技巧和窍门

在结束本章之前，我还想和大家分享一些 Git 使用的技巧和窍门。很多使用 Git 的开发者可能根本就没用过这些技巧，我们也不是说在读过本书后非得用这些技巧不可，但至少应该有所了解吧。说实话，有了这些小窍门，我们的工作可以变得更简单，更轻松，更高效。

自动补全

如果你用的是 Bash shell，可以试试看 Git 提供的自动补全脚本。下载 Git 的源代码，进入 contrib/completion 目录，会看到一个 git-completion.bash 文件。将此文件复制到你自己的用户主目录中（译注：按照下面的示例，还应改名加上点：cp git-completion.bash ~/.git-completion.bash），并把下面一行内容添加到你的 .bashrc 文件中：

source ~/.git-completion.bash

也可以为系统上所有用户都设置默认使用此脚本。Mac 上将此脚本复制到 /opt/local/etc/bash\_completion.d 目录中，Linux 上则复制到 /etc/bash\_completion.d/ 目录中。这两处目录中的脚本，都会在 Bash 启动时自动加载。

如果在 Windows 上安装了 msysGit，默认使用的 Git Bash 就已经配好了这个自动补全脚本，可以直接使用。

在输入 Git 命令的时候可以敲两次跳格键（Tab），就会看到列出所有匹配的可用命令建议：

$ git co<tab><tab>

commit config

此例中，键入 git co 然后连按两次 Tab 键，会看到两个相关的建议（命令） commit 和 config。继而输入 m<tab> 会自动完成 git commit 命令的输入。

命令的选项也可以用这种方式自动完成，其实这种情况更实用些。比如运行 git log 的时候忘了相关选项的名字，可以输入开头的几个字母，然后敲 Tab 键看看有哪些匹配的：

$ git log --s<tab>

--shortstat --since= --src-prefix= --stat --summary

这个技巧不错吧，可以节省很多输入和查阅文档的时间。

Git 命令别名

Git 并不会推断你输入的几个字符将会是哪条命令，不过如果想偷懒，少敲几个命令的字符，可以用 git config 为命令设置别名。来看看下面的例子：

$ git config --global alias.co checkout

$ git config --global alias.br branch

$ git config --global alias.ci commit

$ git config --global alias.st status

现在，如果要输入 git commit 只需键入 git ci 即可。而随着 Git 使用的深入，会有很多经常要用到的命令，遇到这种情况，不妨建个别名提高效率。

使用这种技术还可以创造出新的命令，比方说取消暂存文件时的输入比较繁琐，可以自己设置一下：

$ git config --global alias.unstage 'reset HEAD --'

这样一来，下面的两条命令完全等同：

$ git unstage fileA

$ git reset HEAD fileA

显然，使用别名的方式看起来更清楚。另外，我们还经常设置 last 命令：

$ git config --global alias.last 'log -1 HEAD'

然后要看最后一次的提交信息，就变得简单多了：

$ git last

commit 66938dae3329c7aebe598c2246a8e6af90d04646

Author: Josh Goebel <dreamer3@example.com>

Date: Tue Aug 26 19:48:51 2008 +0800

test for current head

Signed-off-by: Scott Chacon <schacon@example.com>

可以看出，实际上 Git 只是简单地在命令中替换了你设置的别名。不过有时候我们希望运行某个外部命令，而非 Git 的子命令，这个好办，只需要在命令前加上 ! 就行。如果你自己写了些处理 Git 仓库信息的脚本的话，就可以用这种技术包装起来。作为演示，我们可以设置用 git visual 启动 gitk：

$ git config --global alias.visual '!gitk'