing all request**.
13. Access the application at `http://servername:<port>/<web application>`.
    For example: `http://localhost:8080/EmployeeSelfService`.
14. Log in to the application using a valid username and password.

Employee Profile configuration

You can use the configuration checklist, which is a quick reference guide that provides you with a list of the decisions you need to make to configure Employee Profile and Dashboard pages.

Note: These configuration options affect only Employee Self-Service 9.x application. Your choices regarding these settings do not affect any other part of Banner.

For information about what you must do to configure the Employee Profile and Dashboard pages, see the *Banner Employee Profile Handbook*, which you can download from the Ellucian Customer Support site.

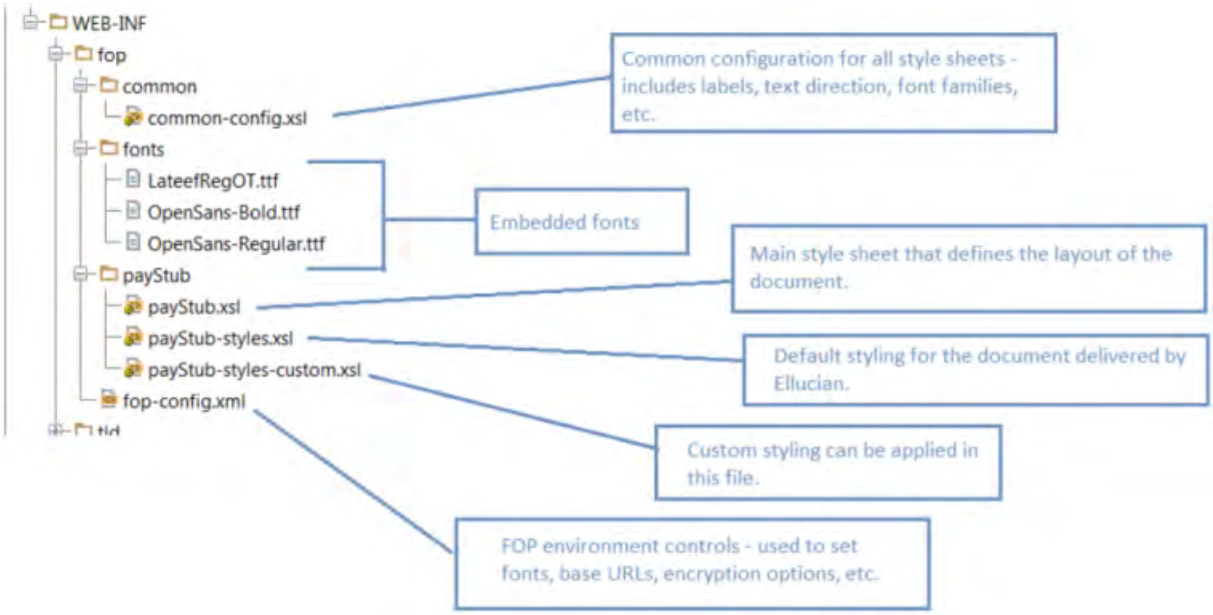
Customize the pay stub PDF

You can customize the pay stub PDF or use the default PDF styling that is generated when you create a PDF version of the pay stub.

About this task

Note: The PDF version of the pay stub supports right-to-left (RTL) display and Arabic characters.

The pay stub PDF is generated using XSL-FO and Apache FOP. Employee Profile transforms a well-formed XML file containing employee pay stub data into a pay stub PDF document using XSLT (Extensible Style Sheet Language Transformations). Below is the FOP tree structure showing all artifacts used in generating the pay stub PDF file. All style sheets have been organized in a way that separates style from layout (similar to the HTML/CSS concept).



Procedure

To customize the pay stub PDF, edit the default styles or attribute sets in the `paystub-styles-custom.xsl` file.

For more information about making changes to the xsl, see the `paystub-styles-custom.xsl` file.

Results

Any customizations to the pay stub styling or layout style sheets are overwritten during subsequent deployments of the application. You must reapply your customizations after each new deployment of 9.x.

Ellucian recommends two options for minimizing the time this requires:

- Keep all customizations limited to the `payStub-styles-custom.xsl`.
- Copy the entire FOP folder to another location on the web server so that customizations are retained after a new deployment. The new location of the FOP folder must be configured before restarting the application.

Position Description configuration

Position Description application can be configured to integrate with Banner Document Management (BDM) and CornerStone on Demand (CSOD).

Configure Position Description to integrate with BDM

This task helps a Banner Document Management (BDM) administrator, or the person at your institution authorized to support BDM, configure the Position Description application for a successful Position Description and BDM integration. This integration enables you to attach electronic documents to a position description.

Before you begin

- You must install BDM 8.6 on your system.

  The source code for the Position Description integration with BDM is included as part of the Banner Employee Self-Service installation and not in the BDM source objects delivered in BDM 8.6.
- You must install EMC ApplicationXtender 7.0.260 (7.0 SP1 is preferred), EMC ApplicationXtender Web Access, and EMC ApplicationXtender Web Services on your system.

About this task

The Position Description configuration contains BDM-related parameters. A BDM administrator, or the person at your institution authorized to support BDM, is required to assist with creating the BDM shared user account and providing some of the parameters defined on the Banner Applications Configurations (GUACONF) page under the global configurations.

BDM uses `B-H-POSN` application to manage the documents uploaded from the Position Description Self-Service page. Additional index fields are added to `B-H-POSN` to support this integration.

For more information about BDM, see the following information resources:

- *Banner Document Management Administration Guide 8.6* or later
- *Banner Document Management Installation 8.6* (if upgrading to BDM 8.6)
- *Banner Document Management Release Guide 8.6*

Procedure

1. Create a folder for temporarily uploading the document files (BDM attachments) on the system where the Banner Employee Self-Service application is deployed.

   For example, on a Windows system, you can create the folder as follows: `C:/BDM_DOCUMENTS_FOLDER/` and on a Linux system, you can create the folder as follows: `/u02/Tomcat7/BDM_DOCUMENTS_FOLDER`.
2. Ensure that the application server has read-write permissions for the folder you have created so that the folder can be accessed to create BDM documents.

3. Create a BDM user account in ApplicationXtender.

   The BDM user account in ApplicationXtender must also be a Banner user and must have appropriate privileges on the PTRUSER page to view attachments from the Position Description application page.

   a) Create a new BDM user in ApplicationXtender.

      The BDM user and the password assigned to the user are required in the configuration file which needs to be updated later in the following steps.

      For more information about creating a new BDM user in ApplicationXtender, see the *Banner Document Management Administration Guide*.

4. Assign privileges to the new BDM user.

   a) In ApplicationXtender, navigate to the **Global-level** drop-down box.

   b) Choose the **Multiple Logins** check box.

   c) Select the application to which the BDM user has access.

   d) Assign these privileges: `Scan/Index Online, Batch Scan, Batch Index, Modify Index, Display, Delete Doc, and Add Page`.

5. Click **Apply**.

6. Exit ApplicationXtender.

7. Verify settings in the ApplicationXtender `web.config` file.

   The `web.config` file is located in your web site's ApplicationXtender Web Access NET folder at `C:\Inetpub\wwwroot\AppXtender\Web.config`.

   a) Open the `web.config` file.

   b) Search for the `AxWebServicesDocumentPointer` parameter.

   c) If the value of `AxWebServicesDocumentPointer` is not equal to 2, then modify the value as follows: `<add key="AXWebServicesDocumentPointer" value="2"/>`.

   d) Save and close the `web.config` file.

   e) Access and locate the ApplicationXtender Web Services `web.config` file at `C:\inetpub\wwwroot\AppXtenderServices`.

   f) Repeat steps `a` to `d` to set the value of `AxWebServicesDocumentPointer` in the `web.config` file.

   g) Perform an `IIRESET` to implement the changes to `web.config` file.

8. Update the BDM server connection parameters on the Banner Applications Configurations (GUACONF) page under the global configurations.

Results

Position Description application and BDM should be successfully integrated.

BDM parameters

You must update certain Banner Document Management (BDM) parameters to establish a successful connection between Position Description and the BDM server that allows you to attach electronic documents to a position description.

BDM parameters are located on the Banner Applications Configurations (GUACONF) page under the global configurations. Each parameter that requires you to define a value has the default value of `<UPDATE ME>`.

The following table lists the BDM parameters and their descriptions.

Parameter name	Description
`bdm.enabled`	`True` = The integration with BDM is being used to create, update, and delete documents. `False` = The integration with BDM is not being used.
`bdm.file.location`	The temporary file upload directory location of the folder created for uploading documents.
`AXWebServicesUrl`	Web Service URL of ApplicationXtender For example, `http://<AX_Server_Name>:port/appxtender/axservicesinterface.asmx`
`AXWebAccessURL`	ApplicationXtender logon URL For example, `http://<AX_Server_Name>:port/appxtender/`
`Username`	User name of BDM user created in the ApplicationXtender, which is used to connect to the BDM server.
`Password`	An empty password is defined in Configuration. This will be updated by an external script as defined in the section "Configure Encrypted BDM Password in DB".
`BdmDataSource`	The BDM/ApplicationXtender Data Source name.
`AppName`	The application name is `B-H-POSN`.

Edit BDM configurations

Update the Banner Applications Configurations (GUACONF) page so that it uses the configurations in the database.

Procedure

Configure the GLOBAL application attributes in the GUACONF page as follows:

```
bdm.enabled
bdm.file.location
bdmserver.AXWebServicesUrl
bdmserver.AXWebAccessURL
bdmserver.Username
bdmserver.BdmDataSource
bdmserver.AppName
bdmserver.Password
```

This allows you to define all BDM configurations as GLOBAL configurations in the database.

Note: For additional information, refer to the section Configure encrypted BDM password in database on page 68.

Configure Position Description to integrate with CSOD

Position Description data is used by CornerStone on Demand (CornerStone Talent Management Suite). Position Description integration with CSOD allows a Position Description administrator to extract information into files for use in Talent Management Suite.

Before you begin

- Ensure that the SFTP location specified by CSOD for Position OU (Organizational Unit) data sweep is defined.
- Ensure that the Position Description administrator role is defined in Web Tailor.

About this task

The advanced search functionality allows the search for active and inactive position descriptions based on Position Number, Position Classification, Employee Classification, Organization, Effective Date range and End Date range.

Procedure

1. Create a CornerStone export file folder, `csod.base.dir` on the application server where the Position Description application is deployed.

   The location of the folder is where the CSOD export files are created temporarily.
2. For all the exported files, verify that the delimiter is set to the default value as follows: `csod.file.delimiter = Defaulted to "|".`

3. From the **Position Description Library**, click **Export** to export the data.
   The filtered data is exported as a zip file that includes information about position, responsibilities, education, certificates, skills, and examinations.
4. Copy the zip file to the SFTP location as specified by CornerStone for Position OU data sweep by CornerStone.

Results

The new position descriptions are now available in CSOD.

Configure encrypted BDM password in database

You have the option to update the BDM password when the password changes in the BDM environment.

Before you begin

You must deploy Employee Self-Service 9.7 before you perform this step.

Procedure

1. Access the Banner Applications Configurations (GUACONF) page.
2. Enter the Application ID GLOBAL.
3. Access the Configuration tab.
4. Modify the bdmserver.Password.
5. Save the changes.

Results

The system saves the encrypted password.

Note: You must resupply the encrypted password in the GUACONF whenever the you update the value in the Institution Profile (GSASECR) page.

Labor Redistribution configuration

An attribute must be configured to determine which Labor Redistribution link - 8.x or 9.x - needs to be enabled in the Employee Self-Service application.

Enable Labor Redistribution 9.x link in Employee Self-Service

The link to the Labor Redistribution 9.x application is defined in the Banner Applications Configurations (GUACONF) page under the Employee application as `//Labor Redistribution Application Link`. The `banner.SS.laborRedistributionAppLinkAvailable` attribute enables this link.

About this task

In Employee Self-Service, you can display either the 8.x or 9.x Labor Redistribution link, but not both. By default, it displays the link to Labor Redistribution 9.x.

Procedure

1. In the Banner Applications Configurations (GUACONF) page under the Employee application, disable the link to Labor Redistribution 8.x by setting `banner8.SS.laborRedistribution8xLinkAvailable` to `N`.
2. Enable the link to Labor Redistribution 9.x by setting `banner.SS.laborRedistributionAppLinkAvailable` to `Y`.

Effort Reporting configuration

You must configure the `//Effort Reporting Application Link` attribute to define whether to enable the 8.x or 9.x Effort Reporting link in the Employee Self-Service application.

Enable Effort Reporting 9.x link in Employee Self-Service

The link to the Effort Reporting 9.x application is defined in the Banner Applications Configurations (GUACONF) page under the Employee application as `//Effort Reporting Application Link`. The `banner.SS.effortReportingAppLinkAvailable` attribute enables this link.

About this task

In Employee Self-Service, you can display either the 8.x or 9.x Effort Reporting link, but not both. By default, the link to Effort Reporting 9.x is displayed.

Procedure

1. In the Banner Applications Configurations (GUACONF) page under the Employee application, disable the link to Effort Reporting 8.x by setting `banner8.SS.effortReporting8xLinkAvailable` to `N`.
2. Enable the link to Effort Reporting 9.x by setting `banner.SS.effortReportingAppLinkAvailable` to `Y`.

PHPECEX Process

If you plan to execute the PHPECEX JobSub Process, you must connect using the `banproxy` privilege.

See the section Verify Oracle user accounts to connect through banproxy on page 13 for more information.

Time Entry and Leave Management Configuration

If you want to use the Notifications feature and send messages to employees throughout time entry processing, you must configure the Banner Time Entry and Leave Management application to integrate with Banner Communication Management (BCM).

Configure Time Entry and Leave Management to integrate with BCM

Perform these steps to configure the Banner Time Entry and Leave Management application to integrate with BCM.

Before you begin

You must install Banner Communication Management 9.4.

Procedure

1. Start the BCM application and log in.
2. Add a Hosted Server organization.

   This is where Employee Self-Service sends undelivered emails when it cannot find a recipient's address.

   a) From the dashboard, select **System Functions**.

   b) On the System Functions page, select **Organizations**.

   c) Add a new organization following these guidelines:

   - Fill in all pages of information including General Settings, Email Account Settings, and Advanced Settings.
   - Use an accurate email address in the Reply To Mailbox Account information.

3. Publish the templates.

   a) From the dashboard, select **Templates, Datafields, and Parameters**.

   b) On the Templates tab, filter the templates using “CM_HR”.

   c) For each template that you want to publish, open the template and click **Publish**.

4. Map Time Entry events to a template.

   a) From the dashboard, select **System Functions**.

   b) On the System Functions page, select **Event Mapping** and filter the events using “CM_HR”.

   c) Activate the events by opening each event and editing it.

   d) Assign the Hosted Server organization, change the Event Status to active, and save.

This completes the steps to enable BCM.

5. Enable BCM integration with the Employee Self-Service application and turn on notifications.
   a) Access the Banner administrative Applications Configurations (GUACONF) page.
   b) Set the `ess.timeEntry.useBannerCommunicationManagement` configuration to Y to enable BCM integration in the Employee Self-Service application.
   c) Set the `ess.timeEntry.scheduler.enabled` configuration to Y to enable the scheduler to turn on notifications in Employee Self-Service.
   d) Adjust other notification configurations if you want to change when the system sends notifications. See the Notification configuration on page 72 section for more information.

Results

You have successfully defined and enabled the integration between Employee Self-Service and BCM.

What to do next

For more information about BCM including instructions for setting up and testing the email server, see the *Banner Communication Management Handbook*.

Notification configuration

The notification configurations control whether to use Banner Communication Management to send notifications and when to send them.

Field	Description
ess.timeEntry.useBanner CommunicationManagement	Defines whether or not to enable Banner Communication Management to manage the delivery of notifications. • Y - enable BCM • N - disable BCM You must have Banner Communication Management installed if you enable this configuration.
ess.timeEntry.scheduler. enabled	Defines whether or not to enable the scheduler and turn notifications on or off. • Y - turn on notifications • N - turn off notifications
ess.timeEntry.scheduler. deadlineHours	Defines when to send notifications. The default values are 24 hours and three hours before the pending deadline.

Field	Description
ess.timeEntry.scheduler.cronJobConfig	Defines when to execute the Cron job that schedules notifications. The job identifies pay events that are eligible to send notifications. The job executes daily at midnight by default. This configuration requires a corn expression. Refer to the Quartz web site at http://www.quartz-scheduler.org/documentation/quartz-2.x/tutorials/crontrigger.html for information about corn expression format. **Note:** If you modify the corn expression, you must restart the web server to schedule the notifications based on the new schedule.

Configure integration with General Person application

You must configure the `//Personal Info Application Link.` to enable integration between the Employee Self-Service and General Person applications.

Procedure

1. Change the URL value of the `//Personal Info Application Link.` attribute to point to the Personal Information application on the General Application.

   ```
   banner.SS.personalInfoURL = 'http://<host_name>:<port_number>/
   <GeneralAppName>/ssb/personalInformation#/personalInformationMain'
   ```

2. Enable the General 9.x application flag by setting the following:`banner.SS.personalInfoAppLinkAvailable = 'Y'.`

Generate SAML 2.0 metadata files

This section discusses the creation and handling of metadata files.

The following files must be created:

- keystore file
- service provider file
- identity service provider file

An IDP Certificate entry must be added in the newly created keystore file. Then the keystore, service provider, and identity service provider files must be added to the WAR file creation location.

Create a keystore (*.jks) file

Perform the following steps to create a keystore (`*.jks`) file.

Procedure

1. Create a keystore file (`*.jks`) with the name used in step 2.
   See Create a keystore (*.jks) file on page 87 for more information.
2. Place this file in the location specified in the following key value:

```
"grails.plugin.springsecurity.saml.keyManager.storeFile =
 'classpath:security/employeeselfservice.jks'"
```

Create a service provider file

Perform the following steps to create a service provider file.

Procedure

1. Create a file `banner-<short-appName>-sp.xml` at the folder path mentioned in Set up SAML SSO configuration on page 82.
2. Edit the `banner-<short-appName>-sp.xml` file for the service provider configuration which will configure the Authentication end point and Logout endpoint.
   a) Replace the parameters below with the configured values for the specific application.
      - `<HOSTNAME>`: Application host name
      - `<PORT>: Deployed` Application port number
      - `<ALIAS_NAME>`: Service provider ID set in Ellucian Ethos Identity service provider setup

- `<EXTRACTED_DATA>`: Extract a X509 Certificate key from the keystore for `banner-<short-appName>-sp.xml` (See Extract a X509 Certificate Key on page 87 for more information.)

b) Place the extracted value in the `banner-<short-appName>-sp.xml` file:

`banner-<short-appName>-sp.xml`

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<md:EntityDescriptor xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
 ID="<ALIAS_NAME>" entityID="<ALIAS_NAME>">
<md:SPSSODescriptor AuthnRequestsSigned="false"
 WantAssertionsSigned="false"
 protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">
<md:KeyDescriptor use="signing">
<ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
<ds:X509Data>
<ds:X509Certificate>
<EXTRACTED_DATA>
</ds:X509Certificate>
</ds:X509Data>
</ds:KeyInfo>
</md:KeyDescriptor>
<md:KeyDescriptor use="encryption">
<ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
<ds:X509Data>
<ds:X509Certificate>
<EXTRACTED_DATA>
</ds:X509Certificate>
</ds:X509Data>
</ds:KeyInfo>
</md:KeyDescriptor>
<md:SingleLogoutService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
 Location="http://<HOSTNAME>:<PORT>/<APPLICATION_NAME>/saml/
SingleLogout/alias/<ALIAS_NAME>"/>
<md:SingleLogoutService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect"
 Location="http://<HOSTNAME>:<PORT>/<APPLICATION_NAME>/saml/
SingleLogout/alias/<ALIAS_NAME>"/>
<md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-
format:emailAddress</md:NameIDFormat>
<md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-
format:transient</md:NameIDFormat>
<md:NameIDFormat>urn:oasis:names:tc:SAML:2.0:nameid-
format:persistent</md:NameIDFormat>
<md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-
format:unspecified</md:NameIDFormat>
<md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-
format:X509SubjectName</md:NameIDFormat>
<md:AssertionConsumerService
 Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
 Location="http://<HOSTNAME>:<PORT>/<APPLICATION_NAME>/saml/SSO/
alias/<ALIAS_NAME>" index="0" isDefault="true"/>
<md:AssertionConsumerService
 Binding="urn:oasis:names:tc:SAML:2.0:profiles:holder-
```

```
of-key:SSO:browser" Location="http://<HOSTNAME>:<PORT>/
<APPLICATION_NAME>/saml/SSO/alias/<ALIAS_NAME>"
 hoksso:ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-
Artifact" index="1"
 xmlns:hoksso="urn:oasis:names:tc:SAML:2.0:profiles:holder-of-
key:SSO:browser"/>
<md:AssertionConsumerService
 Binding="urn:oasis:names:tc:SAML:2.0:profiles:holder-
of-key:SSO:browser" Location="http://<HOSTNAME>:<PORT>/
<APPLICATION_NAME>/saml/SSO/alias/<ALIAS_NAME>"
 hoksso:ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-
POST" index="2"
 xmlns:hoksso="urn:oasis:names:tc:SAML:2.0:profiles:holder-of-
key:SSO:browser"/>
</md:SPSSODescriptor>
</md:EntityDescriptor>
```

Create an identity service provider file

Perform the following steps to create an identity service provider file.

Procedure

1. Create a file `banner-<short-appName>-idp.xml` at the folder path mentioned in SAML SSO configuration on page 83.
2. Edit the `banner-<short-appName>-idp.xml` file for the identity provider configuration. The file contains the identity provider information configured in the Ellucian Ethos Identity server over which the configured application sends the SAML request and receives the SAML response.
   a) Replace the parameters below with the configured values for the specific application.
      - `<HOSTNAME>`: Ellucian Ethos Identity server host name
      - `<PORT>`: Deployed Ellucian Ethos Identity server port number
      - `<ALIAS_NAME>`: Service provider ID set in Ellucian Ethos Identity service provider setup
      - `<EXTRACTED_DATA>`: Extract X509 certificate Data for `banner-<short-appName>-idp.xml` (See Extract X509 certificate data on page 89 for more information.)
   b) Place the extracted value in the `banner-<short-appName>-idp.xml` file:

      `banner-<short-appName>-idp.xml`

      Example:

```
<?xml version="1.0"?>

<md:EntityDescriptor
 xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
entityID="https://<HOSTNAME>:<PORT>/samlsso"
 cacheDuration="PT1440M">

<md:IDPSSODescriptor
```

```
protocolSupportEnumeration="urn:oasis:names:tc:SAML:2.0:protocol">

<md:KeyDescriptor use="signing">

<ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">

<ds:X509Data>

<ds:X509Certificate>

<EXTRACTED_DATA>

</ds:X509Certificate>

</ds:X509Data>

</ds:KeyInfo>

</md:KeyDescriptor>

<md:KeyDescriptor use="encryption">

<ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">

<ds:X509Data>

<ds:X509Certificate>

<EXTRACTED_DATA>

</ds:X509Certificate>

</ds:X509Data>

</ds:KeyInfo>

</md:KeyDescriptor>

<md:SingleLogoutService Location="https://<HOSTNAME>:<PORT>/
samlsso"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect"/>

<md:SingleLogoutService Location="https://<HOSTNAME>:<PORT>/
samlsso"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"/>

<md:NameIDFormat>urn:oasis:names:tc:SAML:1.1:nameid-
format:unspecified</
md:NameIDFormat>

<md:SingleSignOnService Location="https://<HOSTNAME>:<PORT>/
samlsso"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"/>

<md:SingleSignOnService Location="https://<HOSTNAME>:<PORT>/
samlsso"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect"/>
```



```
</md:IDPSSODescriptor>

<md:ContactPerson contactType="administrative"/>

</md:EntityDescriptor>
```

Add IDP certificate entry to the .jks file

During SAML configuration, you must add the IDP certificate entry.

About this task

Add the IDP certificate entry to the new `.jks` file you created as part of this task Create a keystore (*.jks) file on page 87.

Procedure

1. Navigate to where the server certificate exists. By default, it is located at:
   `$EIS_HOME\repository\resources\security`
2. Extract X509 certificate Data for `Idp.xml`.
3. Copy `saml-idp.cer` to the `.jks` file by executing the following command:

```
C:\Program Files\Java\jdk1.7.0_67\jre\bin>keytool -import -
trustcacerts
-alias mykey -file saml-idp.cer -keystore employeeselfservice.jks

Enter keystore password: password
Owner: CN=WSO2 IDP, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown,
 C=Unknown
Issuer: CN=WSO2 IDP, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown,
 C=Unknown
Serial number: 12c20870
Valid from: Mon Aug 18 18:37:51 IST 2014 until: Tue Aug 18 18:37:51
 IST 2015
Certificate fingerprints:
MD5:  8B:65:A6:A0:0F:F3:EA:B6:2A:32:37:7A:21:B5:CF:B6
SHA1: C5:73:6C:FD:63:15:45:C4:74:CF:E2:9D:DE:18:9A:4B:F9:6C:9C:5C
SHA256:
 33:47:F1:95:1E:E7:DD:8B:F9:5C:17:A1:88:82:3E:0D:8B:B9:5C:9E:22:10:0B:57:
8F:51:62:9E:FF:1B:38
Signature algorithm name: SHA256withRSA
Version: 3
Extensions:
#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 2F 20 8D 9E F5 BD 36 0C   B9 CC CE D6 69 FD B4 E6
  / ....6.....i...
0010: 64 75 D5 90 du..
]
```

```
]
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

4. To verify the certificate was added, execute the following commands:

```
C:\Program Files\Java\jdk1.7.0_67\jre\bin>keytool -list -v -keystore
employeeeselfservice.jks Enter keystore password: password
Keystore type: JKS
Keystore provider: SUN
Your keystore contains 2 entries
Alias name: mykey
Creation date: Apr 6, 2015
Entry type: trustedCertEntry
Owner: CN=WSO2 IDP, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown,
 C=Unknown
Issuer: CN=WSO2 IDP, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown,
 C=Unknown
Serial number: 12c20870
Valid from: Mon Aug 18 18:37:51 IST 2014 until: Tue Aug 18 18:37:51
 IST 2015
Certificate fingerprints:
MD5:  8B:65:A6:A0:0F:F3:EA:B6:2A:32:37:7A:21:B5:CF:B6
SHA1: C5:73:6C:FD:63:15:45:C4:74:CF:E2:9D:DE:18:9A:4B:F9:6C:9C:5C
SHA256:
 33:47:F1:95:1E:E7:DD:8B:F9:5C:17:A1:88:82:3E:0D:8B:B9:5C:9E:22:10:0B:57:
8F:51:62:9E:FF:1B:38
Signature algorithm name: SHA256withRSA
Version: 3
Extensions:
#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 2F 20 8D 9E F5 BD 36 0C   B9 CC CE D6 69 FD B4 E6
  / ....6.....i...
0010: 64 75 D5 90 du..
]
]
*******************************************
*******************************************

Alias name: mykey
Creation date: Apr 6, 2015
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=John Doe, OU=Ellucian, O=Ellucian, L=Malvern, ST=PA, C=US
Issuer: CN=John Doe, OU=Ellucian, O=Ellucian, L=Malvern, ST=PA, C=US
Serial number: 126891cb
Valid from: Mon Apr 06 16:06:54 IST 2015 until: Thu Mar 31 16:06:54
 IST 2016
Certificate fingerprints:
MD5:  D7:A6:90:A0:7D:19:DF:7E:D9:FF:01:5B:18:1D:FE:71
SHA1: 46:24:3E:A0:1E:65:76:21:D2:93:0F:29:76:60:17:40:07:0C:72:58
SHA256:
 ED:1B:C7:B5:07:49:80:4A:91:93:87:A1:15:9A:20:23:A7:BB:8B:99:89:02:47:
5F:5C:6E:42:47:AA:68:55
```

```
Signature algorithm name: SHA256withRSA
Version: 3
Extensions:
#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 6F 8C FC 5C BA F1 11 FF   E2 58 C8 4F 2F 60 DA 2B  o..
\.....X.O/`.+
0010: A2 EA D8 78                                        ...x
]
]
*******************************************
*******************************************
```

Add keystore, service provider, and identity service provider files to WAR file creation location

Place the three files created in this section into the `EmployeeSelfService\current\instance\config` directory.

After adding the files, the directory should contain the following:

- `employeeselfservice.jks` (The newly created `.jks` file)
- `banner-<short-appName>-idp.xml`
- `banner-<short-appName>-sp.xml`
- `EmployeeSelfService_configuration.groovy`

Set up SAML SSO configuration

The EmployeeSelfService\current\instance\config directory contains the EmployeeSelfService_configuration.groovy file.

This application specific configuration file contains settings that you can customize for your specific environment.

Authentication provider name

The name identifying the application authentication mechanism.

Values are cas or saml. Specify saml to indicate the application will use CAS SSO protocol for authentication as shown in the following example:

```
/*****************************************************
* BANNER AUTHENTICATION PROVIDER CONFIGURATION *
* *
*****************************************************/
//
banner {
sso {
authenticationProvider = 'saml' // Valid values are:
'saml' and 'cas' for SSO. 'default' value to be used only when creating
the release zip file.
authenticationAssertionAttribute = 'UDC_IDENTIFIER'
if(authenticationProvider != 'default') {
grails.plugin.springsecurity.failureHandler.defaultFailureUrl
= '/login/error'
}
}
```

Logout URL

You can specify where a user is directed after logging out of the application by updating the `EmployeeSelfService_configuration.groovy` file.

There are three ways the application can handle logouts. When using SAML, logging out directs the user to a custom logout page.

- Logouts can display the CAS logout page with a redirect URL.
- Logouts can automatically go to a redirect URL (without displaying the CAS logout page).
- Logouts can take the user to a custom logout page with a redirect URL to Home Page.

SAML SSO configuration

This is a sample of the configuration you can enable for SSO between Banner Employee Self-Service and an Identity Management System that support SAML SSO protocol.

> The following properties describe the configurations required for the application to work in SAML SSO protocol.
>
> **Note:** Uncomment this section when SAML SSO is enabled.

Property	Description
`banner.sso.authentication.saml.localLogout`	Default value is set to `false`, indicating that the application will participate in Global logout. An application participating in global logout will notify the Identity Server about logout within the application. If it is set to `true`, indicating local logout, the application will not notify the Identity Server to log the user out from all applications.
`grails.plugin.springsecurity.auth.loginFormUrl`	Pre-populated to provide the Login URL.
`grails.plugin.springsecurity.saml.afterLogoutUrl`	Pre-populated to provide the Logout URL.
`grails.plugin.springsecurity.saml.keyManager.defaultKey`	Key name used at the time of key-store creation (Create a keystore (*.jks) file on page 87). Example: `employeeselfservice.jks`
`grails.plugin.springsecurity.saml.keyManager.storeFile`	Location of the keystore file. • This could be a classpath. Example: `classpath:employeeselfservice.jks` -OR- • It could be an absolute location on the machine. Example 1: file: `c://temp/employeeselfservice.jks` *or*

Property	Description
	Example 2: `u02/employeeselfservice.jks`
`grails.plugin.springsecurity.saml.keyManager.storePass`	Password used to decrypt the keys in the keystore created above.
`grails.plugin.springsecurity.saml.keyManager.passwords`	Key value pair to validate the key. Contains the alias key name used at the time of key-store creation and password used to decrypt the key.
`grails.plugin.springsecurity.saml.metadata.sp.file`	Location of the service provider metadata file. • This could be a classpath location. Example: `security/sp.xml` -OR- • It could be a absolute location on the machine. Example: file: `C://temp/banner-sp file:/home/u02/banner-sp.xml`
`grails.plugin.springsecurity.saml.metadata.providers`	Key value pair mapping to validate the identity provider configured in `banner-<short-appName>-idp.xml`. Example: `adfs : 'security/banner-idp.xml'` Possible keys are adfs, Okta, Shibb, etc.
`grails.plugin.springsecurity.saml.metadata.defaultIdp`	Provide the default IDP to be used from the IDP providers set. This is the same value specified above for the key.
`grails.plugin.springsecurity.saml.metadata.sp.defaults`	• `local`: Pre-populated to indicate value to be picked up. • `alias`: An alias name that is unique to this application. Example: `banner-<application-shortname>-sp` • `securityProfile`: Pre-populated value. • `signingKey`: A key used to sign the messages that is unique to this application.

Property	Description
	Example: `banner-<application-shortname>-sp` • `encryptionKey`: A key to encrypt the message that is unique to this application. Example: `banner-<application-shortname>-sp` • `tlsKey`: A tls key that is unique to this application. Example: `banner-<application-shortname>-sp` • `requireArtifactResolveSigned`: Pre-Populated to set to false indicating artifact to be signed or not. • `requireLogoutRequestSigned`: Pre-Populated to set to false indicating logout request to be signed or not. • `requireLogoutResponseSigned`: Pre-Populated to set to false indicating logout response to be signed or not.

SAML SSO configuration code sample

Uncomment this section if you are using SAML.

```
/*************************************************************
*                                                            *
*                  SAML2 SSO CONFIGURATION                   *
* Uncomment this section if Registration is configured with SAML2 SSO *
*************************************************************/
/*
grails.plugin.springsecurity.saml.active = false
grails.plugin.springsecurity.auth.loginFormUrl = '/saml/login'
grails.plugin.springsecurity.saml.afterLogoutUrl ='/logout/
customLogout'

banner.sso.authentication.saml.localLogout='false'  // To disable
 single logout set this to true.

grails.plugin.springsecurity.saml.keyManager.storeFile =
 'classpath:security/employeeselfservice.jks'  // for unix based
 'file:/home/u02/employeeselfservice.jks'
```

```
grails.plugin.springsecurity.saml.keyManager.storePass = 'changeit'
grails.plugin.springsecurity.saml.keyManager.passwords = [ 'banner-sp':
 'changeit' ]        // banner-sp is the value set in Ellucian Ethos
 Identity Service provider setup
grails.plugin.springsecurity.saml.keyManager.defaultKey = 'banner-sp'
                       // banner-sp is the value set in Ellucian Ethos
 Identity Service provider setup

grails.plugin.springsecurity.saml.metadata.sp.file = 'se-curity/banner-
sp.xml'               // for unix based '/home/u02/banner-sp.xml'
grails.plugin.springsecurity.saml.metadata.providers = [adfs:
 'security//banner-idp.xml']    // for unix based '/home/u02/banner-
idp.xml'
grails.plugin.springsecurity.saml.metadata.defaultIdp = 'adfs'
grails.plugin.springsecurity.saml.metadata.sp.defaults = [
local: true,
alias: 'banner-employeeselfservice-sp',
                                    // banner-employeeselfservice-sp is
 the value set in Ellucian Ethos Identity Service provider setup
securityProfile: 'metaiop',
signingKey: 'banner-employeeselfservice-sp',
                                    // banner-employeeselfservice-sp is
 the value set in Ellucian Ethos Identity Service provider setup
encryptionKey: 'banner-employeeselfservice-sp',
                                    // banner-employeeselfservice-sp is
 the value set in Ellucian Ethos Identity Service provider setup
tlsKey: 'banner-employeeselfservice-sp',
                                    // banner-employeeselfservice-sp is
 the value set in Ellucian Ethos Identity Service provider setup
requireArtifactResolveSigned: false,
requireLogoutRequestSigned: false,
requireLogoutResponseSigned: false
]
*/
```

Note: After performing the above configuration changes, rebuild and redeploy the WAR file to the corresponding web application servers. Refer to the following sections for instructions:

- Regenerate the WAR file on page 47.
- Configure the web application server and deploy the WAR file on page 48.

SAML 2.0 Configuration Sub-tasks

You need to perform few tasks to complete the SAML configuration.

Note: Few Banner applications support SAML configuration while some of them do not support SAML. Employee Self-Service does not support SAML configuration.

Create a keystore (*.jks) file

Perform the following steps to create a keystore (`*.jks`) file.

Procedure

1. Create a keystore file (`*.jks`) with the name used in step 2.
2. Place this file in the location specified in the following key value:

```
"grails.plugin.springsecurity.saml.keyManager.storeFile =
 'classpath:security/employeeselfservice.jks'"
```

Extract a X509 Certificate Key

During SAML configuration, you must extract a X509 certificate key from the keystore for the `banner-<Application_Name>-sp.xml` file.

Procedure

1. With the `employeeselfservice.jks` file you created, execute the command below to check which certificates are in a Java keystone.

   See “Create a keystore (*.jks) file” on page 90 for more information.

```
C:\Program Files\Java\jdk1.7.0_67\jre\bin>keytool -list -v -keystore
 employeeselfservice.jks
Enter keystore password:
Keystore type: JKS
Keystore provider: SUN
Your keystore contains 1 entry
Alias name: mykey
Creation date: Apr 6, 2015
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=John Doe, OU=Ellucian, O=Ellucian, L=Malvern, ST=PA, C=US
Issuer: CN=John Doe, OU=Ellucian, O=Ellucian, L=Malvern, ST=PA, C=US
Serial number: 78548b7
```

```
Valid from: Mon Apr 06 12:52:39 IST 2015 until: Thu Mar 31 12:52:39
 IST 2016
Certificate fingerprints:
MD5:  5D:55:F4:18:3D:CF:AE:5A:27:B8:85:68:42:47:CA:76
SHA1: CD:09:04:F5:01:60:14:CC:DF:48:07:4A:93:99:17:BF:10:83:F3:55
SHA256:
 79:5A:7F:0C:A4:B1:0E:30:9C:B0:DD:87:2C:CA:19:A1:0E:89:29:2F:95:A1:35:E9:EC:A2:AA:
Signature algorithm name: SHA256withRSA
Version: 3
Extensions:
#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: B2 AC D7 09 01 15 22 A3   32 08 86 64 E8 25 5A 15
  ......".2..d.%Z.
0010: CB A0 C6 D9                                        ....
]
]
*******************************************
*******************************************
```

This command shows all the available keystores in the `.jks` file.

Note: To get information about the specific certificate, execute this command:

```
keytool -list -v -keystore application_name.jks -alias mykey
```

2. Execute the following command to export a certificate from a keystore:

```
C:\Program Files\Java\jdk1.7.0_67\jre\bin>keytool -export -alias
 mykey -file mykey.crt -keystore application_name.jks
Enter keystore password: password
Certificate stored in file <mykey.crt>
```

3. Execute the following command to get the X509 certificate:

```
C:\Program Files\Java\jdk1.7.0_67\jre\bin>keytool -printcert -rfc -
file mykey.crt
-----BEGIN CERTIFICATE-----
MIIDfTCCAmWgAwIBAgIEB4VItzANBgkqhkiG9w0BAQsFADBvMQswCQYDVQQGEwJJTjELMAkGA1UE
CBMCSU4xEjAQBgNVBAcTCUJhbmdhbG9yZTERMA8GA1UEChMIRWxsdWNpYW4xETAPBgNVBAsTCEVs
bHVjaWFuMRkwFwYDVQQDExBTcGhvb3J0aSBBY2hhcnlhMB4XDTE1MDQwNjA3MjIzOVoXDTE2MDMz
MTA3MjIzOVowbzELMAkGA1UEBhMCSU4xCzAJBgNVBAgTAklOMRIwEAYDVQQHEwlCYW5nYWxvcmUx
ETAPBgNVBAoTCEVsbHVjaWFuMREwDwYDVQQLEwhFbGx1Y2lhbjEZMBcGA1UEAxMQU3Bob29ydGkg
QWNoYXJ5YTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAN2cS
+2OX37HioFzwOWLm/S0
F+zt6ldtLfmHc16V9iqkZChkMiXpKgXVPqGLFjBrhwfsWtuMfRy2NYf3forEDFTaV4/
fLXRo+Npd
xTfqWhuZTafDJEyKQc57KY8G3feg1CSjfKkCk1LF
+zbGClHQ0bg1dwUjJlp7eKjWM0rbsKMd5pZ7
0tGAPcYsi6MtGvJupaVhy3jNTDg+kh4/D92y/mTaLlCR4QQr1qlU9+H
+it3m9jiDrZ7svrdBlsDN
1BVcXDooqUGTuc10IBxYEsb7hFucSFpdJnGJvbg35ll9K9F5S8lEmiQmeOUQ1gQe2Ow01kFl56Qz
4evM0xgeskNid9sCAwEAAaMhMB8wHQYDVR0OBBYEFLKs1wkBFSKjMgiGZOglWhXLoMbZMA0GCSqG
SIb3DQEBCwUAA4IBAQANJbYRTcMwhrETz+mo
+n1okrXIs118AIm7s1yJJd1nyJuaKrn7DcPPLzy/
```

```
RjHGKP02uLiupgqar+UUaPqSZjJXSzktLLyq7H6DRrW0Jp2rw48a+Kou
+XOvQ8ZWR9ZXIa1XoAoD
PaSSE2omcVOVGZmQKUYardVeSvQth3IVMW9w9Jl
+DuavXavVjIx5IN6RRhXGfaJjQLKFzIDqZNAp
OcxMEKXHOUqj0ksTRARLpKWSPu7gFOWO/6qapNp5l8rlPjnVxDhqHqCKC3E40VI5n+C
+KJHZQqab
Tfhd6erqEy7S1Cazr655Yq22Jm6L7IXsXgpRwmZnoietLsrFIRyPe1DY
-----END CERTIFICATE-----
```

Extract X509 certificate data

During SAML configuration, you must extract X509 certificate data for `banner-<Application_Name>-sp.xml`.

Procedure

1. Go to the deployed WSO2 server / Ellucian Ethos Identity server. For example, `C:\>cd C:\work\eis\`.
2. Navigate to the server certificate location. By default, it is located at:

   `$EIS_HOME\repository\resources\security`
3. Execute the following command, where `saml-idp.cer` is the certificate file in the server:

   ```
   C:\work\eis\repository\resources\security>keytool -printcert -rfc -
   file saml-idp.cer
   ```

   This would return a value similar to the following:

   ```
   -----BEGIN CERTIFICATE-----
   MIICdDCCAd2gAwIBAgIEEsIIcDANBgkqhkiG9w0BAQsFADBtMRAwDgYDVQQGEwdVbmtub3duMRAw
   DgYDVQQIEwdVbmtub3duMRAwDgYDVQQHEwdVbmtub3duMRAwDgYDVQQKEwdVbmtub3duMRAwDgYD
   VQQLEwdVbmtub3duMREwDwYDVQQDEwhXU08yIElEUDAeFw0xNDA4MTgxMzA3NTFaFw0xNTA4MTgx
   MzA3NTFaMG0xEDAOBgNVBAYTB1Vua25vd24xEDAOBgNVBAgTB1Vua25vd24xEDAOBgNVBAcTB1Vu
   a25vd24xEDAOBgNVBAoTB1Vua25vd24xEDAOBgNVBAsTB1Vua25vd24xETAPBgNVBAMTCFdTTzIg
   SURQMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC44MMcoAPb+aR8l5gtTsSb
   +SzslHFESmKL
   wO1+SAY6p2iiO
   +G9qR/511ufCzKWrMdMMpoCJLC0myDwoUuGvk0dycTpm5NwUX6CnqDtYhtGkYg8
   JT
   +LtG67k6yjXNa9wrE6VBJynDDPnlL8gLUl9ZCIFrmevJ75rOCaLsoFsghHPwIDAQABoyEwHzAd
   BgNVHQ4EFgQULyCNnvW9Ngy5zM7Waf205mR11ZAwDQYJKoZIhvcNAQELBQADgYEApWlSy2GUSaHM
   Kkc8XZmdQ0//SId8DKRKaFaZW388K4dJGTSWUnzq4iCWFrAN9O4D1DBnNE
   +dCDEmV8HvmyQBedsG
   JnAre0VisqKz9CjIELcGUaEABKwkOgLe1YyqV29vS4Y3PuTxAhbkyphFb5PxjHDHH/
   WLQ8pOTbsC
   vX4wO04=
   -----END CERTIFICATE-----
   ```

   a) If no certificate file is found, navigate to the `.jks` file. The `.jks` file can be found through `carbon.xml`. By default, it is located at:

      `$EIS_HOME\repository\conf\carbon.xml`

b) Locate the KeyStore file location in the `carbon.xml` file, as shown in the following example:

```
<KeyStore>
    <!-- Keystore file location-->
    <Location>${carbon.home}/repository/resources/security/
cacerts</Location>
    <!-- Keystore type (JKS/PKCS12 etc.)-->
    <Type>JKS</Type>
    <!-- Keystore password-->
    <Password>changeit</Password>
    <!-- Private Key alias-->
    <KeyAlias>mykey</KeyAlias>
    <!-- Private Key password-->
    <KeyPassword>changeit</KeyPassword>
</KeyStore>
```

c) Create the `saml-idp.cer` file by executing the following command.

```
keytool -export -keystore cacerts -alias mykey -file ~/saml-idp.cer
```

4. Go to the keystore location and execute the following command.

Note: The `password` and `alias` referenced in this example are also contained in the `carbon.xml` file accessed earlier in this task.

```
C:\work\eis\repository\resources\security>keytool -export -keystore
 cacerts -rfc -alias mykey
Enter keystore password:
-----BEGIN CERTIFICATE-----
MIICdDCCAd2gAwIBAgIEEsIIcDANBgkqhkiG9w0BAQsFADBtMRAwDgYDVQQGEwdVbmtub3duMRAw
DgYDVQQIEwdVbmtub3duMRAwDgYDVQQHEwdVbmtub3duMRAwDgYDVQQKEwdVbmtub3duMRAwDgYD
VQQLEwdVbmtub3duMREwDwYDVQQDEwhXU08yIElEUDAeFw0xNDA4MTgxMzA3NTFaFw0xNTA4MTgx
MzA3NTFaMG0xEDAOBgNVBAYTB1Vua25vd24xEDAOBgNVBAgTB1Vua25vd24xEDAOBgNVBAcTB1Vu
a25vd24xEDAOBgNVBAoTB1Vua25vd24xEDAOBgNVBAsTB1Vua25vd24xETAPBgNVBAMTCFdTTzIg
SURQMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQC44MMcoAPb+aR8l5gtTsSb
+SzslHFESmKL
wO1+SAY6p2iiO
+G9qR/511ufCzKWrMdMMpoCJLC0myDwoUuGvk0dycTpm5NwUX6CnqDtYhtGkYg8
JT
+LtG67k6yjXNa9wrE6VBJynDDPnlL8gLUl9ZCIFrmevJ75rOCaLsoFsghHPwIDAQABoyEwHzAd
BgNVHQ4EFgQULyCNnvW9Ngy5zM7Waf205mR11ZAwDQYJKoZIhvcNAQELBQADgYEApWlSy2GUSaHM
Kkc8XZmdQ0//SId8DKRKaFaZW388K4dJGTSWUnzq4iCWFrAN9O4D1DBnNE
+dCDEmV8HvmyQBedsG
JnAre0VisqKz9CjIELcGUaEABKwkOgLe1YyqV29vS4Y3PuTxAhbkyphFb5PxjHDHH/
WLQ8pOTbsC
vX4wO04=
-----END CERTIFICATE-----
```

Add IDP certificate entry to the .jks file

During SAML configuration, you must add the IDP certificate entry.

About this task

Add the IDP certificate entry to the new `.jks` file you created as part of this task Create a keystore (*.jks) file on page 87.

Procedure

1. Navigate to where the server certificate exists. By default, it is located at:
   `$EIS_HOME\repository\resources\security`
2. Extract X509 certificate Data for `Idp.xml`.
3. Copy `saml-idp.cer` to the `.jks` file by executing the following command:

```
C:\Program Files\Java\jdk1.7.0_67\jre\bin>keytool -import -
trustcacerts
-alias mykey -file saml-idp.cer -keystore employeeselfservice.jks

Enter keystore password: password
Owner: CN=WSO2 IDP, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown,
 C=Unknown
Issuer: CN=WSO2 IDP, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown,
 C=Unknown
Serial number: 12c20870
Valid from: Mon Aug 18 18:37:51 IST 2014 until: Tue Aug 18 18:37:51
 IST 2015
Certificate fingerprints:
MD5:  8B:65:A6:A0:0F:F3:EA:B6:2A:32:37:7A:21:B5:CF:B6
SHA1: C5:73:6C:FD:63:15:45:C4:74:CF:E2:9D:DE:18:9A:4B:F9:6C:9C:5C
SHA256:
 33:47:F1:95:1E:E7:DD:8B:F9:5C:17:A1:88:82:3E:0D:8B:B9:5C:9E:22:10:0B:57:
8F:51:62:9E:FF:1B:38
Signature algorithm name: SHA256withRSA
Version: 3
Extensions:
#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 2F 20 8D 9E F5 BD 36 0C   B9 CC CE D6 69 FD B4 E6
  / ....6.....i...
0010: 64 75 D5 90 du..
]
]
Trust this certificate? [no]:  yes
Certificate was added to keystore
```

4. To verify the certificate was added, execute the following commands:

```
C:\Program Files\Java\jdk1.7.0_67\jre\bin>keytool -list -v -keystore
employeeselfservice.jks Enter keystore password: password
```

```
Keystore type: JKS
Keystore provider: SUN
Your keystore contains 2 entries
Alias name: mykey
Creation date: Apr 6, 2015
Entry type: trustedCertEntry
Owner: CN=WSO2 IDP, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown,
 C=Unknown
Issuer: CN=WSO2 IDP, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown,
 C=Unknown
Serial number: 12c20870
Valid from: Mon Aug 18 18:37:51 IST 2014 until: Tue Aug 18 18:37:51
 IST 2015
Certificate fingerprints:
MD5:  8B:65:A6:A0:0F:F3:EA:B6:2A:32:37:7A:21:B5:CF:B6
SHA1: C5:73:6C:FD:63:15:45:C4:74:CF:E2:9D:DE:18:9A:4B:F9:6C:9C:5C
SHA256:
 33:47:F1:95:1E:E7:DD:8B:F9:5C:17:A1:88:82:3E:0D:8B:B9:5C:9E:22:10:0B:57:
8F:51:62:9E:FF:1B:38
Signature algorithm name: SHA256withRSA
Version: 3
Extensions:
#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 2F 20 8D 9E F5 BD 36 0C   B9 CC CE D6 69 FD B4 E6
  / ....6.....i...
0010: 64 75 D5 90 du..
]
]
*******************************************
*******************************************

Alias name: mykey
Creation date: Apr 6, 2015
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=John Doe, OU=Ellucian, O=Ellucian, L=Malvern, ST=PA, C=US
Issuer: CN=John Doe, OU=Ellucian, O=Ellucian, L=Malvern, ST=PA, C=US
Serial number: 126891cb
Valid from: Mon Apr 06 16:06:54 IST 2015 until: Thu Mar 31 16:06:54
 IST 2016
Certificate fingerprints:
MD5:  D7:A6:90:A0:7D:19:DF:7E:D9:FF:01:5B:18:1D:FE:71
SHA1: 46:24:3E:A0:1E:65:76:21:D2:93:0F:29:76:60:17:40:07:0C:72:58
SHA256:
 ED:1B:C7:B5:07:49:80:4A:91:93:87:A1:15:9A:20:23:A7:BB:8B:99:89:02:47:
5F:5C:6E:42:47:AA:68:55
Signature algorithm name: SHA256withRSA
Version: 3
Extensions:
#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 6F 8C FC 5C BA F1 11 FF   E2 58 C8 4F 2F 60 DA 2B  o..
\.....X.O/`.+
```

```
0010: A2 EA D8 78                                               ...x
]
]
*******************************************
*******************************************
```

Add service provider in Ellucian Ethos Identity server

You need to add an Ellucian Ethos Identity server as part of the SAML configuration.

Procedure

1. Log in to the Ellucian Ethos Identity Management Console.
2. Click the **Main** tab.
3. Under **Service Providers**, click **Add**.
4. On the **Add Service Provider** screen, enter a service provider name in the **Service Provider Name** field.
   You can also add an optional description in the **Description** field.
5. Click **Register** to add the service provider.

Add SAML settings for Ellucian Ethos Identity server

Add the SAML settings for the integrating application.

Procedure

1. On the **Service Providers** screen, expand the **Inbound Authentication Configuration** panel.
2. Expand the **SAML2 Web SSO Configuration** panel.
3. Click the **Configure** link.
4. Enter the **Issuer** value that is configured in the service provider application.
   This value is validated against the SAML Authentication Request issued by the service provider.

   Note: Make sure to provide the same value that is configured as the "alias" in the configuration property `'grails.plugin.springsecurity.saml.metadata.sp.defaults'` in the `EmployeeSelfService_configuration.groovy` file.
5. Enter a valid **Assertion Consumer URL** where the browser redirects the SAML Response after authentication.
6. Enter a valid **NameID** format supported by Ellucian Ethos Identity.
   The following values can be used:
   - urn:oasis:names:tc:SAML:2.0:nameid-format:persistent
   - urn:oasis:names:tc:SAML:2.0:nameid-format:transient
   - urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress
   - urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified
   - urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName

- urn:oasis:names:tc:SAML:1.1:nameid-format:WindowsDomainQualifiedName
- urn:oasis:names:tc:SAML:2.0:nameid-format:kerberos
- urn:oasis:names:tc:SAML:2.0:nameid-format:entity

7. Select **Use fully qualified username** in the NameID.

   In most cases, this can be left unselected.

   This lets you preserve the user store domain and user ID in the SAML 2.0 Response.
8. Select **Enable Response Signing** to sign the SAML 2.0 Responses returned after the authentication process.
9. Select **Enable Assertion Signing** to sign the SAML 2.0 Assertions returned after the authentication.
10. Select **Enable Signature Validation in Authentication Requests and Logout Requests**.

    This ensures that the identity provider validates the signature of the SAML 2.0 Authentication and Logout Requests that are sent by the service provider.
11. Select **Assertion Encryption** to encrypt the assertions.
12. Select **Certificate Alias** for the service provider's public certificate.

    This certificate is used to validate the signature of SAML 2.0 Requests and is used to generate encryption.
13. Select **Enable Single Logout**.

    If the service provider supports a different URL than the Assertion Consumer URL for logout, enter a `Custom Logout URL` for logging out. This should match the URL set up in the .xml file property "SingleLogoutService".

    This terminates all sessions across all authenticated service providers when the user signs out from one server.
14. Select **Enable Attribute Profile**.

    This adds a basic attribute profile where the identity provider can include the user's attributes in the SAML Assertions as part of the attribute statement.
15. Select **Include Attributes in the Response Always** if the identity provider should always include the attribute values related to the selected claims in the SAML attribute statement.

    This is required so that UDC_IDENTIFIER configured in claims are sent across.
16. Select **Enable Audience Restriction** to restrict the audience.
17. Add audience members using the **Audience** text box.
18. Click **Add Audience**.
19. Select **Enable IdP Initiated SSO**.

    This will not require the service provider to send the SAML 2.0 Request.
20. Click **Register**.

    This will save your settings and return you to the **Service Provider Configuration** page.
21. Click **Update** to save all settings.

Modify identity provider issuer

Add a resident identity provider as per the `idp_local.xml` configuration.

About this task

The Ellucian Ethos Identity Server can mediate authentication requests between service providers and identity providers. At the same time, the identity server can act as a service provider and an identity provider.

When acting as an identity provider, it is known as the resident identity provider. This converts the identity server into a federated hub. The resident identity provider configuration is relevant for you if you are a service provider and want to send an authentication request or a provisioning request to the identity server (for example, through SAML, OpenID, OpenID Connect, SCIM, and WS-Trust).

Resident identity provider configuration is a one-time configuration for a given tenant. It shows you the identity server's metadata, like the endpoints. In addition, you can secure the WS-Trust endpoint with a security policy. You must change the Identity Provider Entity Id to the expected URL of the Issuer statement in SAML 2.0 Responses.

Complete the following steps to change the ID:

Procedure

1. Log in to the Ellucian Ethos Identity Management Console.
2. Click the **Main** tab.
3. Under the Identity section, in the Main menu, click **List** under Identity Providers.
4. Enter a service provider name in the **Service Provider Name** field.
   You can also add an optional description in the Description field.
5. Click **List** under Identity Providers.
6. Click **Resident Identity Provider**.
7. Expand the **Inbound Authentication Configuration** panel.
8. Expand the **SAML2 Web SSO Configuration** panel.
9. Enter a valid identity provider name or URL to be used by all service providers.
10. Click **Update** to save the settings.