

# SCons Plugin

## Plugin Information

View SCons [on the plugin site](#) for more information.



This plugin is up for adoption. Want to help improve this plugin? [Click here to learn more!](#)

This plugin allows Hudson to invoke [SCons](#) build script as the main build step.

## Configurations

### Project Configuration

The plugin works much like Ant builder, and using it is very easy. You basically just need to specify the SCons options, variables and targets build script that you want to run:

Job name

- Build a maven2 project**  
Build a maven2 project. Hudson takes advantage of your POM files and drastically reduces the configuration.
- Monitor an external job**  
This type of job allows you to record the execution of a process run outside Hudson, even on a remote machine. This is designed so that you can use Hudson as a dashboard of your existing automation system. See [the documentation for more details](#).
- Build a free-style software project**  
This is the central feature of Hudson. Hudson will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
- Build multi-configuration project (alpha)**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

**Build**

Add build step ▾

- Execute shell
- Invoke top-level Maven targets
- Execute Windows batch command
- Invoke Ant
- Invoke scons script**
- Aggregate downstream test results

? ?

**Build**

Invoke scons script ?

Generate and invoke a scons command.  
The command synopsis is `scons [options] [name=val... ] [targets]`  
scons searches for a file named SConstruct, Sconscript, or sconscript (in that order) in the current directory and reads its configuration from the first file found.

SCons Version  ▾

Options  ▾ ?

Specify the scons options.

Variables  ▾ ?

Specify the scons variables.

Targets  ▾ ?

Specify a list of scons targets to be invoked, or leave it empty to invoke the default scons target specified in the scons script.

SConscript root directory  ?

Root Sconscript directory, i.e `-C rootDirectory`; this option can be used to use build files in a subdirectory.

SConscript file  ?

If your build requires a custom file `-f newSconscript`, specify it here. By default, scons searches for a file named SConstruct, Sconscript, or sconscript (in that order) in the root directory; this option can be used to use build files with a different name

## Global Configuration

If 'scons' is not in PATH, or if you have multiple SCons installations and need to be able to configure which project uses which SCons, then you can go to the system configuration and tell Hudson where your Gradle are:

List of SCons installations on this system

---

**Scons**

SCons installation	name	<input type="text" value="scons-1.2.0"/>
	SCons Executable Path	<input type="text" value="C:\integ\scons-1.2.0"/>

List of SCons installations on this system

## Changelog

### Release 0.4 (May 24, 2011)

\* Fixed [JENKINS-9755](#) - SCons-Plugin does not work with current Jenkins release (1.413)

### Release 0.3 (March 06, 2011)

\* Upgraded to Jenkins 1.399 metadata

### Release 0.2.1 (July 07, 2010)

\* Fixed [JENKINS-5791](#)

### Release 0.2 (January 28, 2010)

\* Added the ability to provide a SCons script content without a SCons file.

NOTE: the config.xml section name changed in this release. SConsBuilderScriptFile replaced SConsBuilder, and you will have to manually change the section name in your existing configuration files.

### Release 0.1 (April 08, 2009)

\* Initial