eBook Gratuit

APPRENEZ SCons

eBook gratuit non affilié créé à partir des **contributeurs de Stack Overflow.**

#scons

Table des matières

À propos

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

Chapitre 1: Démarrer avec SCons

Remarques

SCons est un système de construction. Il prend un tas de fichiers d'entrée et exécute des outils pour produire une sortie. SCons est écrit en Python pur, fonctionne de la même manière sous Linux, Windows et OS X, et peut être exécuté sans installation.

Les fichiers `SConstruct` SCons sont des scripts Python avec des commandes intégrées qui créent une `build tree` . Les SCons exécutent le processus de construction en plusieurs phases. La première est la lecture des fichiers et la construction d'une arborescence de construction. Deuxièmement, traversez l'arborescence pour créer des fichiers cibles.

Versions

Version	Date de sortie
2.5.1	2017-11-03

Examples

Commencer

Une fois que vous avez des SCons en cours d'exécution , créez un fichier nommé `SConstruct` :

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

Maintenant, exécutez les `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` est un script Python avec des fonctions SCons supplémentaires.

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

Le script ci-dessus se compresse dans `archive.zip` utilisant la fonction `Zip()` fournie par SCons. `Zip` est un **générateur** - il construit la **cible** spécifiée par le premier argument à partir de plusieurs **sources** , qui sont utilisées comme second argument par Builders par convention.

SCons **constructeurs** commencent par lettre majuscule et fonctionnent sur l' objet de l' **environnement,** qui stocke la construction configuration. SCons fournit un **environnement** par défaut, mais il peut être créé explicitement pour séparer les variables de construction, choisir différents outils, etc.

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

Notez que lorsque vous exécutez le script pour la deuxième fois, il ne génère rien. Les SCons reconstruisent les cibles uniquement lorsque les fichiers source changent. Modifiez `SConstruct` et exécutez à nouveau les `scons` pour voir la différence.

SCons est conçu pour être extensible. Vous ajoutez vos propres méthodes Builder en les attachant à l'environnement, qui peuvent être traitées dans des rubriques ultérieures.

Lire Démarrer avec SCons en ligne: https://riptutorial.com/fr/scons/topic/4780/demarrer-avec-scons

Chapitre 2: C ++

Examples

Une construction simple

Il est très facile de construire un projet C ++ simple. Voici un exemple de fichier `SConstruct` qui le fait:

```
env=Environment()

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

Cela crée l'exécutable `hello` composé de toutes les sources de `src` avec l'extension `cpp` .

Spécification de diverses options de construction

Cet exemple montre des paramètres de construction plus détaillés:

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

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

Cela construit l'exécutable `hello` de tous les fichiers `cpp` dans `src` , avec les paramètres suivants:

- Le chemin de recherche est `/ usr / include / boost '
- La constante `FOO` est définie
- Les liens exécutables avec la `bar`
- C ++ 11 est utilisé en standard

Lire C ++ en ligne: https://riptutorial.com/fr/scons/topic/6158/c-plusplus

Chapitre 3: Obtenir des SCons en cours d'exécution

Introduction

SCons est écrit en Python 2 et ne nécessite aucune dépendance pour fonctionner. Vous pouvez simplement copier ses scripts dans l'arborescence des sources de votre projet et les exécuter. Vous pouvez également utiliser la version fournie pour votre système d'exploitation.

Examples

Installation sous Linux

Sur Debian ou Ubuntu, vous pouvez installer des SCons en utilisant

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

Sur les systèmes basés sur YUM , utilisez

```
$ sudo yum install scons
```

Vous pouvez installer en utilisant un RPM en le téléchargeant, puis en exécutant

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

Installation sous Windows

Prenez l'installateur sur http://scons.org/pages/download.html

Ou essayez l'outil d'installation de `pip` fourni avec Python:

```
pip install scons
```

Si les `scons` ne peuvent toujours pas être trouvés après cela, assurez-vous que Python `Scripts/` folder est ajouté à `PATH` pour votre installation Python.

En cours d'exécution depuis la source

Si vous avez des modifications à partager ou si vous voulez simplement essayer une nouvelle version en développement.

```
$ hg clone https://bitbucket.org/scons/scons
```

```
$ python scons/src/script/scons.py
```

Installation avec Python pip

```
pip install scons
```

Si vous ne souhaitez pas exécuter les `scons` partir de la ligne de commande, vérifiez que le répertoire des scripts Python est ajouté à PATH pour votre installation.

Si vous voulez jouer avec l'API, l' `import SCons` partir de Python ne fonctionnera pas, car les SCons 2.5.x et suivants permettent d'installer plusieurs versions côte à côte. Cela était nécessaire pour basculer entre les différentes versions de SCons pendant le développement et le dépannage. Maintenant, la manière la plus courante pour cela est d'utiliser `virtualenv` ou de simplement l' exécuter à partir de la source .

Lire Obtenir des SCons en cours d'exécution en ligne:
https://riptutorial.com/fr/scons/topic/9377/obtenir-des-scons-en-cours-d-execution

Chapitre 4: SCons exécuter des phases

Introduction

SCons est un système de construction en plusieurs étapes. Tout d'abord, il lit tous les `SConstruct` et `SConscript` pour exécuter du code Python et crée un `build graph` avec des cibles. Ensuite, il analyse le système de fichiers pour détecter les cibles du `build graph` doivent être mises à jour, puis exécute la commande pour créer des cibles obsolètes.

Examples

Inspection des phases SCons

`scons` décrit les phases en cours elles-mêmes. L'exécuter sur un `SConstruct` vide donne ceci:

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

Pour supprimer les messages de phase, ajoutez l'option `-Q` . `--tree=all` permet de voir l' arbre de dépendance pour cible actuelle que `scons` construit lors de la construction.

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

. est la cible par défaut, ce qui signifie "build SConstruct dans le répertoire courant". `SConstruct` est alors une dépendance pour la construction de la cible par défaut.

Lire SCons exécuter des phases en ligne: https://riptutorial.com/fr/scons/topic/10170/scons-executer-des-phases

Crédits

S. No	Chapitres	Contributeurs
1	Démarrer avec SCons	Ami Tavory, anatoly techtonik, bdbaddog, Community
2	C ++	Ami Tavory
3	Obtenir des SCons en cours d'exécution	Ami Tavory, anatoly techtonik
4	SCons exécuter des phases	anatoly techtonik