FREE eBook

LEARNING SCons

Free unaffiliated eBook created from
Stack Overflow contributors.

#scons

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: scons

It is an unofficial and free SCons ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official SCons.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with SCons

Remarks

SCons is a build system. It takes a bunch of input files and run tools on them to produce output. SCons is written in pure Python, works the same way on Linux, Windows and OS X, and may be run without installation.

SCons' `SConstruct` files are Python scripts with built-in commands that create a `build tree`. SCons executes build process in phases. First is reading files and constructing a build tree. Second is traversing the tree to build target files.

Versions

Version	Release Date
2.5.1	2017-11-03

Examples

Getting Started

Once you have SCons running, create a file named `SConstruct`:

```
print('..Building World')
```

Now run `scons`:

```
$ scons
scons: Reading SConscript files ...
..Building World
scons: done reading SConscript files.
scons: Building targets ...
scons: `.' is up to date.
scons: done building targets.
```

`SConstruct` is a Python script with additional SCons functions.

```
Zip('archive', ['SConstruct'])
```

The script above packs itself into `archive.zip` using `Zip()` function provided by SCons. `Zip` is a **Builder** - it builds **target** specified by first argument from multiple **sources**, which come as second argument to Builders by convention.

SCons **Builders** start with uppercase letter and operate on **Environment** object, which stores

build configuration. SCons provides default **Environment**, but it can be created explicitly to separate build variables, choose different tools, etc.

```
env = Environment()
env.Zip('archive', ['SConstruct'])
```

Note that when you run the script for the second time, it doesn't build anything. SCons rebuilds targets only when source files change. Modify `SConstruct` and run `scons` again to see the difference.

SCons is designed to be extensible. You add your own Builder methods by attaching them to the Environment, which can be covered in later topics.

Read Getting started with SCons online: https://riptutorial.com/scons/topic/4780/getting-started-with-scons

Chapter 2: C++

Examples

A Simple Build

It is very easy to build a simple C++ project. Here is an example of a `SConstruct` file that does so:

```
env=Environment()

env.Program('hello', Glob('src/*.cpp'))
```

This creates the executable `hello` composed of all the sources in `src` with extension `cpp`.

Specifying Various Build Options

This example shows more detailed build settings:

```
env=Environment(
    CPPPATH='/usr/include/boost/',
    CPPDEFINES=['foo'],
    LIBS=['bar'],
    SCONS_CXX_STANDARD='c++11')

env.Program('hello', Glob('src/*.cpp'))
```

This builds the executable `hello` from all the `cpp` files in `src`, with the following settings:

- The search path is `/usr/include/boost'
- The constant `FOO` is defined
- The executable links with `bar`
- C++11 is used as a standard

Read C++ online: https://riptutorial.com/scons/topic/6158/cplusplus

Chapter 3: Getting SCons running

Introduction

SCons is written in Python 2 and doesn't need any dependencies to work. You can just copy its scripts to your project source tree and run from here. Or you may want to use version packaged for your operating system.

Examples

Installing on Linux

On Debian or Ubuntu, you can install SCons using

```
$ sudo apt-get install scons
```

On YUM-based systems, use

```
$ sudo yum install scons
```

You can install using an RPM by downloading it, then running

```
$ sudo rpm -Uvh http://prdownloads.sourceforge.net/scons/scons-2.5.0-1.noarch.rpm
```

Installing on Windows

Grab installer from http://scons.org/pages/download.html

Or try `pip` installation tool that comes with Python:

```
pip install scons
```

If `scons` still can't be found after that, make sure that Python `Scripts/` folder is added to `PATH` for your Python installation.

Running from source

If you have modifications to share or just want to try new version in development.

```
$ hg clone https://bitbucket.org/scons/scons
$ python scons/src/script/scons.py
```

Installing with Python pip

```
pip install scons
```

If you are not to run `scons` from command line, check that Python scripts directory is added to PATH for your installation.

If you want to play with API, `import SCons` from Python won't work, because SCons 2.5.x and below allows to install multiple versions side-by-side. This was needed to switch between different SCons versions during development and troubleshooting. Now the more common way for this is to use `virtualenv` or just run it from source.

Read Getting SCons running online: https://riptutorial.com/scons/topic/9377/getting-scons-running

Chapter 4: SCons run phases

Introduction

SCons is a multi-step build system. First it reads all `SConstruct` and `SConscript` to execute Python code and create `build graph` with targets. Then it scans filesystem to detect which targets from the `build graph` should be updated, and after that it executes command to build outdated targets.

Examples

Inspecting SCons phases

`scons` describes running phases itself. Running it over an empty `SConstruct` yields this:

```
$ scons
scons: Reading SConscript files ...
scons: done reading SConscript files.
scons: Building targets ...
scons: `.' is up to date.
scons: done building targets.
```

To suppress phase messages, add `-Q` option. `--tree=all` allows to see dependency tree for current target that `scons` constructed while building.

```
$ scons -Q --tree=all
scons: `.' is up to date.
+-.
  +-SConstruct
```

. is default target, which means "build SConstruct in current directory". `SConstruct` is then a dependency for building the default target.

Read SCons run phases online: https://riptutorial.com/scons/topic/10170/scons-run-phases

Credits

S. No	Chapters	Contributors
1	Getting started with SCons	Ami Tavory, anatoly techtonik, bdbaddog, Community
2	C++	Ami Tavory
3	Getting SCons running	Ami Tavory, anatoly techtonik
4	SCons run phases	anatoly techtonik