使用 SCons 轻松建造程序

江 卫, 软件工程师, 独立撰稿人

简介： 在软件项目开发过程中，make 工具通常被用来建造程序。make 工具通过一个被称为 Makefile 的配置文件可以自动的检测文件之间的依赖关系，这对于建造复杂的项目非常有帮助，然而，编写 Makefile 本身却不是一件容易的事情。SCons 是一个用 Python 语言编写的类似于 make 工具的程序。与 make 工具相比较，SCons 的配置文件更加简单清晰明了，除此之外，它还有许多的优点。本文将简单介绍如何在软件开发项目中使用 SCons，通过本文，读者可以学习到如何使用 SCons 来建造自己的程序项目。

发布日期： 2011 年 4 月 22 日
级别： 中级
访问情况 ： 16027 次浏览
评论： 2 (查看 | 添加评论 - 登录)

前言

make 这个工具自上个世纪 70 年代 Stuart Feldman 在贝尔实验室开发出以来，就一直是类 UNIX 程序员的最爱之一。通过检查文件的修改时间，make 工具可以知道编译目标文件所要依赖的其他文件。在复杂的项目中，如果只有少数几个文件修改过，make 工具知道仅仅需要对哪些文件重新编译就可以确保目标程序被正确的编译链接。这样做的好处就是在编译中，不仅可以节省大量的重复输入，还可以确保程序可以被正确的链接，缩短编译的时间。虽然如此，但是为 make 工具编写建造规则却不是一件容易的事。它复杂的配置规则，即使是有经验的开发者也望而生畏。make 工具的许多替代品便因此而诞生，SCons 就是是其中之一。SCons 是一个用 Python 语言编写的类似于 make 工具的程序。与 make 工具相比较，SCons 的配置文件更加简单清晰明了，除此之外，它还有许多的优点。

SCons 简介

SCons 是一个开放源代码、以 Python 语言编写的下一代的程序建造工具。它最初的名字是 ScCons, 基于由 perl 语言编写的 Cons 软件开发而成，它在 2000 年 8 月获得了由 Software Carpentry 举办的 SC 建造比赛的大奖。现在 ScCons 已经被改名为 SCons，目的是为了表示不再与 Software Carpentry 有联系，当然，还有一个目的，就是为了更方便的输入。

作为下一代的软件建造工具，SCons 的设计目标就是让开发人员更容易、更可靠和更快速的建造软件。与传统的 make 工具比较，SCons 具有以下优点：

- 使用 Python 脚本做为配置文件
- 对于 C,C++ 和 Fortran, 内建支持可靠自动依赖分析 . 不用像 make 工具那样需要 执行"make depends"和"make clean"就可以获得所有的依赖关系。
- 内建支持 C, C++, D, Java, Fortran, Yacc, Lex, Qt，SWIG 以及 Tex/Latex。 用户还可以根据自己的需要进行扩展以获得对需要编程语言的支持。
- 支持 make -j 风格的并行建造。相比 make -j, SCons 可以同时运行 N 个工作，而 不用担心代码的层次结构。
- 使用 Autoconf 风格查找头文件，函数库，函数和类型定义。
- 良好的夸平台性。SCons 可以运行在 Linux, AIX, BSD, HP/UX, IRIX, Solaris, Windows, Mac OS X 和 OS/2 上。

安装 SCons

SCons 支持多种操作系统平台，并为各个系统制作了易于安装的文件，因此在各个系统平台上的安装方法不尽相同，在 SCons 的官方网站上可以查每个平台的具体安装方法。 如果 SCons 没有为你的系统制作相应的安装包，你也可以下载 SCons 的源代码，直接进行安装。 首先，从 SCons 的网站上下载最新的 SCons 源代码包（目前 SCons 的最新版本是 2.0.1）。 其次，解压下载的源代码。视下载的源代码包的格式不同而有不同的方法，在 Windows 平台上，可是使用 winzip 或者其他类似的工具解压。在 Linux 平台上，对于 tar 包，使用 tar 命令进行解压，如：

```
$ tar -zxf scons-2.0.1.tar.gz
```

然后切换进入解压后的目录进行安装，如

```
$ cd scons-2.0.1
$ sudo python setup.py install
```

命令执行如果没有错误，那么 scons 就被安装到系统上了。对于 Linux 来说，scons 会默认安装到 /usr/loca/bin 目录下，而在 Windows 平台上，则会被安装到 C:\Python25\Scripts 下。

使用 SCons

在 SCons 安装完成后，我们就可以使用 SCons 来建造我们的程序或者项目了。像很多编程书籍那样，在这里我们也通过一个简单的 helloscons 例子来说明如何使用 SCons。例子 helloscons 包含两个文件 :

```
$ ls helloscons
helloscons.c  SConstruct
```

其中 helloscons.c 是程序的源文件，SConstruct 是 scons 的配置文件，类似使用 make 工具时的 Makefile 文件，因此，为了编译你的项目，需要手工创建一个 SConstruct 文件（注意：文件名是大小写敏感的）。不过，在编译的时候不需要指定它。 要编译这个例子，切换到 helloscons 的目录下，运行 scons 命令，如下：

```
$ cd helloscons/
$ scons
scons: Reading SConscript files ...
scons: done reading SConscript files.
scons: Building targets ...
gcc -o helloscons.o -c helloscons.c
gcc -o helloscons helloscons.o
scons: done building targets.
```

来查看一下运行 scons 命令后得到的结果 :

```
$ ls
helloscons  helloscons.c  helloscons.o  SConstruct
```

建造结束后，得到了二进制文件 helloscons 以及编译的过程中产生的一些以 .o 结尾的目标文件。试运行 helloscons 一下 , 会得到 :

```
$ ./helloscons
Hello, SCons!
```

现在让我们回过头来解析一下 helloscons 这个例子 . helloscons.c 是这个例子里的唯一一个源代码文件，它所做的事就是在控制台上输出一行简单的"Hello,SCons", 它的源代码如下：

清单 1. helloscons.c

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char* argv[])
{
       printf("Hello, SCons!\n");
       return 0;
}
```

作为项目建造规则的配置文件 SConstruct 的内容如下 :

清单 2. SConstruct 文件

```
Program('helloscons.c')
```

你可能很惊讶 SConstruct 的内容只有一行，然而事实确实如此，它比传统的 Makefile 简单很多。SConstruct 以 Python 脚本的语法编写，你可以像编写 Python 脚本一样来编写它。其中的 Program 是编译的类型，说明你准备想要建造一个可执行的二进制程序，它由 helloscons.c 文件来生成。在这里，没有指定生成的可执行程序的名字。不过不用担心，SCons 会把源代码文件名字的后缀去掉，用来作为可执行文件的名字。在这里，我们甚至不需要像 Makefile 那样指定清理的动作，就可以执行清理任务。在 SCons 中，执行清理任务由参数 -c 指定，如下 :

```
$ scons -c
scons: Reading SConscript files ...
scons: done reading SConscript files.
scons: Cleaning targets ...
Removed helloscons.o
Removed helloscons
scons: done cleaning targets.

$ ls
helloscons.c  SConstruct
```

如果你不想直接编译可执行的二进制文件，那也没有关系。SCons 支持多种编译类型，你可以根据自己的需要，任意选用其中的一种。SCons 支持的编译类型有：

- Program： 编译成可执行程序（在 Windows 平台上即是 exe 文件），这是最常用的一种编译类型。
- Object： 只编译成目标文件。使用这种类型，编译结束后，只会产生目标文件。在 POSIX 系统中，目标文件以 .o 结尾，在 Windows 平台上以 .OBJ 结尾。
- Library： 编译成库文件。SCons 默认编译的库是指静态链接库。
- StaticLibrary： 显示的编译成静态链接库，与上面的 Library 效果一样。
- SharedLibrary： 在 POSIX 系统上编译动态链接库，在 Windows 平台上编译 DLL。

这个简单的 SConstruct 的配置文件从一个侧面说明了使用 SCons 来建造程序是多么的简单。 在实际的项目开发中，程序的建造规则远比 helloscons 这个例子复杂。不过，这些都不是问题，你可以像扩展你自己的 Python 脚本文件那样去扩展 SConstruct. 如果你不想使用 SConstruct 为你设置的默认可执行文件的名字，而是选择你自己喜欢的名字，如 myscons，可以把 SConstruct 的内容修改为 :

```
Program('myscons, 'helloscons.c')
```

其中 myscons 就是你想要的可执行文件的名字，你可以把它换成任意你喜欢的名字， 不过有点注意的是，这个名字必须放在第一位。 然后在 helloscons 目录下运行 scons 命令，就会得到 myscons 这个可执行文件，如 下：

```
$ scons -Q
gcc -o helloscons.o -c helloscons.c
gcc -o myscons helloscons.o
```

其中的 -Q 参数是减少编译时的由 scons 产生的冗余信息。 如果你的项目由多个源文件组成，而且你想指定一些编译的宏定义，以及显式的指定使用某些库，这些对于 SCons 来说，都是非常简单的事情。我们的另外一个例子 helloscons2 很好的说明这种情况。 helloscons2 由 3 个源文件组成，它们是 helloscon2.c, file1.c, file2.c，另外指定了编译的选项，同时还指定了使用哪些具体的库。让我们来看一下 helloscons2 的 SConstruct 文件 :

```
Program('helloscons2', ['helloscons2.c', 'file1.c', 'file2.c'],
       LIBS = 'm',
       LIBPATH = ['/usr/lib', '/usr/local/lib'],
       CCFLAGS = '-DHELLOSCONS')
```

正如你想像的那样，这样一个配置文件并不复杂 . 该 SConstruct 文件指出，它将生成一个名叫 helloscons2 的可执行程序，该可执行程序由 helloscons2.c, file1.c 和 file2.c 组成。注意，多个源文件需要放在一个 Python 列表中。如果你的源程序代码文件很多，有十几个甚至上百个，那不要一个个的将他们都列出来，你可以使用 glob('*.c') 来代替源代码列表。如下 :

```
Program('helloscons2', Glob('*.c')
```

配置文件中 LIBS,LIBAPTH 和 CCFLAGS 是 SCons 内置的关键字，它们的作用如下：

- LIBS： 显示的指明要在链接过程中使用的库，如果有多个库，应该把它们放在一个列表里面。这个例子里，我们使用一个称为 m 的库。
- LIBPATH： 链接库的搜索路径，多个搜索路径放在一个列表中。这个例子里，库的搜索路径是 /usr/lib 和 /usr/local/lib。
- CCFLAGS： 编译选项，可以指定需要的任意编译选项，如果有多个选项，应该放在一个列表中。这个例子里，编译选项是通过 -D 这个 gcc 的选项定义了一个宏 HELLOSCONS。

运行 scons 命令的时候，可以看到这些变量如何被使用的，让我们执行一下 scons 命令 :

```
$ scons -Q
gcc -o file1.o -c -DHELLOSCONS file1.c
gcc -o file2.o -c -DHELLOSCONS file2.c
gcc -o helloscons2.o -c -DHELLOSCONS helloscons2.c
gcc -o helloscons2 helloscons2.o file1.o file2.o -L/usr/lib -L/usr/local/lib -lm
```

scons 命令的输出显示了可执行程序 helloscons2 如何由多个源文件而生成，以及在 SConstruct 中定义的 LIBS,LIBPATH 和 CCFLAGS 如何被使用。 可见，即使对于复杂的项目，SCons 的编译配置文件也很简单。除此之外，SCons 也提供了很多功能以适应不同的需要，如果读者想更深入的了解如何使用 SCons，可以参考 SCons 的帮助手册。

总结

本文简单介绍了 SCons 的特点，如何安装 SCons，以及通过例子来说明如何在项目中使用 SCons。 作为下一代的软件建造工具，SCons 使用 Python 语言作为配置文件，不但功能强大，而且简单易用，对于跨平台的项目，非常适合。 如果你厌烦了 make 工具的那种复杂的编写规则，尝试一下新鲜的 SCons 吧。

下载

描述	名字	大小	下载方法
本文用到的示例代码	sample.zip	10KB	HTTP

关于下载方法的信息

参考资料

学习

- 可以从 SCons 的网站上下载 SCons 工具以及相关的文档
- 在 Python 的网站了解更多的 Python 知识。
- 在 GNU make 的网站了解更多的 GNU make 知识。
- 在 CONS 的网站了解更多的 CONS 知识。
- 在 developerWorks Linux 专区 寻找为 Linux 开发人员（包括 Linux 新手入门）准备的更多参考资料，查阅我们 最受欢迎的文章和教程。
- 在 developerWorks 上查阅所有 Linux 技巧 和 Linux 教程。
- 随时关注 developerWorks 技术活动和网络广播。

获得产品和技术

- 下载 IBM 软件试用版，体验强大的 DB2®，Lotus®，Rational®，Tivoli®和 WebSphere®软件。

讨论

- 参与论坛讨论。
- 查看 developerWorks 博客的最新信息。
- 加入 developerWorks 中文社区，developerWorks 社区是一个面向全球 IT 专业人员，可以提供博客、书签、wiki、群组、联系、共享和协作等社区功能的专业社交网络社区。

关于作者

江卫是一名有丰富软件开发经验的工程师，热爱 Linux 内核及 Python 开发。他目前的工作是为一家刀架服务器厂商维护 Linux 下的网络设备驱动程序。此外可以与他在 developerWorks 中文社区交流：http://www.ibm.com/developerworks/mydeveloperworks/profiles/user/jiangwei909。

单击**提交**则表示您同意developerWorks 的条款和条件。 使用条款.

提交 取消

所有提交的信息确保安全。

平均分 (28个评分)

1 星

1 星
2 星

2 星
3 星

3 星
4 星

4 星
5 星

5 星

提交

添加评论:

请 登录 或 注册 后发表评论。

注意：评论中不支持 HTML 语法

有新评论时提醒我剩余 1000 字符

发布

共有评论 (2)

scons确实很有吸引力,但是如果我们的系统是使用旧的Make系统, 如何才能高效从旧的Make系统升级到新的scons系统. 有没有高效的工具?

由 **carlshen8** 于 2012年06月28日发布

报告滥用

虽然也有gnu autotools和CMake之类的，但是它们用的是专有语言，需要专门的学习而且学完之后难有其它用处，这估计是开发人员不愿费时间学习的主要原因吧。scons以优美的python语言编写，确实很有吸引力。

由 **tangxinfa** 于 2011年05月18日发布

报告滥用