OutSystems® Platform 9
The Detach Process for Java

This document is a step-by-step guide for extracting the source code of your applications from the Java version of the OutSystems® Platform and how to set it up to be executed and maintained independently of the OutSystems Platform, if you ever decide to detach from it.
This is a complement to the OutSystems Platform – Standard Architecture with No Lock-in technical note that gives a high-level view on how the OutSystems Platform, unlike other proprietary technologies and frameworks, generates standard, optimized, and fully documented Java source code that does not require runtime interpreters or engines.

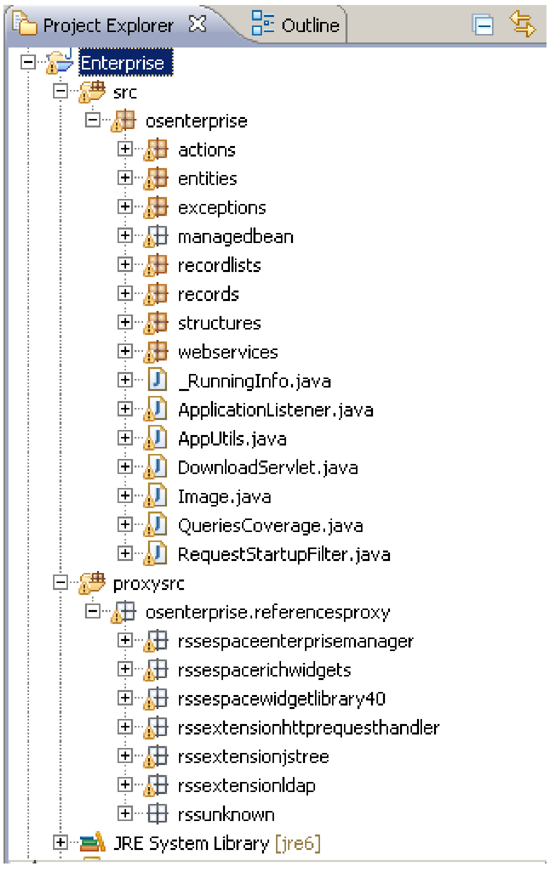
Table of Contents

1 How the OutSystems Platform Structures the Generated Code

When you deploy an application module with the OutSystems Platform through the 1-Click Publish operation, a standard JEE application is generated by the OutSystems Compiler with the output code structured as described below.

The main folders are:

- **src** – the underlying core source code of your application.
- **jsp –** the source code for Web Screens and Web Blocks of your application, and all of the supporting JavaScript files, images and Style Sheets.
- **proxysrc** - the proxy code for referencing other applications deployed with the Platform.



```
Enterprise
  src
    osenterprise
      actions
      entities
      exceptions
      managedbean
      recordlists
      records
      structures
      webservices
      _RunningInfo.java
      ApplicationListener.java
      AppUtils.java
      DownloadServlet.java
      Image.java
      QueriesCoverage.java
      RequestStartupFilter.java
  proxysrc
    osenterprise.referencesproxy
      rssespaceenterprisemanager
      rssespacerichwidgets
      rssespacewidgetlibrary40
      rssextensionhttprequesthandler
      rssextensionjstree
      rssextensionldap
      rssunknown
  JRE System Library [jre6]
```

Figure 1: Code Structure of a generated Application.

In the **src** folder there is a set of packages that hold the different types of code. These are:

- **actions** – the code generated for both built-in actions and functions, and user actions and functions;
- **managedbean** – the code generated for web screens, grouped by web flows;
- **webservices** – the code generated for web services exposed by your application;
- **webreferences** – the proxy code to support the execution of external web services;
- **entities** – the code to manage entities;
- **structures** – the code to manage structures;
- **records** – the code to manage records defined in the development environment;
- **recordList** – the code to manage record lists;

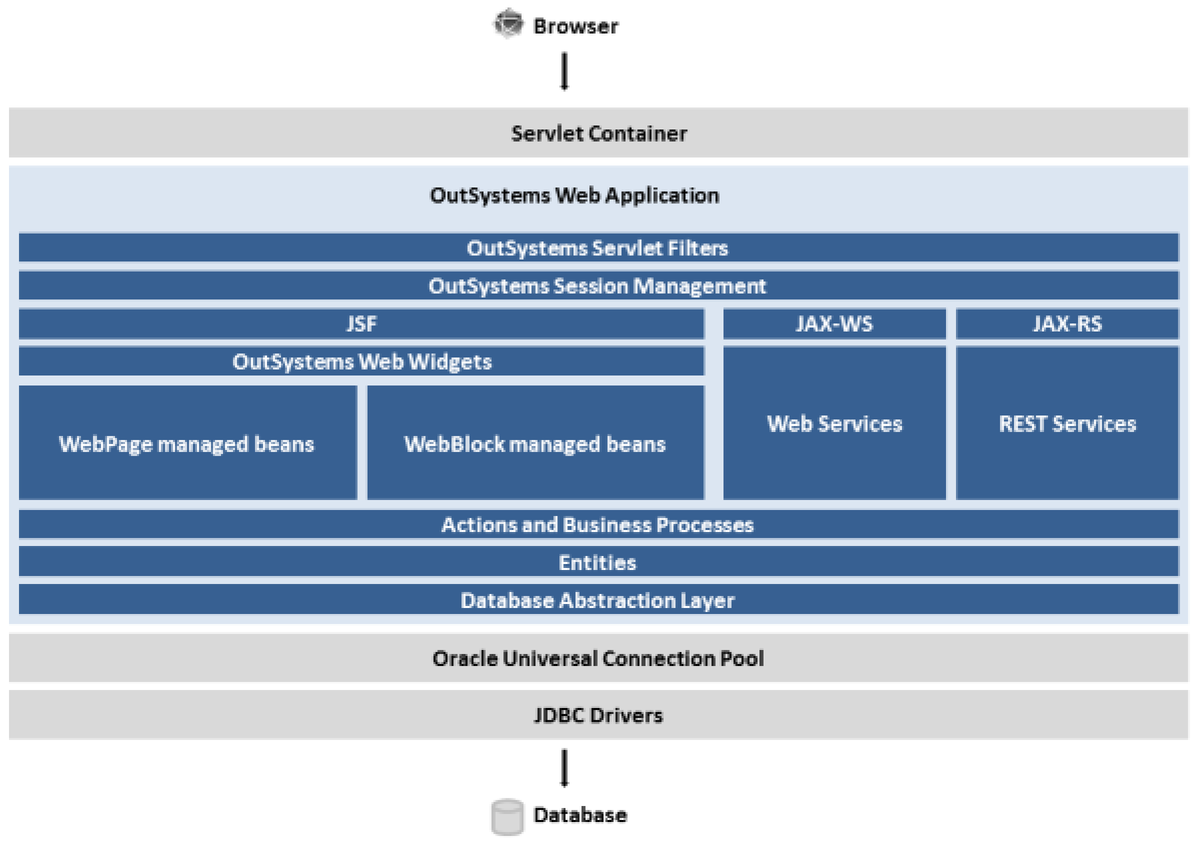
- **exceptions** – the declaration of all user exceptions;
- **processes** – the code to manage BPT processes and process activities;
- **timers** – the code to manage timers;
- **themes** – the definitions of the web screen themes.

Other important files can be found at:

- **copyToApp** – external resources of the application, such as: images, Excel files;
- **WEB-INF** – the definition of both exposed and consumed web services (WSDL files), JavaServer Faces configuration file, `web.xml` deployment descriptor file, and additional Application Server deployment configuration files;
- **build.xml** – the `ant` build file used to generate the `.war` file that is deployed to the Application Server. This file also contains useful targets to build your application and to generate Javadoc documentation.

2 What is the Architecture of the Detached Application

The architecture of an application developed with the OutSystems Platform is shown below:


Browser
Servlet Container
OutSystems Web Application
OutSystems Servlet Filters
OutSystems Session Management
JSF
JAX-WS
JAX-RS
OutSystems Web Widgets
WebPage managed beans
WebBlock managed beans
Web Services
REST Services
Actions and Business Processes
Entities
Database Abstraction Layer
Oracle Universal Connection Pool
JDBC Drivers
Database

Contrary to other technologies, the OutSystems Platform doesn't use a proprietary runtime engine: all applications rely only on standard Java technology:

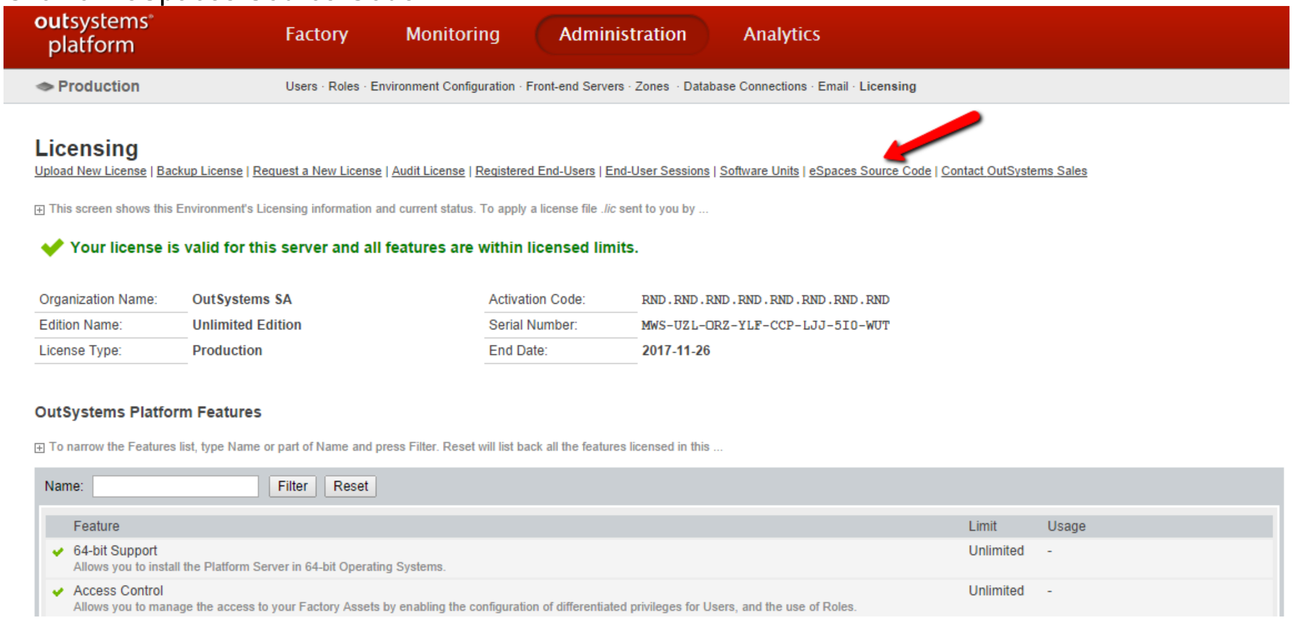
JSF 1.2 - for the presentation layer;
JAX-WS – for web services;
JAX-RS – for REST services;
JMS – queues for high-performance logging;
Oracle Universal Connection Pool – to manage the database access;
JSR 88 – for the application deployment, using Application Server extensions for improved performance.

3 Detaching the Source Code

Do you want to evaluate the detach source code process?
For the purpose of evaluating the detach source code process, OutSystems provides an embedded tutorial in the development environment to guide you through the process.
The steps described in this section are not enabled for evaluation purposes and will only be fully available if you decide to stop using the OutSystems Platform.
Please refer to the first chapter of the OutSystems Platform – Standard Architecture with No Lock-in technical note for more information.

To obtain the source code of an application module, open the Licensing page under the Administration tab in environment management console and detach the source code as follows:

1 Click on 'eSpaces Source Code'.


outsystems platform
Factory
Monitoring
Administration
Analytics
Production
Users · Roles · Environment Configuration · Front-end Servers · Zones · Database Connections · Email · Licensing

Licensing

Upload New License | Backup License | Request a New License | Audit License | Registered End-Users | End-User Sessions | Software Units | eSpaces Source Code | Contact OutSystems Sales

This screen shows this Environment's Licensing information and current status. To apply a license file .lic sent to you by ...

✓ **Your license is valid for this server and all features are within licensed limits.**

Organization Name:	**OutSystems SA**	Activation Code:	RND.RND.RND.RND.RND.RND.RND.RND
Edition Name:	**Unlimited Edition**	Serial Number:	MWS-UZL-ORZ-YLF-CCP-LJJ-5I0-WUT
License Type:	**Production**	End Date:	**2017-11-26**

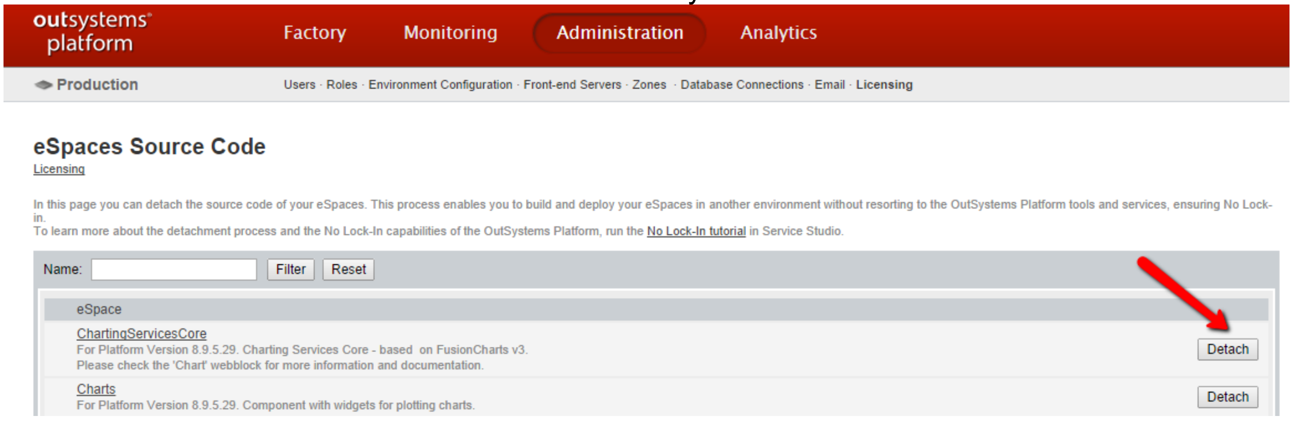
OutSystems Platform Features

To narrow the Features list, type Name or part of Name and press Filter. Reset will list back all the features licensed in this ...

Name: Filter Reset

	Feature	Limit	Usage
✓	64-bit Support Allows you to install the Platform Server in 64-bit Operating Systems.	Unlimited	-
✓	Access Control Allows you to manage the access to your Factory Assets by enabling the configuration of differentiated privileges for Users, and the use of Roles.	Unlimited	-

2 Click on 'Detach' for the module whose source code you want to obtain.


outsystems platform
Factory
Monitoring
Administration
Analytics
Production
Users · Roles · Environment Configuration · Front-end Servers · Zones · Database Connections · Email · Licensing

eSpaces Source Code

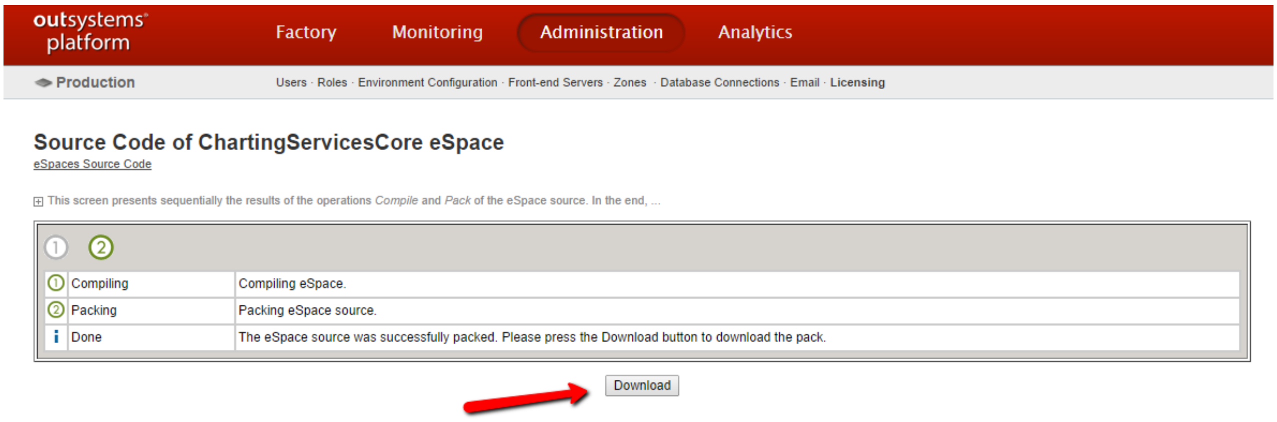
Licensing

In this page you can detach the source code of your eSpaces. This process enables you to build and deploy your eSpaces in another environment without resorting to the OutSystems Platform tools and services, ensuring No Lock-in.
To learn more about the detachment process and the No Lock-In capabilities of the OutSystems Platform, run the No Lock-In tutorial in Service Studio.

Name: Filter Reset

eSpace	
ChartingServicesCore For Platform Version 8.9.5.29. Charting Services Core - based on FusionCharts v3. Please check the 'Chart' webblock for more information and documentation.	Detach
Charts For Platform Version 8.9.5.29. Component with widgets for plotting charts.	Detach

3 Wait until the OutSystems Platform has finished packing the module source code and click on ‘Download’.


outsystems platform
Factory
Monitoring
Administration
Analytics
Production
Users · Roles · Environment Configuration · Front-end Servers · Zones · Database Connections · Email · Licensing
Download

Source Code of ChartingServicesCore eSpace

eSpaces Source Code

This screen presents sequentially the results of the operations *Compile* and *Pack* of the eSpace source. In the end, ...

Compiling	Compiling eSpace.
Packing	Packing eSpace source.
Done	The eSpace source was successfully packed. Please press the Download button to download the pack.

4 Save the zip file and extract it.

4 Database Configuration

Your application data will be preserved, either in the same databases/schemas and database server, or by moving the databases/schemas from the current database server to a new one. However, in order for your application to continue working, even if you do not move the databases/schemas to a new server, you need to manually reconfigure your database connection strings.
These items are located in the `/etc/.java/.systemPrefs/outsystems/prefs.xml` file.
To change the database connection strings, you will need to locate all keys with the "OutSystems.HubEdition" prefix and change the `value` attribute for the ones that represent connection strings.
There are 7 connection strings that need to be changed:

- OutSystems.DB.Application.Log.ConnectionString
- OutSystems.DB.Application.Runtime.ConnectionString
- OutSystems.DB.Application.Session.ConnectionString
- OutSystems.DB.Services.Admin.ConnectionString
- OutSystems.DB.Services.Log.ConnectionString
- OutSystems.DB.Services.Runtime.ConnectionString
- OutSystems.DB.Services.Session.ConnectionString

You will not be able to use any of these connection strings while they are encrypted, so you need to convert them to their plain text format:

1. MySQL:
   jdbc:mysql://SERVER/SCHEMA?user=USER&amp;password=PASSWORD&amp;allowMultiQueries=true&amp;tinyInt1isBit=false&amp;noAccessToProcedureBodies=true
2. Oracle:
   jdbc:oracle:thin:USER/PASSWORD@//SERVER:PORT/SERVICENAME

For each of those connection strings, delete the contents of the `value` attribute, copy the plain text format from the line above and replace the USER, PASSWORD, SERVER, PORT and SERVICENAME parts with the correct values for your installation.
The following table provides a match between the connection string and its user.

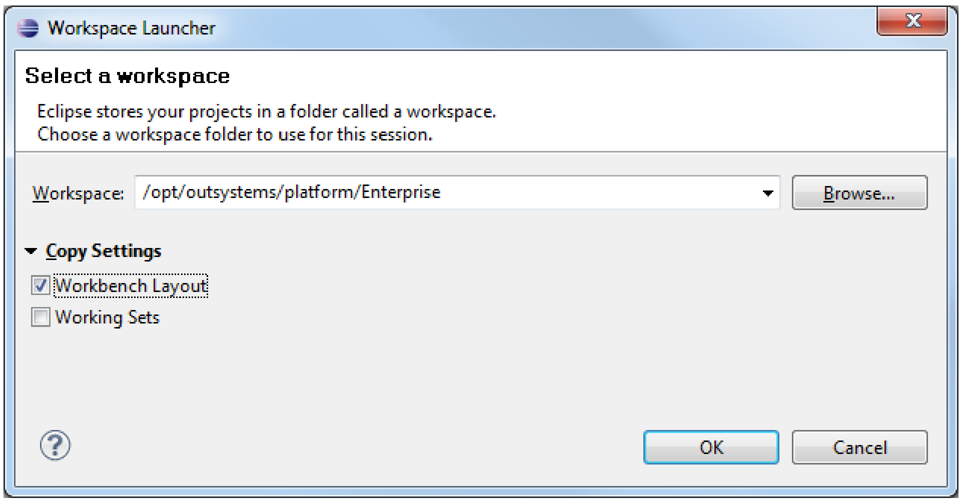
Connection	***catalog***
OutSystems.DB.Application.Log.ConnectionString	*OutSystems*
OutSystems.DB.Application.Runtime.ConnectionString	*OutSystems*
OutSystems.DB.Application.Session.ConnectionString	*ASPState*
OutSystems.DB.Services.Admin.ConnectionString	*OutSystems*
OutSystems.DB.Services.Log.ConnectionString	*OutSystems*
OutSystems.DB.Services.Runtime.ConnectionString	*OutSystems*
OutSystems.DB.Services.Session.ConnectionString	*ASPState*

After changing the connection strings, you will need to restart your application server.

5 Configuring the Eclipse Workspace

5.1 Creating Your Base Workspace

To have all the Java source code in the same Eclipse workspace, we recommend you to set up a new empty workspace: select the *File->Switch Workspace->Other…* option in Eclipse and choose an empty folder in a place of your choice (make sure you do not use the same folder where you saved the modules to be imported). Finally, check 'Workbench Layout'.



Workspace Launcher

Select a workspace

Eclipse stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: /opt/outsystems/platform/Enterprise

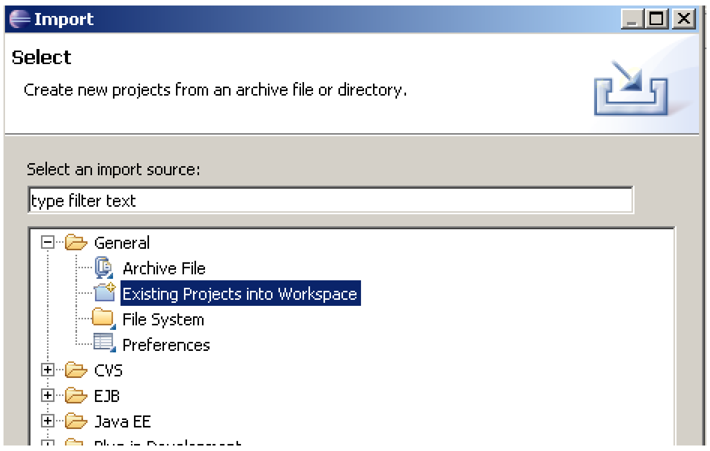
Copy Settings

- [x] Workbench Layout
- [ ] Working Sets

5.2 Importing the Application Generated Code into Eclipse

To import the application code into your newly created workspace follow the steps below:

1. In Eclipse's menu, select the *File->Import…* option and then the *Existing Projects into Workspace* import source inside the *General* folder.



Import

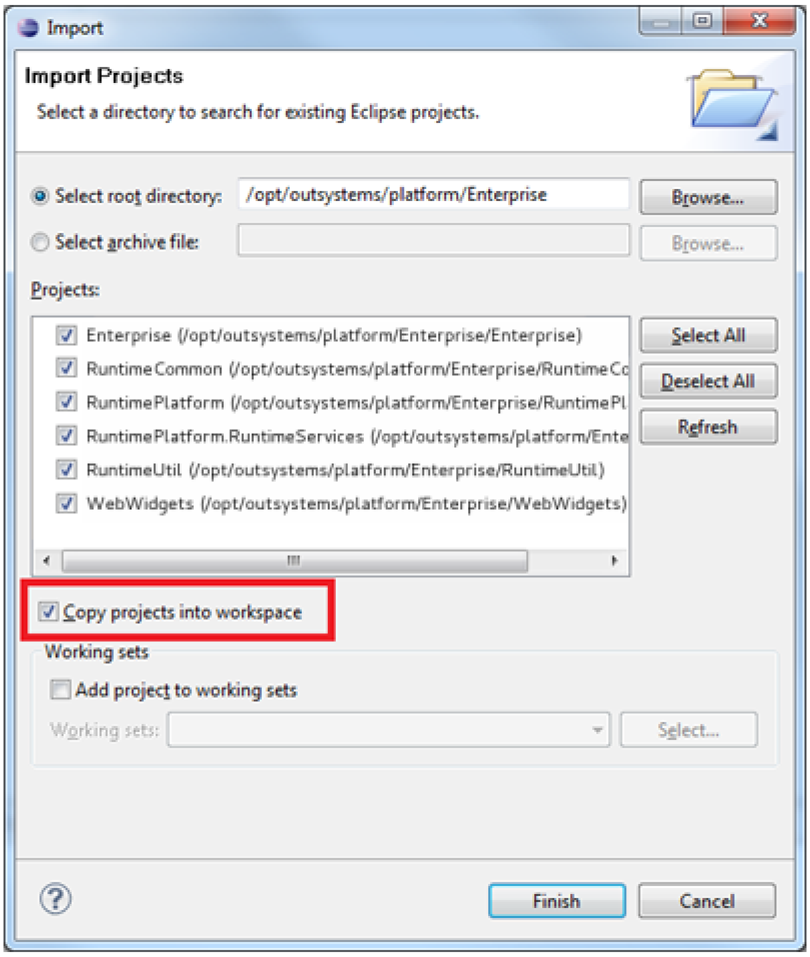
Select

Create new projects from an archive file or directory.

Select an import source:

type filter text

2. Click on the Next button and on the Import Projects dialog select the folder to where you have transferred the generated code. Select all projects and do not forget to check the *Copy projects into workspace* option.


Import

Import Projects

Select a directory to search for existing Eclipse projects.

Select root directory: /opt/outsystems/platform/Enterprise

Select archive file:

Projects:

- [✓] Enterprise (/opt/outsystems/platform/Enterprise/Enterprise)
- [✓] RuntimeCommon (/opt/outsystems/platform/Enterprise/RuntimeCo
- [✓] RuntimePlatform (/opt/outsystems/platform/Enterprise/RuntimePl
- [✓] RuntimePlatform.RuntimeServices (/opt/outsystems/platform/Ente
- [✓] RuntimeUtil (/opt/outsystems/platform/Enterprise/RuntimeUtil)
- [✓] WebWidgets (/opt/outsystems/platform/Enterprise/WebWidgets)

[✓] Copy projects into workspace

Working sets

[ ] Add project to working sets

Working sets:

3. Click on the Finish button for the source code to be imported into your workspace. There should be no compilation errors in Eclipse. If you find any build path errors, check what is missing in the project properties.
4. Finally, copy the following files to the workspace so that the ant targets work:
   - build.xml
   - common.buildfile.xml

5.3 Compiling the application

Each application contains a `build.xml` Ant file with a target for compiling it. This target will generate the application `.war` file at the root folder of your application. Later you will use that file to manually deploy into the application server. To execute this target run the ant command from your workspace directory:

```
ant
```

6 Deploying the Detached Application

The OutSystems Platform for Java supports the following Application Servers:

- RedHat JBoss
- Oracle Weblogic

The instructions below describe how to publish a detached application using a supported Application Server.

6.1 RedHat JBoss

6.1.1 JBoss configuration

There are some dependencies from the OutSystems Platform to JBoss, so you have to run JBoss with the *outsystems* configuration:
In order to setup the *outsystems* configuration, please copy the following files from the machine where you're detaching and has the OutSystems Platform installed, into the same folders in the target machine. Make sure you keep intact the ownership and execution attributes of the copied files.
$JBOSS_HOME/standalone/configuration

- cacerts.truststore
- server.keystore
- standalone-outsystems-mq.xml
- standalone-outsystems.xml

$JBOSS_HOME/bin

- standalone-outsystems.conf
- standalone-outsystems-mq.conf
- standalone-outsystems-mq.properties
- standalone-outsystems.properties

$JBOSS_HOME/modules

- outsystems

/etc/init.d

- jboss-outsystems
- jboss-outsystems-mq

After copying the files, execute the following commands:

- **chkconfig --add jboss-outsystems** , this will install the service, so you can start and stop your service, using service jboss-outsystems [stop|start]
- **chkconfig --add jboss-outsystems-mq** , this will install the service, so you can start and stop your service, using service jboss-outsystems-mq [stop|start]
- Deploy `customHandlers.war` application, located at `/opt/outsystems/platform`:
  ```
  cp /opt/outsystems/platform/customHandlers.war/ ${JBOSS_HOME}/standalone/deployments -R
  ```

- Also, when running JBoss without a load balancer or an Apache Tomcat front-end, ports 80 and 443 must be configured with iptables to route to 8080 and 8443 respectively. For an example on how to do that, check the iptables configuration for the rules marked “outsystems” in the machine from where you are detaching.

6.1.2 Deploying Applications to JBoss

The ant target does not automatically publish the application to JBoss, so you have to publish it manually. To publish the application, just execute the following command from the root folder of your application, in which the `.war` file was generated:

```
        cp Enterprise.war ${JBOSS_HOME}/standalone/deployments
```

6.2 Oracle WebLogic

6.2.1 WebLogic configuration

Some configurations of the server environment are performed by the OutSystems Configuration Tool. You can either run the configuration tool on the target server or go through the following steps:

- Login to your WebLogic Administration console and deploy:
    - `customHandlers.war` application, which is located in `/opt/outsystems/platform` directory;
    - `jsf-1.2.war` library, which is located in `${WL_HOME}/common/deployable-libraries` directory.
- Also, when running WebLogic without a load balancer or an Apache Tomcat front-end, ports 80 and 443 must be configured with iptables to route to 8080 and 8443 respectively. For an example on how to do that, check the iptables configuration for the rules marked “outsystems” in the machine from where you are detaching.

6.2.2 Copying required libraries to the WebLogic library folder

In order for your application to work correctly, you will need to copy some files into the WebLogic library folder. These files can be found in the `${MW_HOME}/user_projects/domains/outsystems_domain/lib` directory of the machine from where you are detaching.
Copy all the files in that directory, except for `outsystems.runtimeservices.jar`, into the `${MW_HOME}/user_projects/domains/outsystems_domain/lib` directory on the destination machine.
Additionally, execute the following command from your workspace directory:

```
        cp RuntimeServices/dist/*.jar ${MW_HOME}/user_projects/domains/outsystems_domain/lib
```

Once you have copied these files, you must restart your WebLogic server.

6.2.3 Deploying Applications to WebLogic

The ant target does not automatically publish the application to WebLogic, so you have to publish it manually. To publish the application, login to your WebLogic Administration console and deploy the `.war` file located in the root folder of your application.

7 Additional Steps

7.1 Dependencies

It is likely that your application depends on other applications. If that is the case, you should deploy them all to be sure that you bring all the needed components (particularly, if there are links between applications). To do that, you need to detach and publish each one individually using the instructions in this document.

7.2 Extensions

Extension modules don't generate source code. As such, source code for custom extensions is always available for detachment and reuse. Just use Integration Studio, as usual, to open and compile the source code using Eclipse.

7.3 OutSystems Scheduler Service

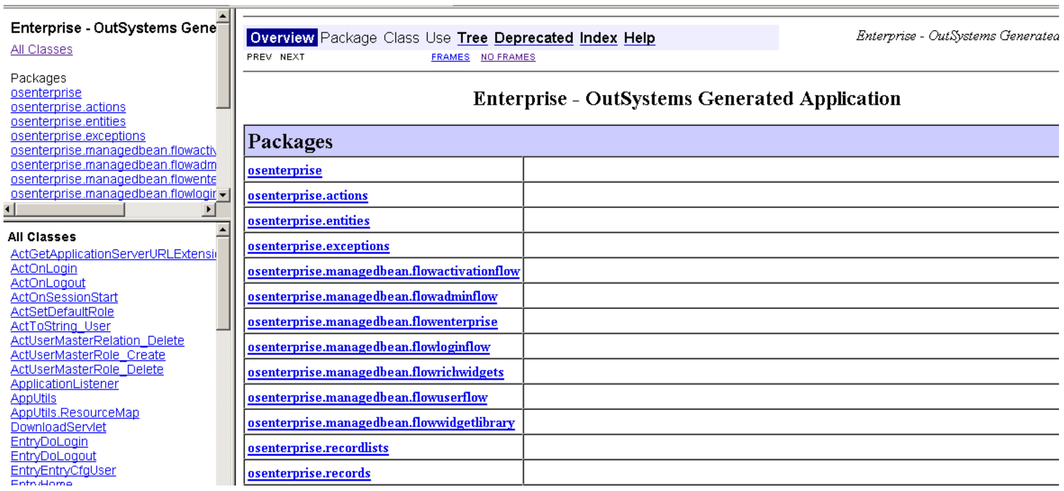
If you are using Timers or BPT Activities in your applications you will be able to keep this functionality even after detaching. Unlike other OutSystems services, the source code of the Scheduler Service will be provided for this purpose. If you choose to re-implement these services using external tools you can later remove this dependency manually.

7.4 Generating Javadoc Documentation

To generate Javadoc documentation all you need to do is to run `ant` with the `javadocs` target

➢ **ant javadocs**

This will create a `docs` subfolder with the Javadoc documentation of your application.



Enterprise - OutSystems Gene
All Classes

Packages
osenterprise
osenterprise.actions
osenterprise.entities
osenterprise.exceptions
osenterprise.managedbean.flowactiv
osenterprise.managedbean.flowadm
osenterprise.managedbean.flowente
osenterprise.managedbean.flowlogir

All Classes
ActGetApplicationServerURLExtensi
ActOnLogin
ActOnLogout
ActOnSessionStart
ActSetDefaultRole
ActToString_User
ActUserMasterRelation_Delete
ActUserMasterRole_Create
ActUserMasterRole_Delete
ApplicationListener
AppUtils
AppUtils.ResourceMap
DownloadServlet
EntryDoLogin
EntryDoLogout
EntryEntryCfgUser

Overview Package Class Use Tree Deprecated Index Help
PREV NEXT FRAMES NO FRAMES
Enterprise - OutSystems Generatea

Enterprise - OutSystems Generated Application

Packages	
osenterprise	
osenterprise.actions	
osenterprise.entities	
osenterprise.exceptions	
osenterprise.managedbean.flowactivationflow	
osenterprise.managedbean.flowadminflow	
osenterprise.managedbean.flowenterprise	
osenterprise.managedbean.flowloginflow	
osenterprise.managedbean.flowrichwidgets	
osenterprise.managedbean.flowuserflow	
osenterprise.managedbean.flowwidgetlibrary	
osenterprise.recordlists	
osenterprise.records	

7.5 Encrypted Configuration Data

The OutSystems Platform encrypts sensitive configuration data such as passwords. After detaching the source code, these configurations need to be reset to plain text.
In the OutSystems Platform database:
Locate the following configurations and reset the encrypted configurations back to plain text:

- Find the OSSYS_DBCATALOG, and update the Password and OwnerPassword column entries to the plain text passwords. You don’t need to change the entry for the Main catalog;
- In the OutSystems Platform database, find the OSSYS_DBCONNECTION table. Update the DATABASE_CONFIGURATION column entries to the plain text using the following format:

  ```
  ProviderKey=Oracle
  <DBConfiguration><ConnectionStringOverride>CONNECTION_STRING_IN_PLAIN_TEXT
  </ConnectionStringOverride></DBConfiguration>
  ```

- Find the OSSYS_PARAMETER table, and update all encrypted entries to plain text. Usually, only passwords are encrypted.

7.6 Database Plugins

The OutSystems Platform already has some database plugins that come within the zip of the detached application. In case of having new plugins, put then in lib folder of the detached applications.