

# **OZONE Widget Framework**

**Welcome to the New Build**

**September 28, 2012**

### **Publication/Revision History**

<b>Release</b>	<b>Date</b>
Initial Document – OWF 6	September 28, 2012

# Contents

<b>1 Introduction .....</b>	<b>1</b>
1.1 Objectives.....	1
1.2 Requirements.....	1
1.2.1 Download and Install Applications .....	1
1.2.2 Setting Environment Variables .....	2
1.2.3 Configuring Ruby Gems (Compass and Sass) .....	3
1.2.4 Verify Tool Installations .....	4
<b>2 Build-Base .....</b>	<b>6</b>
2.1 OWF and Server builds .....	6
2.1.1 Additional build command line options .....	7
2.1.2 Building Offline .....	7
2.1.3 Troubleshooting.....	8
2.2 Ozone Security Jar.....	9
2.3 CAS Server War .....	9
2.4 Tomcat-OWF-Custom.....	10
<b>3 Building a Release .....</b>	<b>12</b>
3.1 Branching and Tagging.....	12
<b>Appendix A Contact Information.....</b>	<b>A-1</b>
A.1 Discussion Group .....	A-1
A.2 Additional POCs.....	A-1

# Tables

Table 1: OWF Development Tools and Version Numbers.....	1
Table 2: Tool Provider Websites .....	2

# 1 Introduction

## 1.1 Objectives

The purpose of this document is to describe how to build the following projects:

- **OWF Server** – A local copy of the bundled (owf-server.war + cas-server.war + owf-samples + Tomcat) zip file as well as the release version.
- **OWF Security jar** – A new version of the owf-security jar file.
- **build-base** – A common set of files used by all the other project builds. Its role is described below.

## 1.2 Requirements

Before any of the build tasks can be run, a number of supporting tools need to be installed and configured. The following sections list the tools, the version requirements for recent OWF releases, and nominal configuration elements. This document is targeted to Widget Developers and assumes a basic familiarity with the target development environment, be it Microsoft Windows or Linux. Instructions provided below assume a Microsoft Windows development environment for illustrative purposes only.

### 1.2.1 Download and Install Applications

OWF Development requires the use of the Java Development Kit (JDK), Apache ANT, Ruby, Ruby Gems Compass and SASS, as well as Groovy. The following table lists the versions of each tool required by the last few OWF and Marketplace releases.

Table 1: OWF Development Tools and Version Numbers

Application	JDK	ANT	GRAILS	RUBY	GEM*	COMPASS*	SASS*	GROOVY
OWF 4	1.6	1.7	1.3.7	1.8.7	1.8.16	0.11.3	3.1.3	-
OWF 5	1.6	1.8	1.3.7	1.8.7	1.8.16	0.11.3	3.1.3	-
OWF 6	1.6	1.8	1.3.7	1.9.2	1.8.16	0.11.3	3.1.1	1.8.8

\* GEM is provided by the Ruby Installation. COMPASS and SASS configuration is described in section [1.2.3](#).

Installation media and instructions for various operating systems can be obtained by the primary websites for each tool or trusted download source. Default locations are provided below. Also, tools should be installed in the order listed below. Once all tools have been installed, the following sections will describe how to configure the environment.

**Table 2: Tool Provider Websites**

Application	LOCATION
JDK	<a href="http://www.oracle.com/technetwork/java/javase/downloads/index.html">http://www.oracle.com/technetwork/java/javase/downloads/index.html</a>
ANT	<a href="http://projects.apache.org/projects/ant.html">http://projects.apache.org/projects/ant.html</a>
GAILS	<a href="http://www.grails.org/">http://www.grails.org/</a>
GROOVY 2.5	<a href="http://groovy.codehaus.org/">http://groovy.codehaus.org/</a>
RUBY	<a href="http://www.ruby-lang.org/en/">http://www.ruby-lang.org/en/</a>

\* GEM is provided by the Ruby Installation. COMPASS and SASS configuration is described in section [1.2.3](#)

## 1.2.2 Setting Environment Variables

The build environment uses Apache Ant for most tasks. For Ant to run properly, it should be able to discover the locations of the other supporting tools. This is accomplished by setting appropriate environment variables describing the install directory of each tool. On Windows 7, the following steps will set the ANT\_HOME variable, assuming an install location of **C:\Apache\_Ant\apache-ant-1.8.3**.

1. Go to the Start menu and select Control Panel.
2. Click the System icon and select the Advanced System Settings link in the resulting window.
3. Click on the Advanced tab in the resulting pop-up and then select the Environment Variables button.
4. For the following steps, if Administrator rights are available, use the System Variables section. If not, use the User variables section.
5. Create/Edit a variable named **ANT\_HOME**. Set its value to **C:\Apache\_Ant\apache-ant-1.8.3**. Save the new variable. On your system, the value should be the location to which Ant was actually installed.
6. Create/Edit the Path variable. If the Path variable already contains the value **%ANT\_HOME%\bin**, click cancel. If the Path doesn't contain the variable, append **;%ANT\_HOME%\bin** (don't include the quotes) to the end of the Path variable. Click OK until the system properties dialog is closed.

This method should be repeated to set a **JAVA\_HOME**, **ANT\_HOME**, **GRAILS\_HOME**, **GROOVY\_HOME**, and **RUBY\_HOME** environment variable for each of their installation folders, respectively.

### 1.2.3 Configuring Ruby Gems (Compass and Sass)

Configuring the required Ruby gems requires that the previous sections were completed. Assuming a correct Ruby installation and Ruby being available on the Path variable, the following steps will install Compass and Sass:

- 1) Open Command Prompt window and enter the following: `gem install compass --version 0.11.7`. There should be a response similar to the following:

```
C:\Users\myusername>gem install compass --version 0.11.7
Fetching: sass-3.1.15.gem (100%)
Fetching: chunky_png-1.2.5.gem (100%)
Fetching: fssm-0.2.9.gem (100%)
Fetching: compass-0.11.7.gem (100%)
Successfully installed sass-3.1.15
Successfully installed chunky_png-1.2.5
Successfully installed fssm-0.2.9
Successfully installed compass-0.11.7
4 gems installed
Installing ri documentation for sass-3.1.15...
Installing ri documentation for chunky_png-1.2.5...
Installing ri documentation for fssm-0.2.9...
Installing ri documentation for compass-0.11.7...
Installing RDoc documentation for sass-3.1.15...
Installing RDoc documentation for chunky_png-1.2.5...
Installing RDoc documentation for fssm-0.2.9...
Installing RDoc documentation for compass-0.11.7...
C:\Users\myusername>
```

- 2) Then run the following command:  
`gem install sass --version 3.1.3`  
There should be a response similar to:

```
C:\Users\myusername>
C:\Users\myusername>gem install sass --version 3.1.3
Fetching: sass-3.1.3.gem (100%)
Successfully installed sass-3.1.3
1 gem installed
Installing ri documentation for sass-3.1.3...
Installing RDoc documentation for sass-3.1.3...
C:\Users\myusername>
```

- 3) Ensure that the correct ruby, sass, and compass versions are installed.
- 4) From a Cmd prompt window, run the following command:  
`sass -v`

There should be a response that Sass 3.1.3 (Brainy Betty) is being used.

If a different version is running or there a Sass error is displayed (e.g., “no such file to load”), execute the following steps to correct the Sass install:

- 1) In the Command Prompt window enter “gem uninstall sass.” Then choose which version to uninstall.

*Note: Do not enter the quotes in the uninstall command.*

```
C:\Users\myusername>gem uninstall sass
Select gem to uninstall:
  1. sass-3.1.3
  2. sass-3.1.15
  3. All versions
> 2
Remove executables:
Scss in addition to the gem? [Yn] y
Removing scss
Successfully uninstalled sass-3.1.15
```

- 2) Run the sass -v again; see the example below:

```
C:\Users\myusername>sass -v
Sass 3.1.3 (Brainy Betty)
C:\Users\myusername>
```

## 1.2.4 Verify Tool Installations

To verify each of the tool installations and their versions, simply use the version command of each tool at a Command Prompt. Example commands and output for an OWF 6 environment follow:

- 1) Enter java -version

```
C:\Users\myusername>java -version
java version "1.6.0_32-ea"
Java(TM) SE Runtime Environment (build 1.6.0_32-ea-b03)
Java HotSpot(TM) 64-Bit Server VM (build 20.7-b02, mixed mode)
```

- 2) Enter ant -version

```
C:\Users\myusername>ant -version
Apache Ant(TM) version 1.8.3 compiled on February 26 2012
```

- 3) enter rails -version

```
C:\Users\myusername>rails -version
Welcome to Rails 1.3.7 - http://rails.org/
Licensed under Apache Standard License 2.0
Rails home is set to: C:\Rails\rails-1.3.7
```

- 4) enter groovy-version

```
C:\Users\myusername>groovy -version  
Groovy Version: 1.8.8 JVM: 1.6.0_32
```

5) enter ruby -v

```
C:\Users\myusername>ruby -v  
ruby 1.9.2p290 (2011-07-09) [i386-mingw32]
```

6) enter gem -v

```
C:\Users\myusername>gem -v  
1.8.16
```

7) enter sass -v

```
C:\Users\myusername>sass -v  
Sass 3.1.3 (Brainy Betty)
```

8) enter compass -v

```
C:\Users\myusername>compass -v  
Compass 0.11.7 (Antares)  
Copyright (c) 2008-2012 Chris Eppstein  
Released under the MIT License. Compass is charityware.  
Please make a tax deductible donation for a worthy cause: http://umdf.org/compass
```



## 2 Build-Base

All of the Ozone project build files make use of Ivy (<http://ant.apache.org/ivy>, version 2.1.0) for their transitive dependency management as well as a set of Subversion (SVN) Ant tasks for executing release tasks. Rather than have each project contain duplicate build information pertaining to the installation and configuration of Ivy and the SVN Ant tasks, the build-base directory in the Ozone project trunk contains the build tasks and files that are common across all of the Ozone projects. Build-base doesn't need to be checked out from SVN despite the other projects dependence on it. The Ozone Ivy SVN repository contains a zipped/versioned instance of build-base and each Ozone project treats build-base as just another dependency. The first time an Ozone project is checked out from the GOSS repo, the Ant task 'init-build' needs to be run from within the project directory. This task does the following:

- 1) Retrieves the correct version of the **build-base.zip** file from the Ivy repo and unzips it into the project directory; creating a sub-directory named **build-base-<version>**.
- 2) Runs the 'init-ivy' task which is located in the **build-base.xml** file in the newly installed build-base directory. This task installs the correct version of Ivy as well as the necessary jars for the SVN Ant tasks.

Once 'init-build' has been run, project specific build tasks can be run as described below:

### 2.1 OWF and Server builds

If changes are being made to the server code, build tasks don't normally need to be run to test changes; simply use the standard grails scripts to run or test the server code. The Ant build comes into play when there is the need to build a server bundle, i.e. package the .war files, along with a Tomcat instance into a .zip file that can then be distributed to an end user. To build a server bundle, open a cmd or shell window and 'cd' into the server's base directory. First, make sure the 'init-build' task has been run as described in the section above:

```
C:\owf-server>ant init-build
```

To build the bundle, type the following on the command line and press enter:

```
C:\owf-server>ant bundle
```

The bundle task will do a clean and build of the server code (retrieving any dependencies), run the server tests (both unit and integration) and then build the zipped bundle. The results of the build are written to the staging directory within the server directory. On a successful build the staging directory should contain the zipped bundle,

as well as the unzipped contents of the bundle. The latter is provided as a convenience so that testing can begin the bundle by running the `start.bat` (or `start.sh` on Linux systems) in the staging/`apache-tomcat-x.x.x` directory. This will start the Tomcat server which is initially configured to load the Ozone server and the CAS server .war files.

### 2.1.1 Additional build command line options

When running the build, any of all of the following command line parameters can be used:

- **-logfile build.log** – This will redirect the output of the build to the specified log file. This can be useful when there are problems with the build as the output is often too verbose to be completely captured by the cmd window.
- **-Dno-test=true** – This will prevent the grails unit and integration tests from running. Often, they've already been run, and using this option will speed up bundle build.
- **-Dno-jsdoc=true** – This will prevent the generation of the Javascript documentation. Generating the documentation is time consuming and slows the build progress.
- **-Dgroovy\_all** – This property can be used to manually set the location of `groovy-all.jar`. When building, either to `GROOVY_HOME` environment variable or this property needs to be set correctly. The purpose of this property is to allow building on systems where Groovy was installed in such a way that the `GROOVY_HOME` variable is not set, such as Linux systems that installed Groovy using a centralized package manager. See below for an example:

```
ant bundle -Dgroovy_all=/usr/share/java/groovy-all.jar
```

### 2.1.2 Building Offline

When building the Marketplace or OWF projects on a network that is not externally connected, try the following steps:

- Download the Ozone Ivy repo from:  
<https://owfgoss.org/svn/repos/ozone/ivy-repo>  
This is a very large (~2GB) download.
- Modify application.properties to include the following property:  
**offline\_repo=c:/path/to/ivy-repo/no-namespace**  
This provides the Grails pointer to the offline repo.  
The value must contain the 'no-namespace' portion of the path.

- Add a flag to the Ant command-line: **-Ddisconnected=true**  
This switches the build process to use ivy properties from **build-base/ivysettings\_offline.xml**.
- A version of the **ivysvnresolver\*.jar** needs to be on the Ant classpath. The version found in the following location can be copied and placed under the **ant/lib** directory: **ivy-repo/no-namespace/org.apache.ant-custom/zips/ant-custom-1.7.1.zip/lib**.  
This provides an SvnResolver implementation for Ivy.
- Modify file **build-base/ivysettings\_offline.xml** -- if the token **\${env.OFFLINE\_REPO}** exists, replace with **\${offline\_repo}**  
The **ivy:settings** task has visibility to the declared Ant properties, but not the environmental properties that the parent project has within its scope.
- [Optional] To run any grails command explicitly, add command-line flag:  
**-DOFFLINE\_REPO= c:/path/to/ivy-repo/no-namespace.**

### 2.1.3 Troubleshooting

- Cannot connect to the OWF GOSS SVN download repository?
  - investigate the use of an HTTP Proxy.
  - verify valid GOSS username and password, issued as a Contributor account – the password can be reset through the “Account Management” link on the [owgoss.org](http://owgoss.org) homepage.
- Grails plugins used by the applications have their own dependencies. In some cases these are not required for the base application; e.g. In Alpha/Beta releases these may be experimental or partially built-out capabilities. If a plugin is causing issues, it may be removed by:
  - modifying the **[root]/application.properties** file and comment out the line beginning with **plugins.[pluginName]**.
  - moving the **[root]/plugins/[pluginName]** directory out of the project.

*Note that other references to the plugin may exist in places and this procedure may be insufficient to avoid all possible errors.*

- Testing Grails compilation outside of the Ant script can be useful for diagnosing Grails-level configuration issues without executing all other aspects of the build process. To do this, run the following command from a prompt:  
**grails compile -DOFFLINE\_REPO= c:/path/to/ivy-repo/no-namespace.**

## 2.2 Ozone Security Jar

On the rare occasion that a new release of the security jar file is needed, the following steps should be taken:

- 1) Go the jar's project directory (...\**commons\ozone-security**) and edit the **build.properties** file; changing the jar's version number.
- 2) From the project's cmd line, run the following 2 commands:

```
C:\commons\ozone-security>ant init-build -f owf-build.xml.
```

```
C:\commons\ozone-security>ant pre-release -f owf-build.xml.
```

Because the security project gets distributed with a project zip file, the normal '**build.xml**' is for just building the jar file. The '**owf-build.xml**' file is for building new release versions of the jars. The 'pre-release' target does a full build of the project and then publishes the project's artifact(s) to the local pre-release repo (**.ivy2/local**).

- 3) To test the pre-release version of the jar, go to the directory of either the OWF or Marketplace server project and edit the server **application.properties** file to update the version number of the dependency ('**owf.security.rev**' and '**mp.security.rev**' in the OWF and Marketplace projects, respectively).
- 4) Build the server bundle (ant bundle). Go to the **staging/apache-tomcat-X.X.X/webapps/<server>.war**. Navigate the server war file and go to the **WEB-INF/libs** directory and verify that the new version of the security jar is there. It is best to test the server app as well.
- 5) If there are problems with the new security jar, make necessary modifications and repeat steps 2 thru 4.
- 6) Once satisfied with the new jar, go to the command line and run the release target.

```
C:\commons\ozone-security>ant release -f owf-build.xml
```

*Note: A Subversion 1.6 client must be used. Currently the jars that Ant's Subversion tasks use do not work with code that has been checked out with a 1.7 Subversion client. If a 1.6 client is not used, a vague exception will be thrown inside the ant SVN tasks.*

Details on the release task (which is common to all of our builds) are explained in 3: Building a Release.

## 2.3 CAS Server War

To make a new release of the CAS server war file, do the following:

- 1) Go the CAS server project directory and edit the **build.properties** file; changing the version number (**cas.server.rev**).
- 2) From the project's command line, run the following 2 commands:  

```
C:\ozone\cas-server>ant init-build
```

```
C:\ozone\cas-server>ant pre-release
```

The 'pre-release' target does a full build of the CAS server project and then publishes the project's artifact(s) to the local pre-release repo (**.ivy2/local**)
- 3) To test the pre-release version of the war, go to the directory of server project that uses this dependency (in this case **owf-server**) and edit the server **application.properties** file to update the version number of the cas server (**cas.server.rev**).
- 4) Build the server bundle (ant bundle). Go to the **staging/apache-tomcat-X.X.X /webapps** directory. It should contain the CAS server war file. The file name should include the new CAS server version. Test the changes by running the server application.
- 5) If there are problems with the new war, make edits it and repeat steps 2 thru 4.
- 6) Once satisfied with the new jar, go to the command line and run the release target:

```
C:\ozone\cas-server>ant release
```

Details on the release task (which is common to all of our builds) are explained in [3: Building a Release](#).

## 2.4 Tomcat-OWF-Custom

To make a new release of the **tomcat-owf-custom.zip** file:

- 1) Go the Tomcat server project directory and edit the **build.properties** file; changing the version number (**tomcat.custom.version**).
- 2) From the project's command line, run the following 2 commands:  

```
C:\ozone\tomcat-owf-custom>ant init-build
```

```
C:\ozone\tomcat-owf-custom>ant pre-release
```

The 'pre-release' target does a full build of the **tomcat-owf-custom** zip file and then publishes it the to the *local* pre-release repo (**.ivy2/local**)
- 3) To test the pre-release version of the zip file, go to the directory of the server project that uses this dependency (in this case **owf-server**) and edit the project's

**application.properties** file to update the version number of the CAS server ('**tomcat.custom.rev**').

- 4) Build the server bundle (ant bundle). Go to the **staging/apache-tomcat-x.x.x** directory and verify that the new Tomcat changes are present.
- 5) If there are problems with the new Tomcat, make edits and repeat steps 2 thru 4.
- 6) Once satisfied with the new Tomcat, go to the command line and run the release target:

```
C:\ozone\tomcat-owf-custom>ant release
```

Details on the release task (which is common to all of our builds) are explained in [3: Building a Release](#).

## 3 Building a Release

All of the projects discussed above have an Ant ‘release’ target available to them through the **base-build.xml** file. A release build performs the following steps:

- 1) Cleans and builds the project’s releasable artifacts (bundle, zips, jars, or wars).
- 2) Prompts for the user’s SVN user name and password. In order to perform a release build there must be SVN write access to the Ivy SVN repo.
- 3) Releases a copy of the projects artifact(s) into the Ivy SVN repo.

Prior to running a release build, the following should be done:

- 1) Has the version number of the project being released been properly updated?
- 2) Have the project changes been tested?
- 3) Have all of the changes to the working project been committed to the GOSS repository? If not, the release target will fail with the message “You have uncommitted changes, please commit before doing a release.”

The actual ant release command for a given project has been described prior in the document.

### 3.1 Branching and Tagging

**build-base.xml** supports separate targets for creating a separate SVN branch or tag of a project. Targets should be executed from the project’s working directory. Both targets work in a similar fashion as described below:

- 1) Go to your projects working directory and type in either of “ant branch” or “ant tag”.
- 2) You will be prompted for you SVN user name and password. In order to perform a release build you must have SVN write access to the ivy SVN branch and tag directories.
- 3) You will also be prompted for a version label to append to the branch/tag name.  
*Note: For greater flexibility, these tasks do NOT use the Ivy revision number as part of the branch/tag name, you must supply your own.*
- 4) Both tasks check to see that you have no un-committed changes in your working directory, otherwise the build will fail with the message “You have uncommitted changes, please commit before doing a release.”
- 5) Branches are created in the [‘https://www.owfgoss.org/svn/repos/ozone/baseline/branches’](https://www.owfgoss.org/svn/repos/ozone/baseline/branches) directory of the GOSS repo and tags are created in [‘https://www.owfgoss.org/svn/repos/ozone/baseline/tags’](https://www.owfgoss.org/svn/repos/ozone/baseline/tags).

## Appendix A Contact Information

### **A.1 Discussion Group**

The OZONE Developers Discussion Group is hosted through Google Groups at <http://groups.google.com/group/ozone-developers>. This forum is for the distribution of release announcements, Q&A related to OWF and for additional inquiries about widgets and features being developed across the user base. To access the group, request an invitation at <http://groups.google.com/group/ozone-developers> or contact the Community Support Team at [goss-support@owfgoss.org](mailto:goss-support@owfgoss.org).

### **A.2 Additional POCs**

For information about the OZONE Widget Framework or access to its resources, please email [goss-support@owfgoss.org](mailto:goss-support@owfgoss.org). Additional resources can be found at <http://owfgoss.org>.