

Vim / Cscope教程



Cscope是一个非常方便的工具，当您不必离开喜欢的编辑器（例如Vim）来使用它时，它甚至会更好。幸运的是，Vim内置了对Cscope的支持。

本教程向您介绍Vim内置的Cscope支持以及使搜索更加方便的一组地图。

假定您了解使用vi样式编辑器的基础知识，但是您不需要有关Vim的任何特定知识（在使用Vim特定功能的情况下，例如多个窗口，简要介绍了这些功能的实用知识介绍）。您也不需要了解任何有关Cscope的知识：我们会逐步介绍基础知识。

简而言之，如果您使用过Vim的Cscope支持，则与Vim的ctags功能非常相似。但是，由于Cscope比ctags具有更多的搜索类型，因此存在一些差异。

这是一个动手教程，因此打开一个外壳，然后按照以下步骤操作：

1. 如果您的计算机上尚未安装Cscope，请获取并安装。理想情况下，您还将拥有Vim 6.x，但是您可以在Vim 5的更高版本中获得大多数功能（垂直分割不起作用，但是如果您按照文件注释中的描述修改地图，则水平分割将起作用）。。

注意：如果您的Vim版本未使用'--enable-cscope'进行编译，则需要使用该标志重新配置和重新编译Vim。Linux发行版附带的大多数Vim二进制文件都启用了Cscope插件。

2. 下载cscope_maps.vim文件，并安排Vim在启动时读取它。如果您使用的是Vim 6.x，请将文件粘贴在\$ HOME / .vim / plugin目录中（或在“运行时路径”中的其他“ plugin”子目录中）。如果使用的是Vim 5.x，则可以将cscope_maps文件的全部内容剪切并粘贴到\$ HOME / .vimrc文件中，也可以将“ source cscope_maps.vim”行粘贴到.vimrc文件中。
3. 进入其中包含一些C代码的目录，然后输入'cscope -R'（'-R'使Cscope解析所有子目录，而不仅仅是当前目录）。由于我们没有传递'-b'标志（它告诉Cscope仅构建数据库，然后退出），因此您还将发现自己处于Cscope基于curses的GUI中。尝试几次搜索（提示：您可以使用箭头键在搜索类型之间移动，并使用“ tab”键在搜索类型和搜索结果之间切换）。点击搜索结果最左边的数字，Cscope将在该位置向右打开Vim。（除非您将EDITOR环境变量设置为Vim以外的其他变量）。退出Vim，您将立即回到上次停止的Cscope GUI。好漂亮

scope，Cscope界面有一个大问题：每次要执行新搜索时，都需要退出Vim。这就是Vim插件的所在。按CTRL-D退出Cscope。

4. 启动Vim。如果需要，可以以C符号（例如：'vim -t main'）开头，并且应该在代码中跳至该符号的定义。
5. 将光标放在程序中多个位置使用的C符号上。快速连续键入“ CTRL-\ s”（Control-\反斜杠，然后是“ s”），您应该在Vim窗口的底部看到一个菜单，其中显示了程序中该符号的所有用法。选

择其中之一，然后按Enter键，您将跳转到该用法。与ctags一样，您可以按“CTRL-t”跳回到搜索之前的原始位置（并且您可以嵌套搜索，而CTRL-t则一次展开它们）。

助记符：'\'键位于'j'键旁边，用于ctags搜索。

6. 尝试相同的搜索，但是这次通过“CTRL-spacebar s”。这次，您的Vim窗口将在水平方向分成两部分，并且Cscope搜索结果将被放置在新窗口中。[如果以前从未使用过多个Vim窗口：通过'CTRL-W w'（或CTRL-W箭头键，或CTRL-W h / j / k / l表示左/上/下/右）在窗口之间移动，通过'CTRL-W c'（或旧的'：q'）关闭窗口，通过'CTRL-W o'将当前窗口设为唯一窗口，通过'CTRL-W s'将窗口分为两个窗口（或'CTRL-W v'进行垂直分割），通过'：spl [it]文件名'在新窗口中打开文件

助记符：屏幕中间有一个很大的，类似于空格键的条，用于分隔Vim窗口。



7. 现在，尝试通过“CTRL-空格键CTRL-空格键s”进行相同的搜索（只需按住CTRL键并点两次空格键即可）。如果您无法快速击键以使其正常工作，请进入cscope_maps.vim脚本并按照注释中的说明更改Vim的超时设置[实际上，我通常建议您关闭Vim的超时]。这次，您的Vim窗口将被垂直分割（注意：这在Vim 5.x中不起作用，因为垂直分割是Vim 6.0中的新增功能）。
8. 到现在为止，我们仅使用了来自'cscope_maps.vim'的击键映射，它们都对恰好位于Vim中光标下方的术语进行搜索。要进行Cscope搜索（使用Vim内置的Cscope支持）的老式方法，请输入“：cscope find symbol foo”（或更简洁地说，是“：cs fs foo”）。要进行水平拆分，请使用“：scscope”（或仅使用“：scs”）（仅限Vim 6.x）。如果要搜索的单词位于光标下方，则使用地图更容易，但是命令行界面可让您转到所键入的任何符号，因此，您肯定会不时使用它。
9. 到目前为止，我们只进行了一种搜索：“s”，用于“查找符号X的所有用法”。尝试使用不同的字母来进行Cscope的其他搜索之一：'g'查找符号的全局定义，'c'查找对函数的所有调用，'f'打开光标下的文件名（注意：默认情况下，Cscope解析它在/usr/include中找到的所有C头文件，您可以使用此文件打开大多数标准的include文件。这些是我最常使用的，但是还有其他（请在cscope_maps.vim文件中查找所有这些，和/或阅读Cscope手册页）。
10. 尽管Cscope最初仅用于C代码，但实际上它是一种非常灵活的工具，可以与C++和Java之类的语言很好地配合使用。您可以将其视为通用的“grep”数据库，能够识别某些其他构造，例如函数调用和变量定义。默认情况下，Cscope仅解析当前目录（和子目录（如果传递-R标志）中的C，lex和yacc文件（.c，.h，.l，.y），并且目前无法更改该文件扩展名列表（是的，我们应该更改它）。因此，您必须列出要解析的文件，并将其命名为“cscope.files”（如果调用“cscope -i foofile”，则可以将其命名为任意名称）。

找。-name'* .java'> cscope.files

现在运行“cscope -b”来重建数据库（-b只是在不启动Cscope GUI的情况下建立数据库），您将能够浏览Java文件中的所有符号。显然，外面有人使用Cscope浏览和编辑大量文档文件，这表明Cscope的解析器非常灵活。

对于较大的项目，您可能还需要使用-q标志和/或使用更复杂的“查找”命令。有关更多信息，请参见有关[在大型项目中使用Cscope的教程](#)。



11. 尝试将\$ CSCOPE_DB环境变量设置为指向您创建的Cscope数据库，这样就不必总是在与数据库相同的目录中启动Vim。这对于将代码拆分为多个子目录的项目特别有用。注意：要使此方法有效，您应该使用绝对路径名构建数据库：cd到/，然后执行

```
查找/ my / project / dir -name'* .c'-o -name'* .h'> /foo/cscope.files
```

然后在与cscope.files文件相同的目录中运行Cscope（或使用'cscope -i /foo/cscope.files'），然后设置并导出\$ CSCOPE_DB变量，将其指向生成的cscope.out文件）：

```
cd / foo
cscope -b
CSCOPE_DB = / foo / cscope.out; 导出CSCOPE_DB
```

（上面的最后一条命令是针对Bourne / Korn / Bash shell的：我忘记了如何在基于csh的shell中导出变量，因为我避免像瘟疫一样避开它们）。

现在，您应该能够在计算机上的任何目录中运行'vim -t foo'，并使Vim跳至'foo'的定义。我倾向于为我的所有不同项目编写小的shell脚本（仅定义和导出CSCOPE_DB），这使我可以使用简单的“source projectA”命令在它们之间进行切换。



错误：在15.4之前的Cscope版本中，有一个愚蠢的错误可能会导致Vim冻结，除非您调用数据库而不是默认的'cscope.out'：在Cscope中使用'-f foo'调用将您的数据库命名为“foo.out”，您将可以正常运行。

12. 而已！如果您有疑问，请使用“：help cscope”（在Vim中）和/或“man cscope”（在您的外壳中），以了解要点。



Cscope支持由Andy Kahn
Tutorial 添加到Vim，由Jason Duell 添加到 cscope_maps.vim。 Petr
Sorfa添加了Cscope艺术

[返回Cscope主页](#)