

# Ambari Development

## Checking out Ambari source

Follow the instructions under [Checkout source code](#) section of "How to contribute" guide.

We'll refer to the top-level "ambari" directory as `AMBARI_DIR` in this document.

## Tools needed to build Ambari

The following tools are needed to build Ambari from source.

Alternatively, you can easily launch a VM that is preconfigured with all the tools that you need. See the **Pre-Configured Development Environment** section in the [Quick Start Guide](#).

- xCode (if using Mac - free download from the apple store)
  - JDK 8 (Ambari 2.6 and below can be compiled with JDK 7, from Ambari 2.7, it can be compiled with at least JDK 8)
  - [Apache Maven](#) 3.3.9 or later
- Tip: In order to persist your changes to the `JAVA_HOME` environment variable and add Maven to your path, create the following files:  
File: `~/.profile`

```
source ~/.bashrc
```

File: `~/.bashrc`

```
export PATH=/usr/local/apache-maven-3.3.9/bin:$PATH
export JAVA_HOME=$( /usr/libexec/java_home )
export _JAVA_OPTIONS="-Xmx2048m -XX:MaxPermSize=512m -Djava.awt.headless=true"
```

- Python 2.6 (Ambari 2.7 or later require Python 2.7 as minimum supported version)
- Python setuptools:  
for Python 2.6: [Download](#) setuptools and run:

```
sh setuptools-0.6c11-py2.6.egg
```

for Python 2.7: [Download](#) setuptools and run:

```
sh setuptools-0.6c11-py2.7.egg
```

- `rpmbuild` (rpm-build package)
- `g++` (gcc-c++ package)

## Running Unit Tests

- `mvn clean test`
- Run unit tests in a single module:

```
mvn -pl ambari-server test
```

- Run only Java tests:

```
mvn -pl ambari-server -DskipPythonTests
```

- Run only specific Java tests:

```
mvn -pl ambari-server -DskipPythonTests -Dtest=AgentHostInfoTest test
```

- Run only Python tests:

```
mvn -pl ambari-server -DskipSurefireTests test
```

- Run only specific Python tests:

```
mvn -pl ambari-server -DskipSurefireTests -Dpython.test.mask=TestUtils.py test
```

- Run only Checkstyle and RAT checks:

```
mvn -pl ambari-server -DskipTests test
```

NOTE: Please make sure you have npm in the path before running the unit tests.

## Generating Findbugs Report

- mvn clean install

This will generate xml and html report under target/findbugs. You can also add flags to skip unit tests to generate report faster.

## Building Ambari

Note: if you can an error that too many files are open while building, then run: ulimit -n 10000 (for example)

To build Ambari RPMs, run the following.

Note: Replace \${AMBARI\_VERSION} with a 4-digit version you want the artifacts to be (e.g., -DnewVersion=1.6.1.1)

**Note:** If running into errors while compiling the ambari-metrics package due to missing the artifacts of jms, jmxri, jmxtools:

```
[ERROR] Failed to execute goal on project ambari-metrics-kafka-sink: Could not resolve dependencies for project
org.apache.ambari:ambari-metrics-kafka-sink:jar:2.0.0-0: The following artifacts could not be resolved: javax.jm
s:jms:jar:1.1, com.sun.jdmk:jmxtools:jar:1.2.1, com.sun.jmx:jmxri:jar:1.2.1: Could not transfer artifact javax.j
ms:jms:jar:1.1 from/to java.net (https://maven-repository.dev.java.net/nonav/repository): No connector available
to access repository java.net (https://maven-repository.dev.java.net/nonav/repository) of type legacy using the
available factories WagonRepositoryConnectorFactory
```

The work around is to manually install the three missing artifacts:

```
mvn install:install-file -Dfile=jms-1.1.pom -DgroupId=javax.jms -DartifactId=jms -Dversion=1.1 -Dpackaging=jar
mvn install:install-file -Dfile=jmxtools-1.2.1.pom -DgroupId=com.sun.jdmk -DartifactId=jmxtools -Dversion=1.2.1 -
Dpackaging=jar
mvn install:install-file -Dfile=jmxri-1.2.1.pom -DgroupId=com.sun.jmx -DartifactId=jmxri -Dversion=1.2.1 -
Dpackaging=jar
```

If when compiling it seems stuck, and you've already increased Java and Maven heapsize, it could be that Ambari Views has a lot of artifacts, and the rat-check is choking up. In this case, try running

```
git clean -df (this will remove untracked files and directories)
mvn clean package -DskipTests -Drat.ignoreErrors=true
or
mvn clean package -DskipTests -Drat.skip
```

## Setting the Version Using Maven

Ambari 2.8+ uses a newer method to update the version when building Ambari.

#### **RHEL/CentOS 6:**

```
# Update the revision property to the release version
mvn versions:set-property -Dproperty=revision -DnewVersion=2.8.0.0.0

mvn -B clean install package rpm:rpm -DskipTests -Dpython.ver="python >= 2.6" -Preplaceurl
```

#### **SUSE/SLES 11**

```
# Update the revision property to the release version
mvn versions:set-property -Dproperty=revision -DnewVersion=2.8.0.0.0

mvn -B clean install package rpm:rpm -DskipTests -Psuse11 -Dpython.ver="python >= 2.6" -Preplaceurl
```

#### **Ubuntu 12:**

```
# Update the revision property to the release version
mvn versions:set-property -Dproperty=revision -DnewVersion=2.8.0.0.0

mvn -B clean install package jdeb:jdeb -DskipTests -Dpython.ver="python >= 2.6" -Preplaceurl
```

#### **Ambari 2.7 and Earlier Releases (Deprecated)**

#### **RHEL/CentOS 6:**

```
mvn versions:set -DnewVersion=${AMBARI_VERSION}

#Note: The ambari-metrics project is not wired up to the main ambari project. However there is a dependency on
ambari-metrics-common to build the ambari-server RPM.
#Hence you also need to set ambari-metrics project version as well.
pushd ambari-metrics
mvn versions:set -DnewVersion=${AMBARI_VERSION}
popd

mvn -B clean install package rpm:rpm -DskipTests -Dpython.ver="python >= 2.6" -Preplaceurl
```

#### **SUSE/SLES 11**

```
mvn versions:set -DnewVersion=${AMBARI_VERSION}

#Note: The ambari-metrics project is not wired up to the main ambari project. However there is a dependency on
ambari-metrics-common to build the ambari-server RPM.
#Hence you also need to set ambari-metrics project version as well.
pushd ambari-metrics
mvn versions:set -DnewVersion=${AMBARI_VERSION}
popd

mvn -B clean install package rpm:rpm -DskipTests -Psuse11 -Dpython.ver="python >= 2.6" -Preplaceurl
```

## Ubuntu 12:

```
mvn versions:set -DnewVersion=${AMBARI_VERSION}

#Note: The ambari-metrics project is not wired up to the main ambari project. However there is a dependency on
ambari-metrics-common to build the ambari-server RPM.
#Hence you also need to set ambari-metrics project version as well.
pushd ambari-metrics
mvn versions:set -DnewVersion=${AMBARI_VERSION}
popd

mvn -B clean install package jdeb:jdeb -DskipTests -Dpython.ver="python >= 2.6" -Preplaceurl
```

Ambari Server will create following packages

- RPM will be created under `AMBARI_DIR/ambari-server/target/rpm/ambari-server/RPMS/noarch`.
- DEB will be created under `AMBARI_DIR/ambari-server/target/`

Ambari Agent will create following packages

- RPM will be created under `AMBARI_DIR/ambari-agent/target/rpm/ambari-agent/RPMS/x86_64`.
- DEB will be created under `AMBARI_DIR/ambari-agent/target`

Optional parameters:

- `-X -e`: add these options for more verbose output by Maven. Useful when debugging Maven issues.
- `-DdefaultStackVersion=STACK-VERSION`
- Sets the default stack and version to be used for installation (e.g., `-DdefaultStackVersion=HDP-1.3.0`)
- `-DenableExperimental=true`
- Enables experimental features to be available via Ambari Web (default is false)
- All views can be packaged in RPM by adding `-Dviews` parameter
  - `mvn -B clean install package rpm:rpm -Dviews -DskipTests`
- Specific views can be built by adding `--projects` parameter to the `-Dviews`
  - `mvn -B clean install package rpm:rpm --projects ambari-web,ambari-project,ambari-views,ambari-admin,contrib/views/files,contrib/views/pig,ambari-server,ambari-agent,ambari-client,ambari-shell -Dviews -DskipTests`

*NOTE: Run everything as root below.*

## Building Ambari Metrics

If you plan on installing the Ambari Metrics service, you will also need to build the Ambari Metrics project.

```
cd ambari-metrics
mvn clean package -Dbuild-rpm -DskipTests

For Ubuntu:
cd ambari-metrics
mvn clean package -Dbuild-deb -DskipTests
```

### Note:

The metrics rpms will be found at: `ambari-metrics-assembly/target/`. These would be need for installing the Ambari Metrics service.

## Running the Ambari Server

First, install the Ambari Server RPM.

### On RHEL/CentOS:

```
yum install ambari-server/target/rpm/ambari-server/RPMS/noarch/ambari-server-*.noarch.rpm
```

On SUSE/SLES:

```
zypper install ambari-server/target/rpm/ambari-server/RPMS/noarch/ambari-server-*.noarch.rpm
```

#### On Ubuntu 12:

```
dpkg --install ambari-server/target/ambari-server-*.deb          # Will fail with missing dependencies errors
apt-get update                                                  # Update locations of dependencies
apt-get install -f                                              # Install all failed dependencies
dpkg --install ambari-server/target/ambari-server-*.deb        # Will succeed
```

#### Initialize Ambari Server:

```
ambari-server setup
```

#### Start up Ambari Server:

```
ambari-server start
```

#### See Ambari Server log:

```
tail -f /var/log/ambari-server/ambari-server.log
```

#### To access Ambari, go to

```
http://{ambari-server-hostname}:8080
```

from your web browser and log in with username *admin* and password *admin*.

## Install and Start the Ambari Agent Manually on Each Host in the Cluster

Install the Ambari Agent RPM.

#### On RHEL/CentOS:

```
yum install ambari-agent/target/rpm/ambari-agent/RPMS/x86_64/ambari-agent-*.rpm
```

#### SUSE/SLES:

```
zypper install ambari-agent/target/rpm/ambari-agent/RPMS/x86_64/ambari-agent-*.rpm
```

#### Ubuntu12:

```
dpkg --install ambari-agent/target/ambari-agent-*.deb
```

Edit the location of Ambari Server in `/etc/ambari-agent/conf/ambari-agent.ini` by editing the *hostname* line.

#### Start Ambari Agent:

```
ambari-agent start
```

#### See Ambari Agent log:

```
tail -f /var/log/ambari-agent/ambari-agent.log
```

# Setting up Ambari in Eclipse

```
$ mvn clean eclipse:eclipse
```

After doing the above you should be able to import the project via Eclipse "Import > Maven > Existing Maven Project". Choose the root directory where you cloned the git repository. You should be able to see the following projects on eclipse:

```
ambari
|
|- ambari-project
|- ambari-server
|- ambari-agent
|- ambari-web
```

Select the top-level "ambari pom.xml" and click Finish.

[Coding Guidelines for Ambari](#)