Build Scala Project using sbt and Jenkins

Agenda

- **Scala - A Scalable language**
- **Scala Download**
- **Scala Software Requirement**
- **Scala IDEs**
- **Scala Install Configuration**
- **Scala First Program**
- **Compile and Run**
- **Building Scala Projects using Jenkins**
- **Sbt downloand and configure**

Scala - A Scalable language

Scala is an acronym for “Scalable Language”. This means that Scala grows with you. You can play with it by typing one-line expressions and observing the results. But you can also rely on it for large mission critical systems, as many companies, including Twitter, LinkedIn, or Intel do.

To some, Scala feels like a scripting language. Its syntax is concise and low ceremony; its types get out of the way because the compiler can infer them. There’s a REPL and IDE worksheets for quick feedback. Developers like it so much that Scala won the ScriptBowl contest at the 2012 JavaOne conference.

At the same time, Scala is the preferred workhorse language for many mission critical server systems. The generated code is on a par with Java’s and its precise typing means that many problems are caught at compile-time rather than after deployment.

At the root, the language’s scalability is the result of a careful integration of object-oriented and functional language concepts.

- Object-Oriented

- Functional
- Seamless Java Interop
- Functions are Objects
- Future-Proof
- Fun

Scala Download

Download and Install from:

http://www.scala-lang.org/download/

Scala Software Requirement

Java Runtime Version 1.6 or later

http://www.java.com/en/

Scala IDEs

http://scala-ide.org/?_ga=1.244227205.1414599726.1425237373

Scala Install Configuration

Environment	Variable	**Value (example)**
Unix	`$SCALA_HOME` `$PATH`	`/usr/local/share/scala` `$PATH:$SCALA_HOME/bin`

Windows	`%SCALA_HOME%`
	`%PATH%`
		`c:\Progra~1\Scala`
		`%PATH%;%SCALA_HOME%\bin`

Scala First Program

As a first example, we use the standard “Hello, world” program to demonstrate the use of the Scala tools without knowing too much about the language.

```scala
object HelloWorld {
  def main(args: Array[String]) {
    println("Hello, world!")
  }
}
```

The structure of this program should be familiar to Java programmers: it consists of the method `main` which prints out a friendly greeting to the standard output.

Run it interactively!

The `scala` command starts an interactive shell where Scala expressions are interpreted interactively.

```
> scala
This is a Scala shell.
Type in expressions to have them evaluated.
Type :help for more information.

scala> object HelloWorld {
     |   def main(args: Array[String]) {
     |     println("Hello, world!")
     |   }
     | }
defined module HelloWorld

scala> HelloWorld.main(null)
Hello, world!

scala>:q
>
```

The shortcut `:q` stands for the internal shell command `:quit` used to exit the interpreter.

Compile it!

The `scalac` command compiles one (or more) Scala source file(s) and generates Java bytecode which can be executed on any standard JVM. The Scala compiler works similarly to `javac`, the Java compiler of the Java SDK.

```
> scalac HelloWorld.scala
```

By default `scalac` generates the class files into the current working directory. You may specify a different output directory using the `-d` option.

```
> scalac -d classes HelloWorld.scala
```

Execute it!

The `scala` command executes the generated bytecode with the appropriate options:

```
> scala HelloWorld
```

`scala` allows us to specify command options, such as the `-classpath` (alias `-cp`) option:

```
> scala -cp classes HelloWorld
```

The argument of the `scala` command has to be a top-level object. If that object extends trait `App`, then all statements contained in that object will be executed; otherwise you have to add a method `main` which will act as the entry point of your program.
Here is how the “Hello, world” example looks like using the `App` trait:

```
object HelloWorld extends App {
  println("Hello, world!")
}
```

Script it!

We may also run our example as a shell script or batch command (see the examples in the man pages of the `scala` command).
The bash shell script `script.sh` containing the following Scala code (and shell preamble)

```
#!/bin/sh
exec scala "$0" "$@"
!#
object HelloWorld extends App {
  println("Hello, world!")
}
HelloWorld.main(args)
```

can be run directly from the command shell:

```
> ./script.sh
```

Note: We assume here that the file `script.sh` has execute access and the search path for the `scala` command is specified in the `PATH` environment variable.

Building Scala Projects using Jenkins

You would require following…

scala (Build Machine)
Install and configure as mentioned above.

Sbt (Build Machine)
sbt is a build tool for Scala, java, and more.
Download - http://www.scala-sbt.org/download.html
Install - http://www.scala-sbt.org/0.12.4/docs/Getting-Started/Setup.html
Document - http://www.scala-sbt.org/documentation.html

sbt-plugins (Jenkins)
This plugins allows building Scala projects using sbt.
https://wiki.jenkins-ci.org/display/JENKINS/sbt+pluginsbt

scala plugins for Jenkins (Jenkins)
The plugin offers the facility to Install and execute Scala scripts as a build step:

- Scala Installer (available in 'Manage Jenkins')
- Scala Forked Executer (available as a Build Step in Jobs)
- Scala In-VM Executer (available as a Build Step in Jobs)

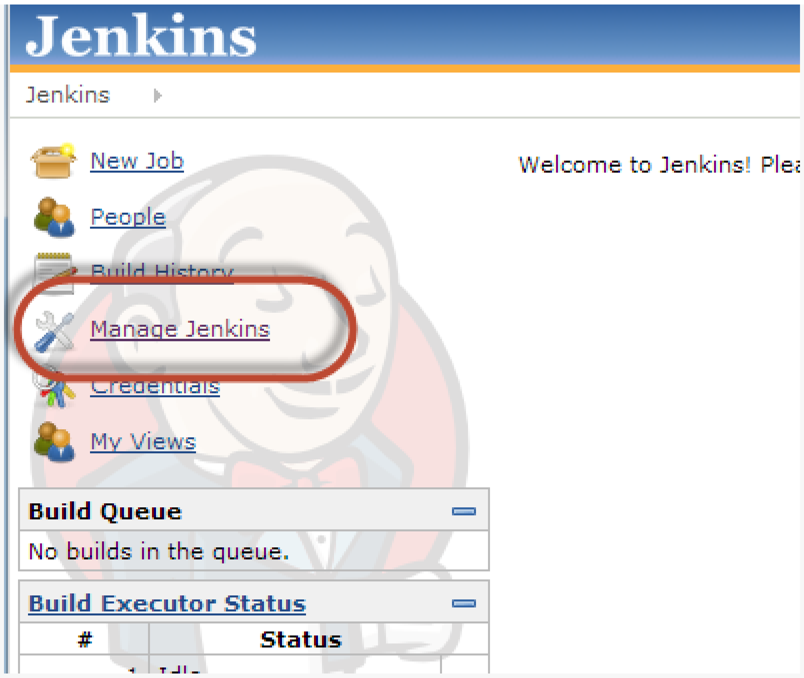
https://github.com/adamretter/jenkins-scala-plugin

Configure the sbt plugin

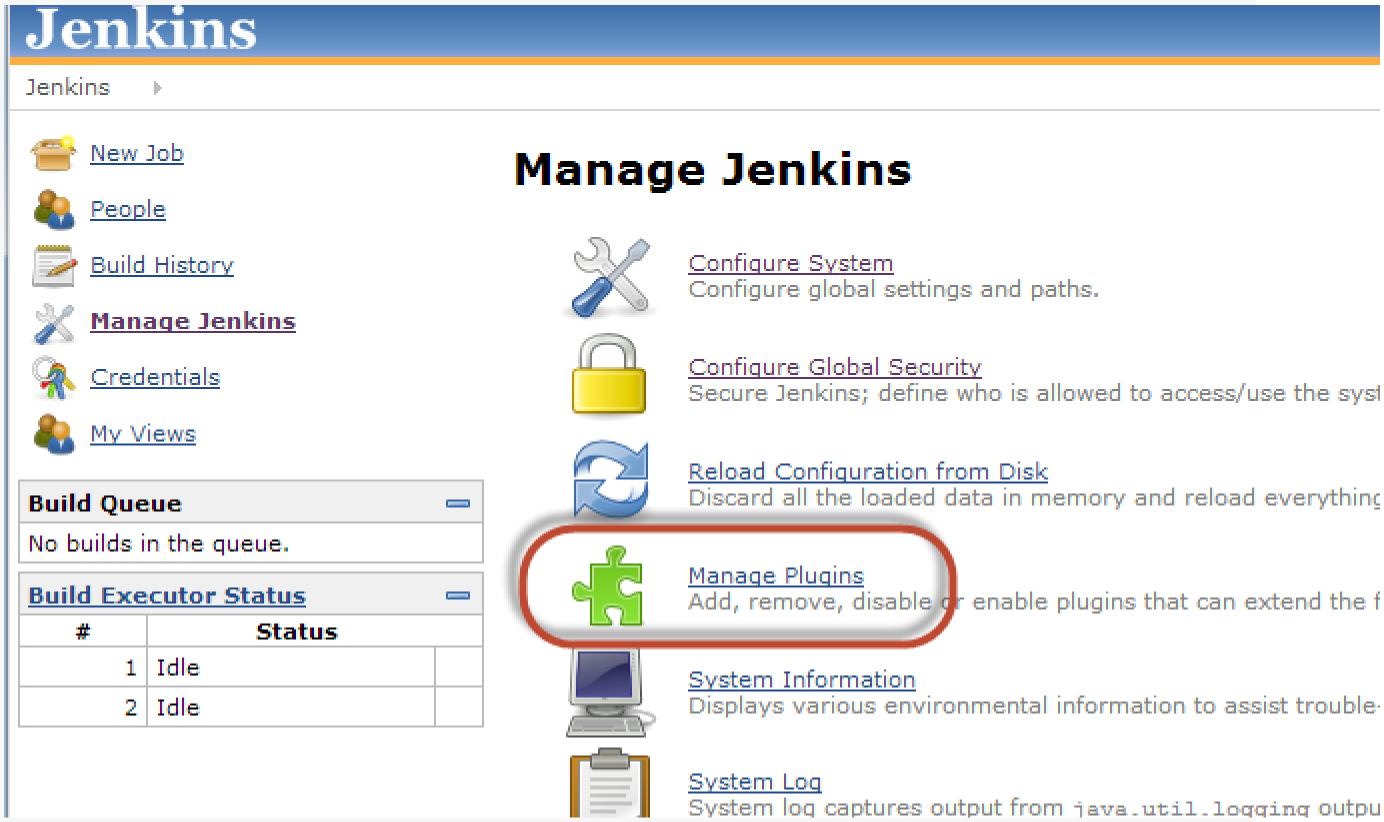
Install plug in

Open your Jenkins install in a web browser. Mine happens to be located at http://192.168.0.7:8080/

After you login click on Manage Jenkins



Click on Manage Plugins



#	Status
1	Idle
2	Idle

Manage Jenkins

Configure System
Configure global settings and paths.

Configure Global Security
Secure Jenkins; define who is allowed to access/use the syst

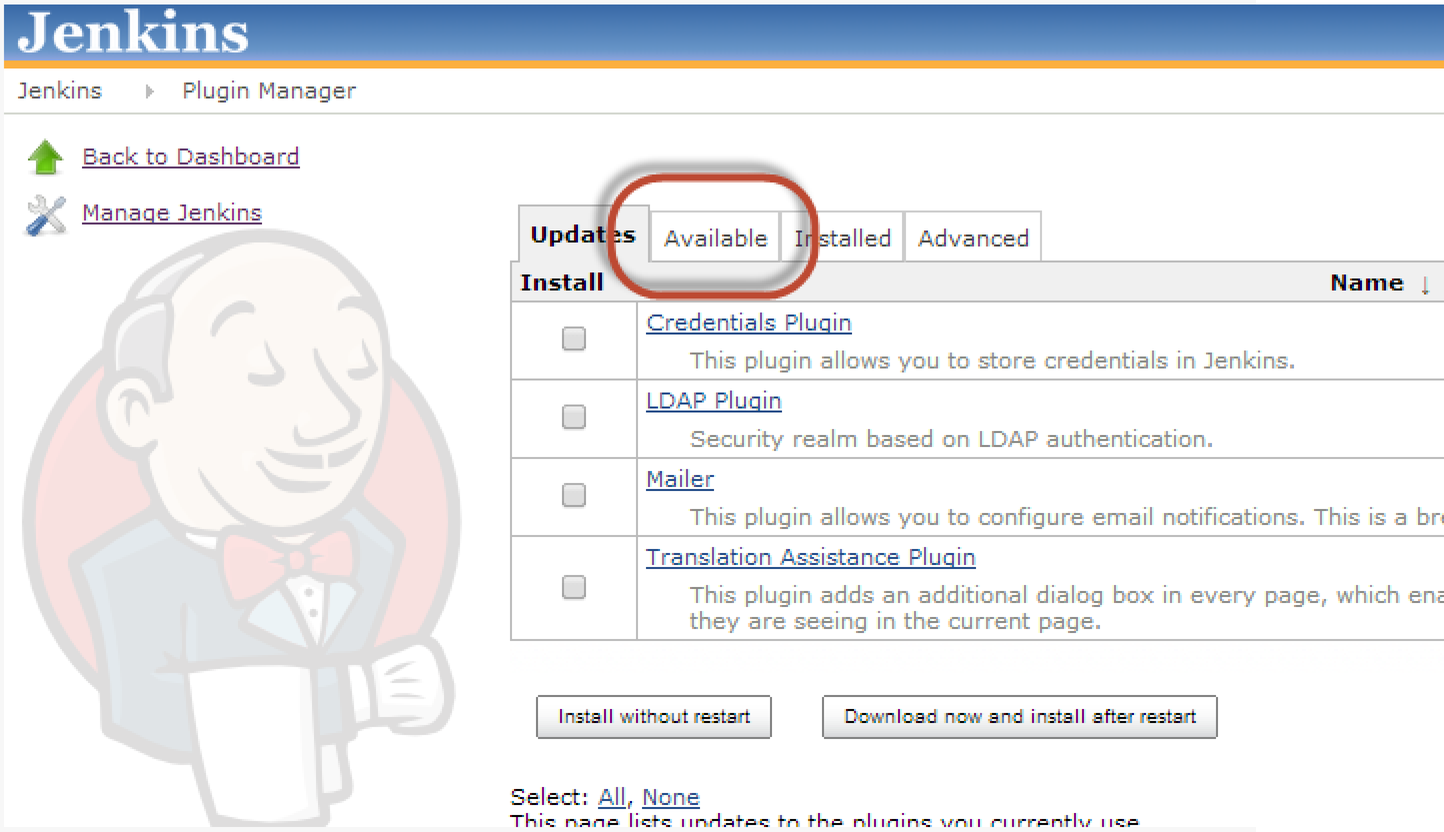
Reload Configuration from Disk
Discard all the loaded data in memory and reload everything

Manage Plugins
Add, remove, disable or enable plugins that can extend the f

System Information
Displays various environmental information to assist trouble-

System Log
System log captures output from java.util.logging outpu

Click on the Available Tab

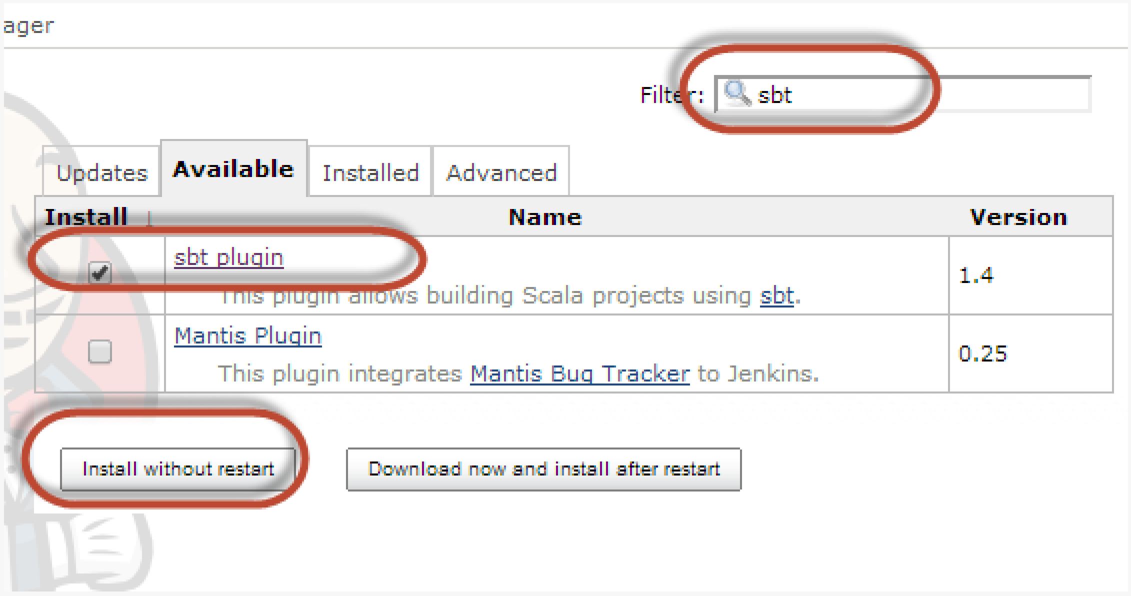


Install	Name
☐	Credentials Plugin This plugin allows you to store credentials in Jenkins.
☐	LDAP Plugin Security realm based on LDAP authentication.
☐	Mailer This plugin allows you to configure email notifications. This is a br
☐	Translation Assistance Plugin This plugin adds an additional dialog box in every page, which ena they are seeing in the current page.

Select: All, None
This page lists updates to the plugins you currently use

In the filter enter "sbt"

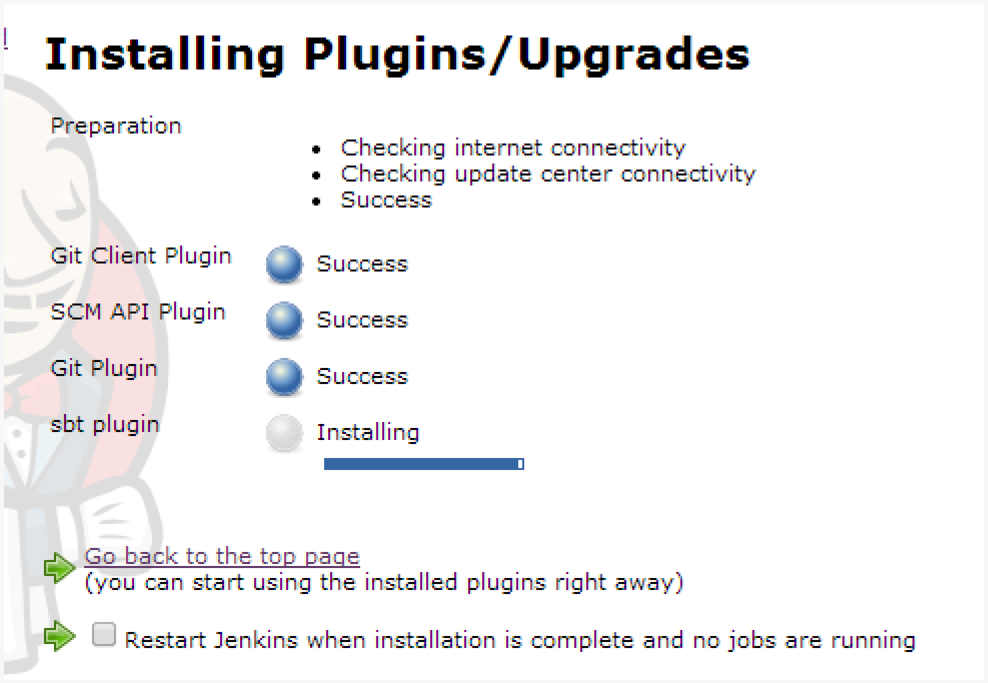
Then select "sbt Plugin" and finally click on "Install without restart"



Filter: sbt

Install ↓	Name	Version
✓	sbt plugin This plugin allows building Scala projects using sbt.	1.4
☐	Mantis Plugin This plugin integrates Mantis Bug Tracker to Jenkins.	0.25

Plug-in installation starts



Installing Plugins/Upgrades

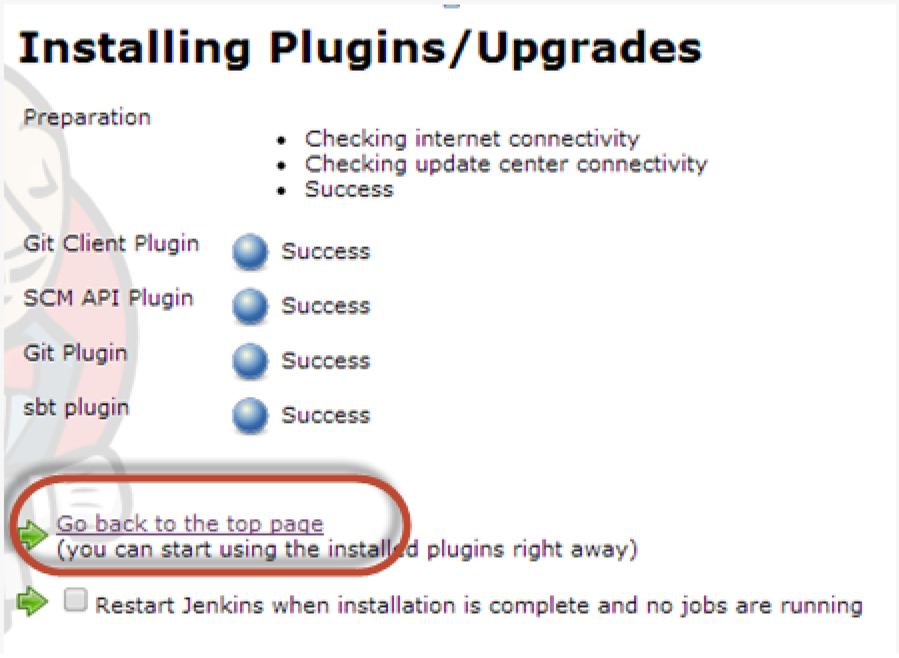
Preparation
- Checking internet connectivity
- Checking update center connectivity
- Success

Git Client Plugin	Success
SCM API Plugin	Success
Git Plugin	Success
sbt plugin	Installing

Go back to the top page
(you can start using the installed plugins right away)

Restart Jenkins when installation is complete and no jobs are running

After the plug-in is successfully installed click on "Go back to the top page"



Installing Plugins/Upgrades

Preparation
- Checking internet connectivity
- Checking update center connectivity
- Success

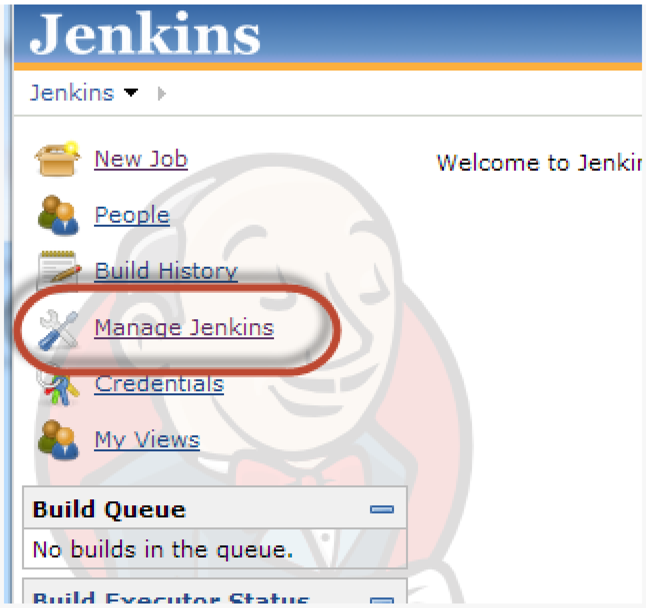
Git Client Plugin	Success
SCM API Plugin	Success
Git Plugin	Success
sbt plugin	Success

Go back to the top page
(you can start using the installed plugins right away)

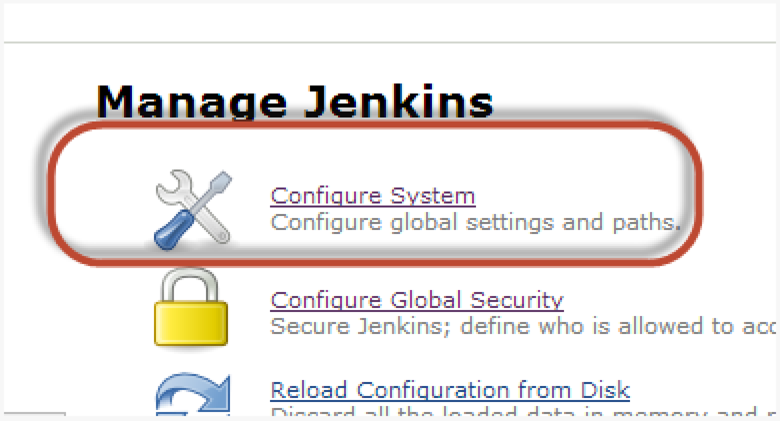
Restart Jenkins when installation is complete and no jobs are running

CONFIGURE JENKINS TO USE SBT LAUNCH JARS

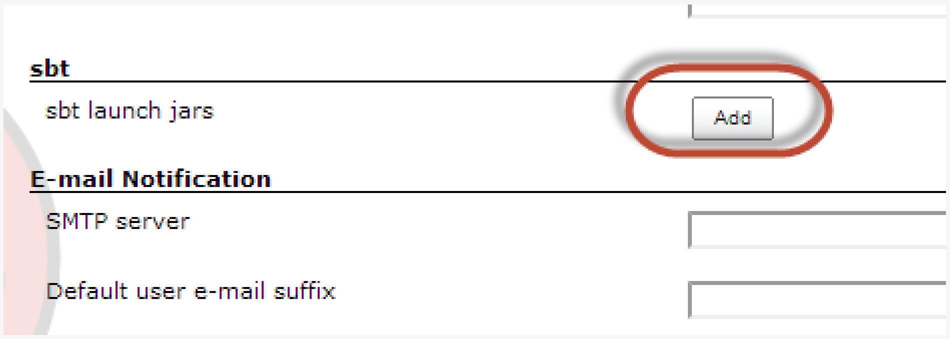
Click on Manage Jenkins



Click on Configure system


Manage Jenkins
Configure System
Configure global settings and paths.
Configure Global Security
Secure Jenkins; define who is allowed to acc
Reload Configuration from Disk

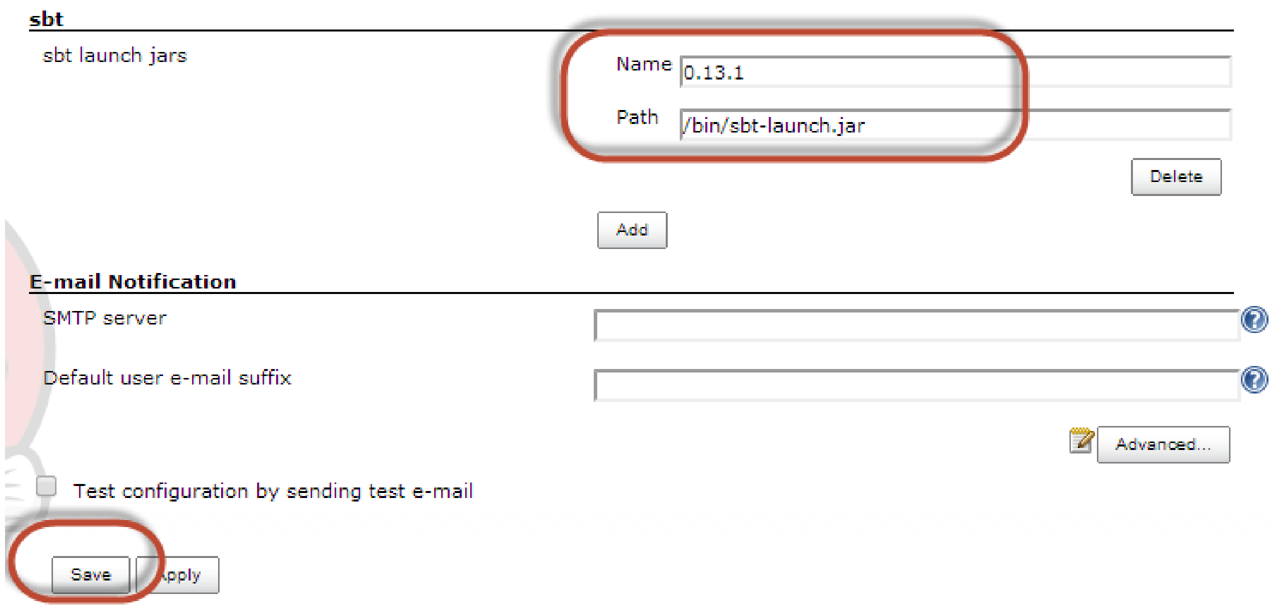
Scroll down to sbt and click Add



Give it a name (I named it after the sbt-launch version 0.13.1)

And set the path to the sbt-launch.jar location.

Click Save



sbt

sbt launch jars

Name 0.13.1

Path /bin/sbt-launch.jar

E-mail Notification

SMTP server

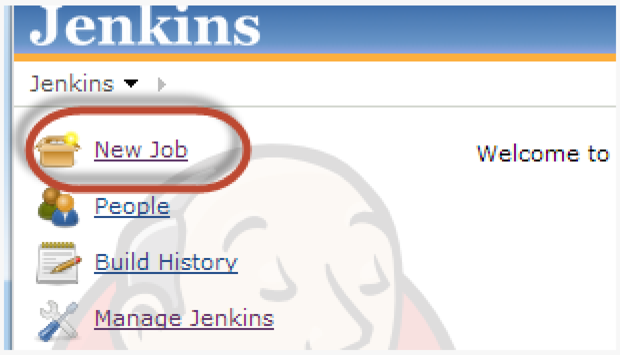
Default user e-mail suffix

Test configuration by sending test e-mail

In order to set up sbt-plugin, you need to specify the names and locations of one or more sbt launch jars. Press the **Manage Jenkins** link and then the **Configure System**. You should now see the sbt configuration section where you will be asked to specify names and locations for your sbt launch jars.

Add build step to your project

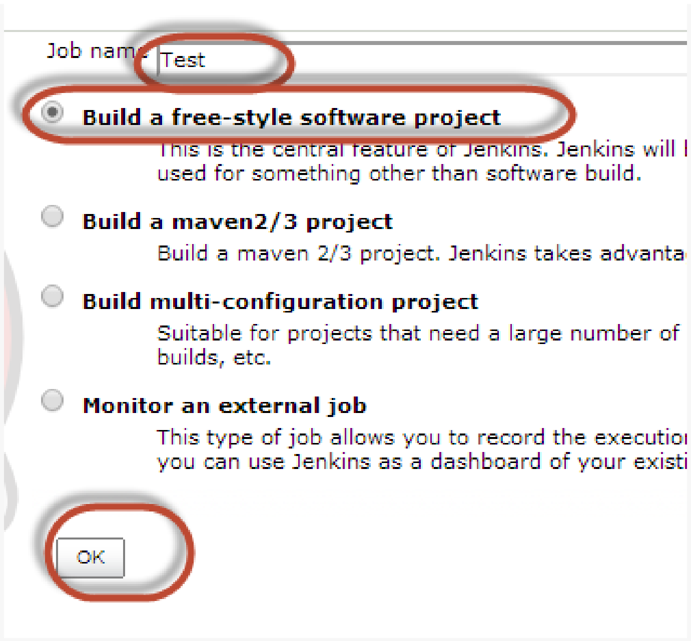
Click on New Job



Give the job a name

Select "Build a free-style software project"

Click OK



Job name: Test

● Build a free-style software project
This is the central feature of Jenkins. Jenkins will
used for something other than software build.

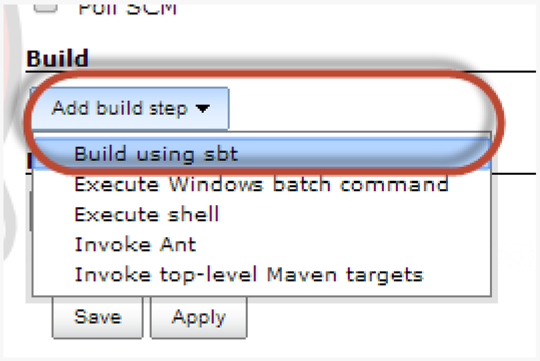
○ Build a maven2/3 project
Build a maven 2/3 project. Jenkins takes advanta

○ Build multi-configuration project
Suitable for projects that need a large number of
builds, etc.

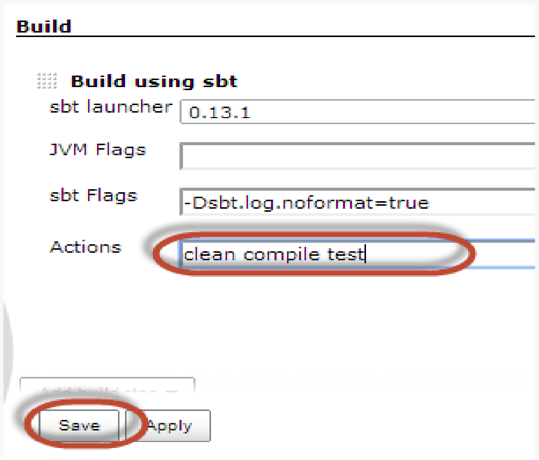
○ Monitor an external job
This type of job allows you to record the executio
you can use Jenkins as a dashboard of your existi

Scroll down and click "add build step"

And select "Build using sbt"



Enter your action and click Save



Build

Build using sbt

sbt launcher: 0.13.1

JVM Flags:

sbt Flags: -Dsbt.log.noformat=true

Actions: clean compile test

Reference:

Scala Official Website - **http://www.scala-lang.org/what-is-scala.html**

Scala Documentation - **http://www.scala-lang.org/documentation/**

Slideshare PPT

http://www.slideshare.net/knoldus/scalas-evolving-ecosystem-introduction-to-scalajs?qid=68228958-aeaf-4220-80f6-c24656c1e720&v=default&b=&from_search=1

http://www.slideshare.net/scalaconfjp/the-evolution-of-scala-scala?qid=68228958-aeaf-4220-80f6-c24656c1e720&v=default&b=&from_search=9