The Refspec

这本书读到这里，你已经使用过一些简单的远程分支到本地引用的映射方式了，这种映射可以更为复杂。 假设你像这样添加了一项远程仓库：

$ git remote add origin git@github.com:schacon/simplegit-progit.git

它在你的 .git/config 文件中添加了一节，指定了远程的名称 (origin), 远程仓库的URL地址，和用于获取操作的 Refspec:

[remote "origin"]

url = git@github.com:schacon/simplegit-progit.git

fetch = +refs/heads/\*:refs/remotes/origin/\*

Refspec 的格式是一个可选的 + 号，接着是 <src>:<dst> 的格式，这里 <src> 是远端上的引用格式， <dst> 是将要记录在本地的引用格式。可选的 + 号告诉 Git 在即使不能快速演进的情况下，也去强制更新它。

缺省情况下 refspec 会被 git remote add 命令所自动生成， Git 会获取远端上 refs/heads/ 下面的所有引用，并将它写入到本地的 refs/remotes/origin/. 所以，如果远端上有一个 master 分支，你在本地可以通过下面这种方式来访问它的历史记录：

$ git log origin/master

$ git log remotes/origin/master

$ git log refs/remotes/origin/master

它们全是等价的，因为 Git 把它们都扩展成 refs/remotes/origin/master.

如果你想让 Git 每次只拉取远程的 master 分支，而不是远程的所有分支，你可以把 fetch 这一行修改成这样：

fetch = +refs/heads/master:refs/remotes/origin/master

这是 git fetch 操作对这个远端的缺省 refspec 值。而如果你只想做一次该操作，也可以在命令行上指定这个 refspec. 如可以这样拉取远程的 master 分支到本地的 origin/mymaster 分支：

$ git fetch origin master:refs/remotes/origin/mymaster

你也可以在命令行上指定多个 refspec. 像这样可以一次获取远程的多个分支：

$ git fetch origin master:refs/remotes/origin/mymaster \

topic:refs/remotes/origin/topic

From git@github.com:schacon/simplegit

! [rejected] master -> origin/mymaster (non fast forward)

\* [new branch] topic -> origin/topic

在这个例子中， master 分支因为不是一个可以快速演进的引用而拉取操作被拒绝。你可以在 refspec 之前使用一个+ 号来重载这种行为。

你也可以在配置文件中指定多个 refspec. 如你想在每次获取时都获取 master 和 experiment 分支，就添加两行：

[remote "origin"]

url = git@github.com:schacon/simplegit-progit.git

fetch = +refs/heads/master:refs/remotes/origin/master

fetch = +refs/heads/experiment:refs/remotes/origin/experiment

但是这里不能使用部分通配符，像这样就是不合法的：

fetch = +refs/heads/qa\*:refs/remotes/origin/qa\*

但无论如何，你可以使用命名空间来达到这个目的。如你有一个QA组，他们推送一系列分支，你想每次获取master 分支和QA组的所有分支，你可以使用这样的配置段落：

[remote "origin"]

url = git@github.com:schacon/simplegit-progit.git

fetch = +refs/heads/master:refs/remotes/origin/master

fetch = +refs/heads/qa/\*:refs/remotes/origin/qa/\*

如果你的工作流很复杂，有QA组推送的分支、开发人员推送的分支、和集成人员推送的分支，并且他们在远程分支上协作，你可以采用这种方式为他们创建各自的命名空间。

推送 Refspec

采用命名空间的方式确实很棒，但QA组成员第1次是如何将他们的分支推送到 qa/ 空间里面的呢？答案是你可以使用 refspec 来推送。

如果QA组成员想把他们的 master 分支推送到远程的 qa/master 分支上，可以这样运行：

$ git push origin master:refs/heads/qa/master

如果他们想让 Git 每次运行 git push origin 时都这样自动推送，他们可以在配置文件中添加 push 值：

[remote "origin"]

url = git@github.com:schacon/simplegit-progit.git

fetch = +refs/heads/\*:refs/remotes/origin/\*

push = refs/heads/master:refs/heads/qa/master

这样，就会让 git push origin 缺省就把本地的 master 分支推送到远程的 qa/master 分支上。

删除引用

你也可以使用 refspec 来删除远程的引用，是通过运行这样的命令：

$ git push origin :topic

因为 refspec 的格式是 <src>:<dst>, 通过把 <src> 部分留空的方式，这个意思是是把远程的 topic 分支变成空，也就是删除它。