Table of contents

Runs a Grails application as a JAR file with an embedded Tomcat or Jetty server

Grails Standalone Plugin - Reference Documentation

Authors: Burt Beckwith

Version: 1.1

Table of Contents

1 Introduction to the Standalone Plugin

The Standalone plugin builds a runnable JAR file with an embedded war built from your application and an embedded Tomcat 7 or Jetty 7 instance. This allows you to build a single archive that can be run on any computer with Java 5 or higher by running `java -jar standalone.jar`. This can be convenient for demos or even very lightweight installs of low-traffic Grails applications.

Release History

- August 6, 2012
  - 1.1 release
- August 4, 2012
  - 1.0.1 release
- July 8, 2011
  - initial 1.0 release

2 Running the application

Building the jar

The first step is to run the build-standalone script, e.g.

```
grails prod build-standalone
```

or

```
grails -Dgrails.env=demo build-standalone our_cool_demo.jar
```

If you pass the --jetty flag the embedded server will be Jetty instead of the default Tomcat, for example

```
grails prod build-standalone --jetty
```

or

```
grails -Dgrails.env=demo build-standalone our_cool_demo.jar --jetty
```

Running the server

As long as the target machine has Java 5 or higher available, all you need to do next is run

```
java -jar /path/to/jar_name.jar
```

There are a few arguments you can pass to customize how the application runs:

1. context path; if not specified it defaults to "" (the "root" context)
2. host name; if not specified it defaults to "localhost"
3. HTTP port; if not specified it defaults to 8080
4. HTTPS port; there is no default for this, but if specified you can also specify the keystore path and password
5. SSL keystore path; if not specified a temporary keystore will be generated
6. SSL keystore password; required if an existing keystore path is specified

So the full syntax would be:

```
java -jar [jar path] [context path] [host name] [HTTP port] [HTTPS port]
          [SSL keystore path] [SSL keystore password]
```

For example running

```
java -jar /path/to/jar_name.jar
```

will start a server at `http://localhost:8080/` and

```
java -jar /path/to/jar_name.jar cool_demo localhost 9000
```

will start a server at `http://localhost:9000/cool_demo`

```
java -jar /path/to/jar_name.jar cool_demo localhost 8080 8443
```

will start a server at `http://localhost:8080/cool_demo` and will also support SSL at `https://localhost:8443/cool_demo`