



# **Red Hat JBoss Migration Toolkit 2.7 Windup User Guide**

---

Simplify Migration of Java Applications

Windup Team



## Simplify Migration of Java Applications

## Legal Notice

Copyright © 2017 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

## Abstract

This guide describes how to use Windup to simplify migration of Java applications.

---

## Table of Contents

<b>CHAPTER 1. INTRODUCTION</b> .....	<b>3</b>
1.1. WHAT IS WINDUP?	3
1.2. FEATURES OF WINDUP	3
1.3. SUPPORTED MIGRATION PATHS	4
1.4. ABOUT WINDUP RULES	5
1.5. SYSTEM REQUIREMENTS	5
1.6. ABOUT THE WINDUP_HOME VARIABLE	6
<b>CHAPTER 2. GETTING STARTED</b> .....	<b>7</b>
2.1. INSTALL WINDUP	7
2.2. EXECUTE WINDUP	7
2.3. ACCESS THE REPORT	14
<b>CHAPTER 3. REVIEW THE REPORTS</b> .....	<b>15</b>
3.1. APPLICATION REPORT	16
3.2. ARCHIVES SHARED BY MULTIPLE APPLICATIONS	21
3.3. RULE PROVIDER EXECUTION REPORT	21
3.4. WINDUP FREEMARKER FUNCTIONS AND DIRECTIVES REPORT	21
3.5. SEND FEEDBACK FORM	22
<b>CHAPTER 4. EXPORT THE REPORT IN CSV FORMAT</b> .....	<b>24</b>
4.1. EXPORT THE REPORT	24
4.2. IMPORT THE CSV FILE INTO A SPREADSHEET PROGRAM	24
4.3. OVERVIEW OF THE CSV DATA STRUCTURE	24
<b>CHAPTER 5. MAVENIZE YOUR APPLICATION</b> .....	<b>26</b>
5.1. GENERATE THE MAVEN PROJECT STRUCTURE	26
5.2. REVIEW THE MAVEN PROJECT STRUCTURE	26
<b>CHAPTER 6. OPTIMIZE WINDUP PERFORMANCE</b> .....	<b>29</b>
6.1. TIPS TO OPTIMIZE PERFORMANCE	29
<b>CHAPTER 7. ADDITIONAL RESOURCES</b> .....	<b>30</b>
7.1. REVIEW THE WINDUP QUICKSTARTS	30
7.2. GET INVOLVED	30
7.3. IMPORTANT LINKS	31
7.4. KNOWN WINDUP ISSUES	31
7.5. REPORT ISSUES WITH WINDUP	31
<b>APPENDIX A. REFERENCE MATERIAL</b> .....	<b>33</b>
A.1. GLOSSARY OF TERMS USED IN WINDUP	33



# CHAPTER 1. INTRODUCTION

This guide is for engineers, consultants, and others who plan to use Windup to migrate Java applications or other components.

## 1.1. WHAT IS WINDUP?



### Overview

Windup is an extensible and customizable rule-based tool that helps simplify migration of Java applications.

Windup examines application artifacts, including project source directories and applications archives, then produces an HTML report that highlights areas needing changes. Windup can be used to migrate Java applications from previous versions of *Red Hat JBoss Enterprise Application Platform* or from other containers, such as *Oracle® WebLogic Server* or *IBM® WebSphere® Application Server*.

### How Does Windup Simplify Migration?

Windup looks for common resources and highlights technologies and known trouble spots when migrating applications. The goal is to provide a high-level view into the technologies used by the application and provide a detailed report organizations can use to estimate, document, and migrate enterprise applications to Java EE and JBoss EAP.

## 1.2. FEATURES OF WINDUP

### Shared Data Model

Windup creates a shared data model graph that provides the following benefits.

- ✦ It enables complex rule interaction, allowing rules to pass findings to other rules.
- ✦ It enables third-party plugins to interact with other plugins, rules, and reports.
- ✦ The findings in the data graph model can be searched and queried during rule execution and used for reporting purposes.

### Enhanced Rule Capabilities

Windup provides the ability to create powerful and complex rules.

- ✦ XML-based rules are simple to write and and easy to implement.
- ✦ Java-based rule add-ons provide greater flexibility and power when creating rules.
- ✦ Rules can be nested to handle more complex situations. This means you can nest simple statements rather than use complex XPATH or REGEX expressions.
- ✦ Rules can be linked using and/or statements.

### Extensibility

Windup can be extended by developers, users, and third-party software.

- ✦ It provides a plug-in API to inject other applications into Windup.
- ✦ It enables third-parties to create simple plugins that can interact with the data graph.
- ✦ It allows users with domain knowledge to implement their own rules.

### Targeted Reporting

Windup reports are targeted for specific audiences.

- ✦ Project Management: Reports detail the type of work and estimation of effort to complete the tasks.
- ✦ Developers: Reports provide hints and suggested code changes by class or file.

### Work Estimation

Estimates for the *level of effort* are based on the skills required and the classification of migration work needed. Level of effort is represented as *story points* in the Windup reports.

## 1.3. SUPPORTED MIGRATION PATHS

Windup supports application migration from several platforms to Red Hat JBoss Enterprise Application Platform (JBoss EAP). See the below table for which JBoss EAP version is currently supported by Windup for direct migration from your source platform.

Source Platform	Migration to JBoss EAP 6	Migration to JBoss EAP 7
Oracle® WebLogic Server	✓	✗ <a href="#">[a]</a>
IBM® WebSphere® Application Server	✓	✗ <a href="#">[a]</a>
JBoss EAP 4	✓	✗ <a href="#">[a]</a>
JBoss EAP 5	✓	✓



Source Platform	Migration to JBoss EAP 6	Migration to JBoss EAP 7
JBoss EAP 6	N/A	✓
[a] Although Windup does not currently provide rules for this migration path, Red Hat Consulting can assist with migration from any source platform.		

You will specify the source and target technologies by passing the `--source` and `--target` arguments to the Windup command.

### Example Windup Command: WebLogic to JBoss EAP 6

```
$ WINDUP_HOME/bin/windup --source weblogic --target eap:6 --input
/path/to/application.war
```

### Example Windup Command: JBoss EAP 6 to JBoss EAP 7

```
$ WINDUP_HOME/bin/windup --source eap:6 --target eap:7 --input
/path/to/application.war
```

See [Execute Windup](#) for more information on running Windup.

## 1.4. ABOUT WINDUP RULES

Windup is a rule-based migration tool that analyzes the APIs, technologies, and architectures used by the applications you plan to migrate. In fact, the Windup tool executes its own core set of rules through all phases of the migration process. It uses rules to extract files from archives, decompile files, scan and classify file types, analyze XML and other file content, analyze the application code, and build the reports.

Windup builds a data model based on the rule execution results and stores component data and relationships in a graph database, which can then be queried and updated as needed by the migration rules and for reporting purposes.

Windup rules use the following rule pattern:

```
when(condition)
  perform(action)
otherwise(action)
```

Windup provides a comprehensive set of standard migration rules out-of-the-box. Because applications may contain custom libraries or components, Windup allows you to write your own rules to identify use of components or software that may not be covered by the existing ruleset. If you plan to write your own custom rules, see the [Windup Rules Development Guide](#) for detailed instructions.

## 1.5. SYSTEM REQUIREMENTS

### 1.5.1. Software

- ✦ Java Platform, JRE version 7+
- ✦ Windup is tested on Linux, Mac OS X, and Windows. Other operating systems with Java 7+ support should work equally well.

### 1.5.2. Hardware

The following memory and disk space requirements are the minimum needed to run Windup. If your application is very large or you need to evaluate multiple applications, you may want to increase these values to improve performance. For tips on how to optimize performance, see [Optimize Windup Performance](#).

- ✦ A minimum of 4 GB RAM. For better performance, a 4-core processor with 8 GB RAM is recommended. This allows 3 - 4 GB RAM for use by the JVM.
- ✦ A minimum of 4 GB of free disk space. A fast disk, especially a Solid State Drive (SSD), will improve performance.

## 1.6. ABOUT THE WINDUP\_HOME VARIABLE

This documentation uses the **WINDUP\_HOME** replaceable value to denote the path to the Windup distribution. When you encounter this value in the documentation, be sure to replace it with the actual path to your Windup installation.

- ✦ If you download and install the latest distribution of Windup, **WINDUP\_HOME** refers to the **windup-distribution-2.7.0.Final** folder extracted from the downloaded ZIP file.
- ✦ If you build Windup from GitHub source, **WINDUP\_HOME** refers to the **windup-distribution-2.7.0.Final** folder extracted from the **windup-distribution/target/windup-distribution-2.7.0-SNAPSHOT-offline.zip** file.

## CHAPTER 2. GETTING STARTED

### 2.1. INSTALL WINDUP

1. Download the latest [Windup ZIP distribution](#).
2. Extract the ZIP file in to a directory of your choice.

### 2.2. EXECUTE WINDUP

#### 2.2.1. Run Windup

Use the following steps to run Windup against your application.

1. Open a terminal and navigate to the **WINDUP\_HOME/bin** directory.
2. Execute the **windup** script, or **windup.bat** for Windows, and specify the appropriate arguments.

```
$ ./windup --input /path/to/jee-example-app-1.0.0.ear --output  
/path/to/output --source weblogic --target eap:6 --packages  
com.acme org.apache
```

- ✦ **--input**: The application to be evaluated. See the **--input** argument description.
- ✦ **--output**: The output directory for the generated reports. See the **--output** argument description.
- ✦ **--source**: The source technology for the application migration. See the **--source** argument description.
- ✦ **--target**: The target technology for the application migration. See the **--target** argument description.
- ✦ **--packages**: The packages to be evaluated. This argument is highly recommended to improve performance. See the **--packages** argument description.

See [Windup Command-Line Arguments](#) for a detailed description of all available command-line arguments.

3. Review the report.

Upon completion, you will see the following message in the terminal.

```
Windup report created: PATH_TO_REPORTS/index.html  
Access it at this URL:  
file:///PATH_TO_REPORTS/index.html
```

Navigate to the output directory and open the **index.html** file in a web browser. See the [Review the Reports](#) section for more details on evaluating report data.

See [Windup Command Examples](#) below for concrete examples of commands that use source code directories and archives located in the Windup GitHub repository.

## 2.2.2. Windup Command Examples

### Running Windup on Source Code

The following command runs against the Windup [seam-booking-5.2](#) test application source code. It evaluates all **org.jboss.seam** packages and creates a directory named 'seam-booking-report' in the `/home/username/windup-reports/` directory to contain the reporting output.

```
$ WINDUP_HOME/bin/windup --sourceMode --input /home/username/windup-source/test-files/seam-booking-5.2/ --output /home/username/windup-reports/seam-booking-report --target eap:6 --packages org.jboss.seam
```

### Running Windup on an Application Archive

The following command runs against the Windup [jee-example-app-1.0.0.ear](#) test EAR archive. It evaluates all **com.acme** and **org.apache** packages and creates a directory named 'jee-example-app-1.0.0.ear-report' in the `/home/username/windup-reports/` directory to contain the reporting output.

```
$ WINDUP_HOME/bin/windup --input /home/username/windup-source/test-files/jee-example-app-1.0.0.ear --output /home/username/windup-reports/jee-example-app-1.0.0.ear-report --target eap:6 --packages com.acme org.apache
```

### Override Windup Properties

To override the default *Fernflower* decompiler, pass the **-Dwindup.decompiler** argument on the command line. For example, to use the *Procyon* decompiler, use the following syntax:

```
$ WINDUP_HOME/bin/windup -Dwindup.decompiler=procyon --input INPUT_ARCHIVE_OR_DIRECTORY --output OUTPUT_REPORT_DIRECTORY --target TARGET_TECHNOLOGY --packages PACKAGE_1 PACKAGE_2
```

## 2.2.3. Windup Help

To see the complete list of available arguments for the **windup** command, open a terminal, navigate to the `WINDUP_HOME` directory, and execute the following command:

```
$ WINDUP_HOME/bin/windup --help
```

## 2.2.4. Windup Command-Line Arguments

The following is a detailed description of the available Windup command line arguments.



### Note

To run the Windup command without prompting, for example when executing from a script, use **--batchMode** to take the default values for unspecified parameters and **--overwrite** to force delete the output directory. Also be sure to specify the required **--input** and **--target** arguments.

See the description for each argument for more details.

### **--additionalClassPath** JAR\_OR\_DIRECTORY\_1 JAR\_OR\_DIRECTORY\_2

Use this option to add additional JAR files or directories to the class path so they are available for decompilation or other analysis. For example, **--additionalClassPath MyClasses.jar com/mycompany/**.

### **--addonDir** DIRECTORY

Add the specified directory as a custom add-on repository.

### **--batchMode**

Specifies that Windup should be run in a non-interactive mode without prompting for confirmation. This mode takes the default values for any parameters not passed in via the command line.

### **--debug**

Run Windup in debug mode.

### **--discoverPackages**

Lists all available packages in the input application or source.

### **--enableClassNotFoundAnalysis**

Enables analysis of Java files that are not available on the class path. This should be left off if some classes will be unavailable at analysis time.

### **--enableCompatibleFilesReport**

Enables generation of 'Compatible Files' report. Due to processing all files without found issues, this report may take a long time for large applications.

### **--enableTattletale**

Enables Tattletale-embedded processing and Windup will generate a Tattletale report for each application.

### **--excludePackages** PACKAGE\_1 PACKAGE\_2 PACKAGE\_N

This is a space-delimited list of the packages to be excluded by Windup. For example, entering "com.mycompany.commonutilities" would exclude all classes whose package name begins with "com.mycompany.commonutilities".

This parameter is very important when dealing with large applications as it can greatly reduce execution time.

### **--excludeTags** TAG\_1 TAG\_2

Do not process rules that contain the specified tags. This option can be used if it is found

that a particular set of rules is highlighting too much unnecessary information in the report. If this option is not specified, all tags are processed.



#### Note

For the list of the available tags, pass the **--listTags** argument to the **windup** command.

#### **--explodedApp**

If used, indicates the directory contains source files for a single application or directory entries for multiple applications. See the [Input File Argument Description Table](#) for details.

#### **--exportCSV**

Export the report data to a CSV file on your local file system. Windup creates the file in the directory specified by the **--output** argument. The CSV file can be imported into a spreadsheet program for data manipulation and analysis. For details, see [Export the Report in CSV Format](#).

#### **--help**

Display the Windup help message.

#### **--immutableAddonDir DIRECTORY**

Add the specified directory as a custom read-only add-on repository.

#### **--includeTags TAG\_1 TAG\_2**

In Windup, each rule is associated with a set of tags. Tags are just simple strings that succinctly describe the function of the rule. Common tags include "ejb", "log4j", and "hibernate". To see the full list of tags, use the **--listTags** argument.

When one or more tags are specified here, then only rules with these tags will be processed. If this option is not specified, then all tags are processed.

#### **--input INPUT\_ARCHIVE\_OR\_DIRECTORY [...]**

Each input argument is a fully qualified path to a file or directory containing one or more applications to be migrated. Multiple paths are separated by a space. This argument is required and can appear multiple times in the command.

When used in combination with the following arguments, the file input type is evaluated as follows.

**Table 2.1. Input File Argument Description Table**

Input File Type	<b>--explodedApp</b> Argument	<b>--sourceMode</b> Argument	Neither Argument Specified
Directory	Directory evaluated as a single application.	Directory evaluated as a single application.	Each directory entry is evaluated as a single application.

Input File Type	--explodedApp Argument	--sourceMode Argument	Neither Argument Specified
File	Argument is ignored and the file is evaluated as a single application.	The file is evaluated as a compressed project.	The file is evaluated as a single application.

**--install GROUP\_ID:ARTIFACT\_ID[:VERSION]**

Install the specified add-ons. For example, **--install core-addon-x** or **--install org.example.addon:example,1.0.0**.

**--keepWorkDirs**

Instructs Windup to not delete temporary working files, such as the graph database and unzipped archives. This is useful for debugging purposes.

**--list**

List installed add-ons.

**--listSourceTechnologies**

List all available source technologies.

**--listTags**

List all available tags.

**--listTargetTechnologies**

List all available target technologies.

**--mavenize**

Create a Maven project directory structure based on the structure and content of the application. This creates **pom.xml** files using the appropriate Java EE API and the correct dependencies between project modules. See also the **--mavenizeGroupId** option.

**--mavenizeGroupId**

When used with the **--mavenize** option, all generated **pom.xml** files will use this value for their **<groupId>**. If this parameter is omitted, Windup will attempt to determine an appropriate **<groupId>** based on the application, or will default to **com.mycompany.mavenized**.

**--offline**

If specified, do all processing offline and do not fetch updates or other data from the Internet.

**--output OUTPUT\_REPORT\_DIRECTORY**

This is the fully qualified path to the directory that will contain the report information produced by Windup.

- ✦ If omitted, the report will be generated in an **INPUT\_ARCHIVE\_OR\_DIRECTORY.report** directory.

- ✦ If the output directory exists, you will be prompted with the following (with a default of N).

```
Overwrite all contents of
"/home/username/OUTPUT_REPORT_DIRECTORY" (anything already in
the directory will be deleted)? [y,N]
```

However, if you specify the **--overwrite** argument, Windup will proceed to delete and recreate the directory. See the description of this argument for more information.

### **--overwrite**

Specify this argument only if you are certain you want to force Windup to delete the existing `OUTPUT_REPORT_DIRECTORY` directory. If you do not specify this argument and the **--output** directory exists, you are prompted to choose whether to overwrite the contents.

#### **Warning**

Be careful not to specify a report output directory that contains important information!

### **--packages PACKAGE\_1 PACKAGE\_2 PACKAGE\_N**

A space delimited list of the packages to be evaluated by Windup. It is highly recommended to use this argument.

- ✦ In most cases, you are interested only in evaluating custom application class packages and not standard Java EE or 3rd party packages. The **PACKAGE\_N** argument is a package prefix; all subpackages will be scanned. For example, to scan the packages **com.mycustomapp** and **com.myotherapp**, use **--packages com.mycustomapp com.myotherapp** argument on the command line.
- ✦ While you can provide package names for standard Java EE 3rd party software like **org.apache**, it is usually best not to include them as they should not impact the migration effort.

#### **Warning**

If you omit the **--packages** argument, every package in the application is scanned, which can impact performance. It is best to provide this argument with one or more packages. For additional tips on how to improve performance, see [Optimize Windup Performance](#).

### **--remove GROUP\_ID:ARTIFACT\_ID[:VERSION]**

Remove the specified add-ons. For example, **--remove core-addon-x** or **--remove org.example.addon:example,1.0.0**.

### **--source SOURCE\_1 SOURCE\_2**



A space-delimited list of one or more source technologies, servers, platforms, or frameworks to migrate from. This argument, in conjunction with the `--target` argument, helps to determine which rulesets are used. Use the `--listSourceTechnologies` argument to list all available sources.

The `--source` argument now provides version support, which follows the [Maven version range syntax](#). This instructs Windup to only run the rulesets matching the specified versions. For example, `--source eap:5`.

### Warning

When migrating to JBoss EAP, be sure to specify the version, for example, `eap:6`. Specifying only `eap` will run rulesets for all versions of JBoss EAP, including those not relevant to your migration path.

See [Supported Migration Paths](#) for which JBoss EAP version is appropriate for your source platform.

### `--sourceMode`

If used, indicates the application to be evaluated contains source files rather than compiled binaries. See the [Input File Argument Description Table](#) for details.

### `--target TARGET_1 TARGET_2`

A space-delimited list of one or more target technologies, servers, platforms, or frameworks to migrate to. This argument, in conjunction with the `--source` argument, helps to determine which rulesets are used. If you do not specify this option, you are prompted to select a target. Use the `--listTargetTechnologies` argument to list all available targets.

The `--target` argument now provides version support, which follows the [Maven version range syntax](#). This instructs Windup to only run the rulesets matching the specified versions. For example, `--target eap:7`.

### Warning

When migrating to JBoss EAP, be sure to specify the version in the target, for example, `eap:6`. Specifying only `eap` will run rulesets for all versions of JBoss EAP, including those not relevant to your migration path.

See [Supported Migration Paths](#) for which JBoss EAP version is appropriate for your source platform.

### `--updateRulesets`

Update the core rulesets distributed with Windup. It first checks for the existence of newer release, and if found, replaces the current rulesets directory with the new one.



### Note

To update the rulesets without analyzing an application, pass only this argument on the **windup** command line as in the following example.

```
$ WINDUP_HOME/bin/windup --updateRulesets
```

### **--userIgnorePath CUSTOM\_IGNORE\_DIRECTORY**

Windup looks for file names matching the pattern **\*windup-ignore.txt** to identify files that should be ignored. By default, it looks for these files in the **~/.windup/ignore/** and **WINDUP\_HOME/ignore/** directories, but this option allows you to create files with this pattern name in a different directory.

### **--userRulesDirectory CUSTOM\_RULES\_DIRECTORY**

By default, Windup looks for rules in the **`\${user.home}/.windup/rules/** directory. This option allows you to provide the fully qualified path to a user directory containing additional custom XML rules that should be loaded and executed by Windup. The ruleset files must use one of the following extensions: **\*.windup.groovy** or **\*.windup.xml**.

### **--version**

Display the Windup version.

## 2.3. ACCESS THE REPORT

When you execute Windup, the report is generated in the **OUTPUT\_REPORT\_DIRECTORY** that you specify using the **--output** argument in the command line. This output directory contains the following files and subdirectories:

```
OUTPUT_REPORT_DIRECTORY/
├─ index.html           // Landing page for the report
├─ EXPORT_FILE.csv     // Optional export of data in CSV format
├─ archives/          // Archives extracted from the application
├─ mavenized/         // Optional Maven project structure
├─ reports/           // Generated HTML reports
├─ stats/             // Performance statistics
```

Use a browser to open the **index.html** file located in the report output directory. See the [Review the Reports](#) section for information on navigating the Windup reports.

## CHAPTER 3. REVIEW THE REPORTS

The report examples shown in the following sections are a result of analyzing the **com.acme** and **org.apache** packages in the [jee-example-app-1.0.0.ear](#) example application, which is located in the Windup GitHub source repository. The report was generated using the following command.

```
WINDUP_HOME/bin/windup --input /home/username/windup-source/test-
files/jee-example-app-1.0.0.ear/ --output /home/username/windup-
reports/jee-example-app-1.0.0.ear-report --target eap:6 --packages
com.acme org.apache
```

Use a browser to open the **index.html** file located in the report output directory. This opens a landing page that lists the applications that were processed. Each row contains a high-level overview of the story points, number of incidents, and technologies encountered in that application.

**Figure 3.1. Application List**

### Application List

The Application List report shows all applications which were analyzed. Click on an individual application to see individual reports or you can follow the global Migration Issues report.

**jee-example-app-1.0.0.ear**

WebLogic EJB XML WebLogic Web XML Apache License 2.0 EJB XML 2.1 Manifest  
Maven XML Properties Unknown License Web XML 2.4

**84**  
story points

Number of incidents  
45 Potential Issues  
19 Optional  
48 Mandatory  
112 Total

[Executed rules overview](#) | [Windup FreeMarker methods](#) | [Send feedback](#)



#### Note

The incidents and estimated story points change as new rules are added to Windup. The values here may not match what you see when you test this application.

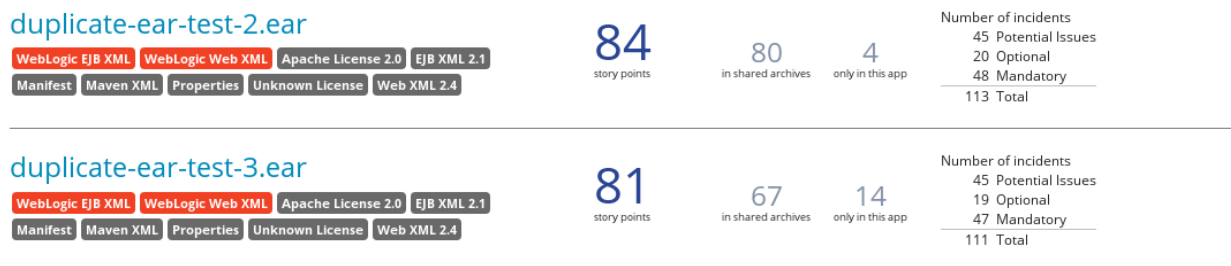
Click on the name of the application, **jee-example-app-1.0.0.ear**, to view the [application report](#). The following table lists all of the reports that can be access from this main Windup landing page.

Report	How to Access the Report
<a href="#">Application</a>	Click on the name of the application.
<a href="#">Archives shared by multiple applications</a>	Click on the <b>Archives shared by multiple applications</b> link. Note that this link is only available when there are shared archives across multiple applications.
<a href="#">Rule Provider Executions</a>	Click on the <b>Executed rules overview</b> link at the bottom of the page.

Report	How to Access the Report
<a href="#">Windup Freemarker Functions and Directives</a>	Click on the <b>Windup FreeMarker methods</b> link at the bottom of the page.
<a href="#">Send Feedback Form</a>	Click on the <b>Send feedback</b> link at the bottom of the page to open a form that allows you to submit feedback to the Windup team.

Note that if an application shares archives with other analyzed applications, you will see a breakdown of how many story points are from shared archives and how many are unique to this application.

Figure 3.2. Shared Archives



Information about the archives that are shared among applications can be found in the [Archives Shared by Multiple Applications](#) reports.

## 3.1. APPLICATION REPORT

### 3.1.1. Report Index

Access this report from the report landing page by clicking on the application name in the **Application List**.

The application report page gives an overview of the entire application migration effort. It summarizes:

- » The incidents and story points by category (mandatory, optional, or potential issues)
- » The incidents and story points by level of effort of the suggested changes
- » The incidents by package

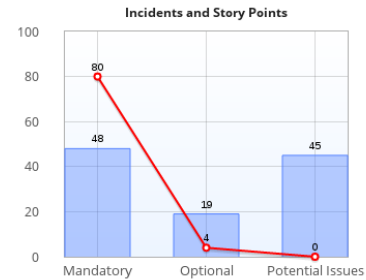
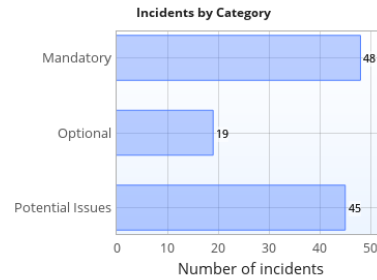
Figure 3.3. Report Index

## Report Index

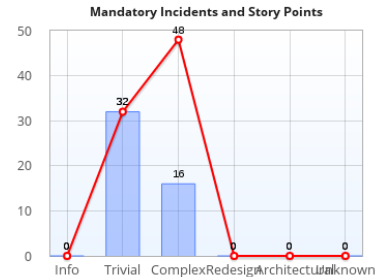
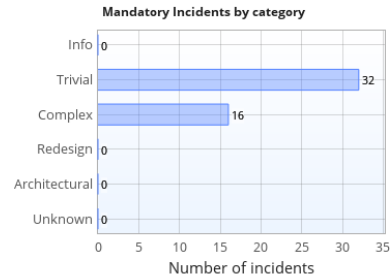
jee-example-app-1.0.0.ear

This report provides summary information about findings from the migration analysis, as well as links to additional reports with detailed information.

Incidents by Category	Incidents	Total Story Points
Mandatory	48	80
Optional	19	4
Potential Issues	45	0



Mandatory Incidents by Type	Incidents	Total Story Points
Info	0	0
Trivial	32	32
Complex	16	48
Redesign	0	0
Architectural	0	0
Unknown	0	0



At the bottom of the page is a list of reports that contain additional details about the migration of this application. Note that only those reports that are applicable to the current application will be available.

Report	Description
Migration Issues	Provides a concise summary of all issues that require attention.
Application Details	Provides a detailed overview of all resources found within the application that may need attention during the migration.
Unparsable	Shows all files that Windup could not parse in the expected format. For instance, a file with a <code>.xml</code> or <code>.wsdl</code> suffix is assumed to be an XML file. If the XML parser fails, the issue is reported here and also where the individual file is listed.
Dependencies	Displays all Java-packaged dependencies found within the application.
Remote Services	Displays all remote services references that were found within the application.
EJBs	Contains a list of EJBs found within the application.

Report	Description
JBPM	Contains all of the JBPM-related resources that were discovered during analysis.
JPA	Contains details on all JPA-related resources that were found in the application.
Hibernate	Contains details on all Hibernate-related resources that were found in the application.
Server Resources	Displays all server resources (for example, JNDI resources) in the input application.
Spring Beans	Contains a list of Spring beans found during the analysis.
Hard-coded IP Addresses	Provides a list of all hard-coded IP addresses that were found in the application.
Ignored Files	Lists the files found in the application that, based on certain rules and Windup configuration, were not processed. See the <code>--userIgnorePath</code> option for more information.
About	Describes the current version of Windup and provides helpful links for further assistance.

### 3.1.2. Application Details Report

Access this report from the report index by clicking the **Application Details** link.

The report lists the story points, the Java incidents by package, and a count of the occurrences of the technologies found in the application. Next is a display of application messages generated during the migration process. Finally, there is a breakdown of this information for each archive analyzed during the process.

#### Figure 3.4. Application Details Report

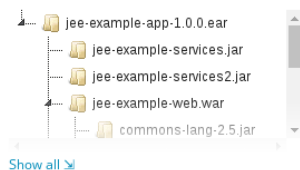
# Application Details Report

jee-example-app-1.0.0.ear

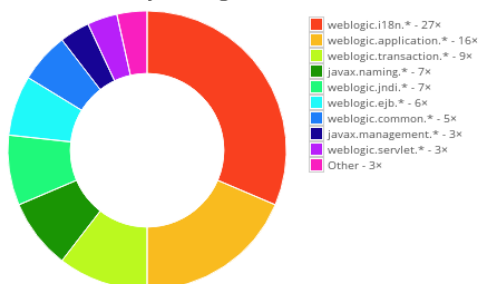
This provides a detailed overview of all resources found within the application that may need attention during the migration.

# 84

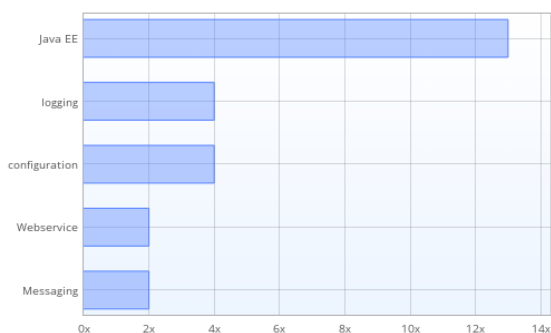
Story Points



## Java Incidents by Package



## Technologies found - occurrence count



Expand the **jee-example-app-1.0.0.ear/jee-example-services.jar** to review the story points, Java incidents by package, and a count of the occurrences of the technologies found in this archive. This summary begins with a total of the story points assigned to its migration, followed by a table detailing the changes required for each file in the archive. The report contains the following columns.

Column Name	Description
Name	The name of the file being analyzed.
Technology	The type of file being analyzed, for example: Java Source, Decompiled Java File, Manifest, Properties, EJB XML, Spring XML, Web XML, Hibernate Cfg, Hibernate Mapping
Issues	Warnings about areas of code that need review or changes.
Story Points	Level of effort required to migrate the file.  <i>Story Points</i> are covered in more detail in the <a href="#">Windup Rules Development Guide</a> .

Note that if an archive is duplicated several times in an application, it will be listed just once in the report and will be tagged with **[Included Multiple Times]**.

Figure 3.5. Duplicate Archive in an Application

🔍 [Included Multiple Times] jee-example-services.jar (61 story points)

# 61

Story Points

<b>Maven coordinates</b>	org.windup.example:test-app
<b>Organization</b>	Unknown
<b>Name</b>	JEE Example EJB Services
<b>Version</b>	1.0.0
<b>Links</b>	
<b>Description</b>	
<b>Duplicates</b>	test-app.ear/copy1/jee-example-services.jar test-app.ear/copy2/jee-example-services.jar

The story points for archives that are duplicated within an application will be counted only once in the total story point count for that application.

### 3.1.3. Source Report

The analysis of the `jee-example-services.jar` lists the files in the JAR and the warnings and story points assigned to each one. Notice the `com.acme.anvil.listener.AnvilWebLifecycleListener` file, at the time of this test, has 22 warnings and is assigned 16 story points. Click on the file link to see the detail.

- The **Information** section provides a summary of the story points and notes that the file was decompiled by Windup.
- This is followed by the file source code listing. Warnings appear in the file at the point where migration is required.

In this example, warnings appear at various import statements, declarations, and method calls. Each warning describes the issue and the action that should be taken.

Figure 3.6. Source Report

## Source Report

jee-example-app-1.0.0.ear/jee-example-services.jar/com/acme/anvil/listener/AnvilWebLifecycleListener.java

🔍 This report displays what Windup found in individual files. Each item is shown below the line it was found on, and next to it, you may find a link to the rule which it was found by.

**Information**

# 16

Story Points

Technologies

Decompiled Java File

```

01. package com.acme.anvil.listener;
02.
03. import com.acme.anvil.management.AnvilInvokeBeanImpl;
04. import java.util.Properties;
05. import javax.management.MBeanServer;
06. import javax.management.ObjectName;
07. import javax.naming.InitialContext;
08. import javax.naming.NamingException;
09. import org.apache.log4j.Logger;
10. import weblogic.application.ApplicationLifecycleEvent;
          
```

🔍 **WebLogic ApplicationLifecycleEvent**

WebLogic ApplicationLifecycleEvent must be replaced with standard Java EE ServletContextEvent. Otherwise, a custom solution using CDI's ApplicationScoped beans or EJB's @Startup beans is required in order to propagate a custom event object because ServletContextEvent types are not extendible in the standard Java EE programming model.

Use a javax.servlet.ServletContextListener with @javax.annotation.servlet.WebListener, or an EJB 3.1 @javax.ejb.Startup @javax.ejb.Singleton service bean.

- [Migrate WebLogic ApplicationLifecycleEvent to standard EJB with JBoss EAP](#)
- [Java EE ServletContextEvent JavaDoc](#)
- [WebLogic custom ApplicationLifecycleEvent Documentation](#)



## 3.2. ARCHIVES SHARED BY MULTIPLE APPLICATIONS

Access these reports from the report landing page by clicking the **Archives shared by multiple applications** link. Note that this link is only available if there are applicable shared archives.

Figure 3.7. Archives Shared by Multiple Applications

### Cross-application Reports

These reports contain information about all issues found in archives which were included in multiple applications.

#### Archives shared by multiple applications

WebLogic EJB XML WebLogic Web XML Apache License 2.0 EJB XML 2.1  
Manifest Maven XML Properties Unknown License Web XML 2.4

80  
story points

Number of incidents  
45 Potential Issues  
17 Optional  
46 Mandatory  
108 Total

This allows you to view the detailed reports for all archives that are shared across multiple applications.

## 3.3. RULE PROVIDER EXECUTION REPORT

Access this report from the report landing page by clicking the **All Rules** link.

This report provides the list of rules that executed when running the Windup migration command against the application.

Figure 3.8. Rule Provider Report

### Rule Provider Executions

This report lists "rule providers", or sets of Windup rules. They may originate from a `.windup.xml` file or a Java class implementing `RuleProvider`.

Phase: DependentPhase

Phase: InitializationPhase

#### RegisterApiPackagesInTypeInterestFactoryRuleProvider

Phase: InitializationPhase

Rule-ID	Rule	Statistics	Status?	Result?	Failure Cause
RegisterApiPackagesInTypeInterestFactoryRuleProvider_1	<pre>addRule()   .perform(org.jboss.windup.rules.apps.mavenize     .RegisterApiPackagesInTypeInterestFactoryRuleProvider\$2@2ae81c68   )   withId("RegisterApiPackagesInTypeInterestFactoryRuleProvider_1")</pre>	Vertices Created: 0 Edges Created: 0 Vertices Removed: 0 Edges Removed: 0	Condition met.	success	
RegisterApiPackagesInTypeInterestFactoryRuleProvider_2	<pre>addRule()   .perform(org.jboss.windup.rules.apps.mavenize     .RegisterApiPackagesInTypeInterestFactoryRuleProvider\$1@24b44bb8   )   withId("RegisterApiPackagesInTypeInterestFactoryRuleProvider_2")</pre>	Vertices Created: 0 Edges Created: 0 Vertices Removed: 0 Edges Removed: 0	Condition met.	success	


## 3.4. WINDUP FREEMARKER FUNCTIONS AND DIRECTIVES REPORT

Access this report from the report landing page by clicking the **Windup FreeMarker methods** link.

This report lists all the registered functions and directives that were used to build the report. It is useful if you plan to build your own custom report or for debugging purposes.

**Figure 3.9. FreeMarker Functions and Directives**

## Windup FreeMarker Functions and Directives

 This report shows the custom Freemarker extensions created for and used by Windup.

Functions	
Function Name	Description
getMigrationEffortPointsForFile	Takes a FileModel as a parameter and returns an int containing the effort estimate for this file. (Implemented by: org.jboss.windup.reporting.freemarker.GetEffortForFile)
getArchivesBySHA1	Takes a single String parameter (SHA1 Hash) and returns an Iterable of ArchiveModel instances with the given Hash (Implemented by: org.jboss.windup.rules.apps.java.archives.freemarker.GetArchivesBySHA1Method)
getAllFreeMarkerDirectives	This method takes no parameters, and returns a List of hashes containing a 'name', 'class', and 'description' field. (Implemented by: org.jboss.windup.reporting.freemarker.GetAllFreeMarkerDirectivesMethod)
generateGUID	Generates a unique identifier as a String. (Implemented by: org.jboss.windup.reporting.freemarker.GenerateGUIDMethod)
projectModelToApplicationIndex	Takes a parameter of type ProjectModel and returns the associated ApplicationReportIndexModel. (Implemented by: org.jboss.windup.reporting.freemarker.ProjectModelToApplicationIndexMethod)
getProblemSummaries	Returns a summary of all classification and hints found during analysis in the form of a List. (Implemented by: org.jboss.windup.reporting.freemarker.problemsummary.GetProblemSummariesMethod)

### 3.5. SEND FEEDBACK FORM

Access this feedback form from the report landing page by clicking the **Send feedback** link.

This form allows you to rate the product, talk about what you like and suggestions for improvements.

**Figure 3.10. Send Feedback Form**

### Got Feedback?

**i** Please provide your feedback below:

Rate this page\*  😄 Awesome!  😊 Good  😐 Meh!  😞 Bad  🤢 Horrible!

What do you like?\*

What needs to be improved?\*

Attach file  No file chosen

Name

Email

## CHAPTER 4. EXPORT THE REPORT IN CSV FORMAT

Windup provides the ability to export the report data, including the **Classifications** and **Hints**, to a flat file on your local file system. The export function currently supports the CSV file format, which presents the report data as fields delimited by a comma ( , ) separator.

The CSV file can be imported and manipulated by spreadsheet software such as Microsoft Excel or LibreOffice Calc. Spreadsheet software provides the ability to sort, analyze, evaluate, and manage the result data from a Windup report.

### 4.1. EXPORT THE REPORT

To export the report into a CSV file, run Windup with `--exportCSV` argument. The CSV file will be created in the directory specified by the `--output` argument.

### 4.2. IMPORT THE CSV FILE INTO A SPREADSHEET PROGRAM

1. Start the spreadsheet software (LibreOffice Calc, OpenOffice Calc, Microsoft Excel, etc.).
2. Choose **File** → **Open**.
3. Navigate to the CSV exported file and select it.
4. The data is now ready to analyze in the spreadsheet software.

For more information or to resolve any issues, check the help for your spreadsheet software.

### 4.3. OVERVIEW OF THE CSV DATA STRUCTURE

The CSV formatted output file contains the following data fields:

**Rule Id**

The ID of the rule that generated the given item.

**Problem type**

"Hint" or "Classification"

**Title**

The title of the *Classification* or *Hint*. This field summarizes the issue for the given item.

**Description**

The detailed description of the issue for the given item.

**Links**

URLs that provide additional information about the issue. A link consists of two attributes: the link and a description of the link.

**Application**

The name of the application for which this item was generated.

**File Name**

The name of the file for the given item.

**File Path**

The file path of the file for the given item.

**Line**

The line number of the file for the given item.

**Story points**

The number of story points, which represent the level of effort, assigned to the given item.

## CHAPTER 5. MAVENIZE YOUR APPLICATION

Windup 2.6 introduced the ability to generate an Apache Maven project structure based on the application provided. This will create a directory structure with the necessary Maven Project Object Model (POM) files that specify the appropriate dependencies.

Note that this feature is not intended to create a final solution for your project. It is meant to give you a starting point and identify the necessary dependencies and APIs for your application. Your project may require further customization.

### 5.1. GENERATE THE MAVEN PROJECT STRUCTURE

You can generate a Maven project structure for the provided application by passing in the `--mavenize` flag when executing Windup.

The following example runs Windup using the [jee-example-app-1.0.0.ear](#) test application.

```
$ WINDUP_HOME/bin/windup --input /path/to/jee-example-app-1.0.0.ear --
output /path/to/output --target eap:6 --packages com.acme org.apache --
mavenize
```

This generates the Maven project structure in the `/path/to/output/mavenized` directory.



#### Note

You can only use the `--mavenize` option when providing a compiled application for the `--input` argument. This feature is not available when running Windup against source code.

You can also use the `--mavenizeGroupId` option to specify the `<groupId>` to be used for the POM files. If unspecified, Windup will attempt to identify an appropriate `<groupId>` for the application, or will simply default to `com.mycompany.mavenized`.

### 5.2. REVIEW THE MAVEN PROJECT STRUCTURE

The `/path/to/output/mavenized/APPLICATION_NAME/` directory will contain the following items:

- ✦ A [root POM file](#). This is the `pom.xml` file at the top-level directory.
- ✦ A [BOM file](#). This is the POM file in the directory ending with `-bom`.
- ✦ One or more [application POM files](#). Each module has its POM file in a directory named after the archive.

The example `jee-example-app-1.0.0.ear` application is an EAR archive that contains a WAR and several JARs. There is a separate directory created for each of these artifacts. Below is the Maven project structure created for this application.

```
/path/to/output/mavenized/jee-example-app/
  jee-example-app-bom/pom.xml
  jee-example-app-ear/pom.xml
```

```

jee-example-services2-jar/pom.xml
jee-example-services-jar/pom.xml
jee-example-web-war/pom.xml
pom.xml

```

Review each of the generated files and customize as appropriate for your project. To learn more about Maven POM files, see the [Introduction to the POM](#) section of the Apache Maven documentation.

## Root POM File

The root POM file for the **jee-example-app-1.0.0.ear** application can be found at `/path/to/output/mavenized/jee-example-app/pom.xml`. This file identifies the directories for all of the project modules.

The following modules are listed in the root POM for the example **jee-example-app-1.0.0.ear** application.

```

<modules>
  <module>jee-example-app-bom</module>
  <module>jee-example-services2-jar</module>
  <module>jee-example-services-jar</module>
  <module>jee-example-web-war</module>
  <module>jee-example-app-ear</module>
</modules>

```



### Note

Be sure to reorder the list of modules if necessary so that they are listed in an appropriate build order for your project.

The root POM is also configured to use the [Red Hat JBoss Enterprise Application Platform Maven repository](#) to download project dependencies.

## BOM File

The Bill of Materials (BOM) file is generated in the directory ending in **-bom**. For the example **jee-example-app-1.0.0.ear** application, the BOM file can be found at `/path/to/output/mavenized/jee-example-app/jee-example-app-bom/pom.xml`. The purpose of this BOM is to have the versions of third-party dependencies used by the project defined in one place. For more information on using a BOM, see the [Introduction to the Dependency Mechanism](#) section of the Apache Maven documentation.

The following dependencies are listed in the BOM for the example **jee-example-app-1.0.0.ear** application

```

<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>log4j</groupId>
      <artifactId>log4j</artifactId>
      <version>1.2.6</version>
    </dependency>

```

```

<dependency>
  <groupId>commons-lang</groupId>
  <artifactId>commons-lang</artifactId>
  <version>2.5</version>
</dependency>
</dependencies>
</dependencyManagement>

```

## Application POM Files

Each application module that can be mavenized has a separate directory containing its POM file. The directory name contains the name of the archive and ends in a **-jar**, **-war**, or **-ear** suffix, depending on the archive type.

Each application POM file lists that module's dependencies, including:

- ✧ Third-party libraries
- ✧ Java EE APIs
- ✧ Application submodules

For example, the POM file for the **jee-example-app-1.0.0.ear** EAR, **/path/to/output/mavenized/jee-example-app/jee-example-app-ear/pom.xml**, lists the following dependencies.

```

<dependencies>
  <dependency>
    <groupId>log4j</groupId>
    <artifactId>log4j</artifactId>
    <version>1.2.6</version>
  </dependency>
  <dependency>
    <groupId>org.jboss.seam</groupId>
    <artifactId>jee-example-web-war</artifactId>
    <version>1.0</version>
    <type>war</type>
  </dependency>
  <dependency>
    <groupId>org.jboss.seam</groupId>
    <artifactId>jee-example-services-jar</artifactId>
    <version>1.0</version>
  </dependency>
  <dependency>
    <groupId>org.jboss.seam</groupId>
    <artifactId>jee-example-services2-jar</artifactId>
    <version>1.0</version>
  </dependency>
</dependencies>

```



## CHAPTER 6. OPTIMIZE WINDUP PERFORMANCE

Windup performance depends on a number of factors, including hardware configuration, the number and types of files in the application, the size and number of applications to be evaluated, and whether the application contains source or compiled code. For example, a file that is larger than 10 MB may need a lot of time to process.

In general, Windup spends about 40% of the time decompiling classes, 40% of the time executing rules, and the remainder of the time processing other tasks and generating reports. This section describes what you can do to improve the performance of Windup.

### 6.1. TIPS TO OPTIMIZE PERFORMANCE

#### 6.1.1. Application and Command-Line Suggestions

Try these suggestions first before upgrading hardware.

- ✦ If possible, execute Windup against the source code instead of the archives. This eliminates the need to decompile additional JARs and archives.
- ✦ Specify a comma-separated list of the packages to be evaluated by Windup using the **--packages** argument on the **WINDUP\_HOME/bin/windup** command line. If you omit this argument, Windup will decompile everything, which has a big impact on performance.
- ✦ Specify the **--excludePackages** and **--excludeTags** arguments where possible to exclude them from processing.
- ✦ Add additional proprietary packages that should not be processed to the **ignore/proprietary.package-ignore.txt** file in the Windup distribution directory. Windup can still find the references to the packages in the application source code, but avoids the need to decompile and analyze the proprietary classes.
- ✦ If you have access to a server that has better resources than your laptop or desktop machine, you may want to consider running Windup on that server.

#### 6.1.2. Hardware Upgrade Suggestions

If the application and command-line suggestions above do not improve performance, you may need to upgrade your hardware.

- ✦ If you have access to a server that has better resources than your laptop/desktop, then you may want to consider running Windup on that server.
- ✦ Very large applications that require decompilation have large memory requirements. 8 GB RAM is recommended. This allows 3 - 4 GB RAM for use by the JVM.
- ✦ An upgrade from a single or dual-core to a 4-core CPU processor provides better performance.
- ✦ Disk space and fragmentation can impact performance. A fast disk, especially a solid-state drive (SSD), with greater than 4 GB of defragmented disk space should improve performance.

## CHAPTER 7. ADDITIONAL RESOURCES

### 7.1. REVIEW THE WINDUP QUICKSTARTS

The Windup quickstarts provide working examples of how to create custom Java-based rule additions and XML rules. You can use them as a starting point for creating your own custom rules.

You can download a ZIP file of the latest released version of the quickstarts. Or, if you prefer to play around with the source code, you can fork and clone the `windup-quickstarts` project repository.

#### 7.1.1. Download the Latest Quickstart ZIP

1. Open a browser and navigate to <https://github.com/windup/windup-quickstarts/releases>.
2. Click on the most recent release to download the ZIP file to your local file system.

#### 7.1.2. Fork and Clone the Quickstart GitHub Project

If you do not have the GitHub client (`git`), download it from <http://git-scm.com/> and install it.

1. Click the **Fork** link on the [Windup quickstart](#) GitHub page to create the project in your own Git. The forked GitHub repository URL created by the fork should look like this:  
[https://github.com/YOUR\\_USER\\_NAME/windup-quickstarts.git](https://github.com/YOUR_USER_NAME/windup-quickstarts.git)
2. Clone your Windup quickstart repository to your local file system:

```
$ git clone https://github.com/YOUR_USER_NAME/windup-quickstarts.git
```

3. This creates and populates a **windup-quickstarts** directory on your local file system. Navigate to the newly created directory, for example

```
$ cd windup-quickstarts/
```

4. If you want to be able to retrieve the latest code updates, add the remote **upstream** repository so you can fetch any changes to the original forked repository.

```
$ git remote add upstream https://github.com/windup/windup-quickstarts.git
```

5. To get the latest files from the **upstream** repository.

```
$ git reset --hard upstream/master
```

### 7.2. GET INVOLVED

To help us make Windup cover most application constructs and server configurations, including yours, you can help with any of the following items.

- ✦ Send an email to [windup-users@lists.jboss.org](mailto:windup-users@lists.jboss.org) and let us know what Windup migration rules should cover.
- ✦ Provide example applications to test migration rules.
- ✦ Identify application components and problem areas that may be difficult to migrate.
  - Write a short description of these problem migration areas.
  - Write a brief overview describing how to solve the problem migration areas.
- ✦ [Try Windup](#) on your application. Be sure to [report any issues](#) you encounter.
- ✦ Contribute to the Windup rules repository.
  - Write a Windup rule to identify or automate a migration process.
  - Create a test for the new rule.
  - Details are provided in the [Windup Rules Development Guide](#).
- ✦ Contribute to the project source code.
  - Create a core rule.
  - Improve Windup performance or efficiency.
  - See the [Windup Core Development Guide](#) for information about how to configure your environment and set up the project.

Any level of involvement is greatly appreciated!

### 7.3. IMPORTANT LINKS

- ✦ Windup wiki: <https://github.com/windup/windup/wiki>
- ✦ Windup forums: <https://community.jboss.org/en/windup>
- ✦ Windup JIRA issue trackers
  - Core Windup: <https://issues.jboss.org/browse/WINDUP>
  - Windup Rules: <https://issues.jboss.org/browse/WINDUPRULE>
- ✦ Windup users mailing List: [windup-users@lists.jboss.org](mailto:windup-users@lists.jboss.org)
- ✦ Windup on Twitter: [@JBossWindup](#)
- ✦ Windup IRC channel: Server FreeNode ([irc.freenode.net](https://www.freenode.net)), channel [#windup](#) ([transcripts](#)).

### 7.4. KNOWN WINDUP ISSUES

You can review known issues for Windup here: [Open Windup issues](#)

### 7.5. REPORT ISSUES WITH WINDUP

Windup uses JIRA as its issue tracking system. If you encounter an issue executing Windup, please file a JIRA Issue.

**Note**

If you do not have one already, you must sign up for a JIRA account in order to create a JIRA issue.

### 7.5.1. Create a JIRA Issue

1. Open a browser and navigate to <https://issues.jboss.org/secure/CreateIssue!default.jspa>.

If you have not yet logged in, click the **Log In** link at the top right side of the page and enter your credentials.

2. Choose the following options and click the **Next** button.

- ✦ **Project**

For core Windup issues, choose *Windup: (WINDUP)*.

For issues with Windup rules, choose: *Windup rules (WINDUPRULES)*.

- ✦ **Issue Type:** *Bug*

3. On the next screen complete the following fields:

- ✦ **Summary:** Enter a brief description of the problem or issue.

- ✦ **Environment:** Provide the details of your operating system, version of Java, and any other pertinent information.

- ✦ **Description:** Provide a detailed description of the issue. Be sure to include logs and exceptions traces.

- ✦ **Attachment:** If the application or archive causing the issue does not contain sensitive information and you are comfortable sharing it with the Windup development team, attach it to the issue using the **Select Files** button.

4. Click the **Create** button to create the JIRA issue.

## APPENDIX A. REFERENCE MATERIAL

### A.1. GLOSSARY OF TERMS USED IN WINDUP

#### A.1.1. Rules Terms

##### Rule

A piece of code that performs a single unit of work during the migration process. The following is an example of an XML-based rule that finds and reports on instances of **weblogic-ejb-jar** XML files and provides a link to an article on the Red Hat Customer Portal that describes how to migrate the file.

```
<rule id="weblogic-xml-descriptor-04000">
  <when>
    <xmlfile matches="/weblogic-ejb-jar" />
  </when>
  <perform>
    <classification title="WebLogic EJB XML"
severity="mandatory" effort="3">
      <link href="https://access.redhat.com/articles/1326823"
title="Map weblogic-ejb-jar.xml Elements to the jboss-ejb3.xml
Descriptor" />
      <tag>ejb</tag>
    </classification>
  </perform>
</rule>
```

##### RuleProvider

An implementation of OCPSoft ConfigurationProvider class specifically for Windup. It provides Rule instances and the relevant RuleProviderMetadata for those Java-based and XML-based Rule instances.

##### Ruleset

A ruleset is a group of one or more RuleProviders that targets a specific area of migration, for example, **Spring** → **Java EE 6** or **WebLogic** → **JBoss EAP**. A ruleset is packaged as a JAR and contains additional information needed for the migration, such as operations, conditions, report templates, static files, metadata, and relationships to other rulesets. The following Windup projects are rulesets.

- ✱ rules-java-ee
- ✱ rules-xml

##### Rules Metadata

Information about whether a particular ruleset applies to a given situation. The metadata can include the source and target platform and frameworks.

##### Rules Pipeline

A collection of rules that feed information into the knowledge graph.

## A.1.2. Reporting Terms

### Level of effort

The effort required to complete the migration task. *Level of effort* is represented as *story points* in the Windup reports.

### Story Point

A term commonly used in Scrum Agile software development methodology to estimate the *level of effort* needed to implement a feature or change. It does not necessarily translate to man-hours, but the value should be consistent across tasks. Story points are covered in more detail in the [Windup Rules Development Guide](#).

*Revised on 2017-02-17 00:07:20 EST*