

- [首页](#)
- [头条](#)
- [小组](#)
- [资源](#)
- [注册](#)
- [登录](#)



- [首页](#)
- [最新文章](#)
- [在线课程](#)
- [业界](#)
- [开发](#)
- [IT技术](#)
- [设计](#)
- [创业](#)
- [IT职场](#)
- [在国外](#)
- [频道](#)
- [更多 >](#)



- 导航条 -

[伯乐在线](#) > [首页](#) > [所有文章](#) > [IT技术](#) > Git远程操作详解

Git远程操作详解

2014/06/12 | 分类: [IT技术](#) | [1 条评论](#) | 标签: [Git](#)

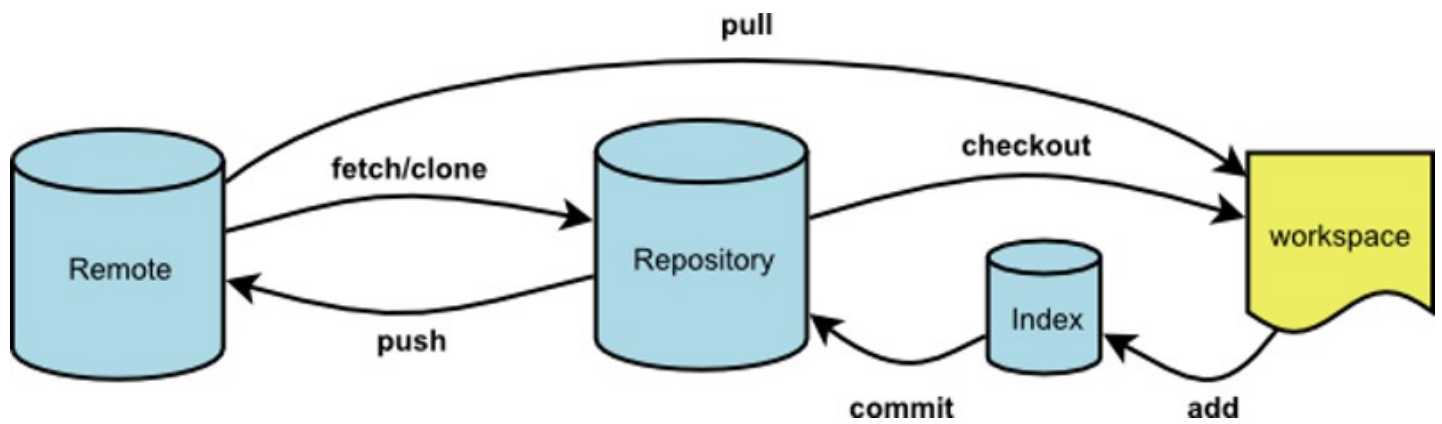
分享到: 11 原文出处: [阮一峰](#)

[Git](#)是目前最流行的[版本管理系统](#)，学会Git几乎成了开发者的必备技能。

Git有很多优势，其中之一就是远程操作非常简便。本文详细介绍5个Git命令，它们的概念和用法，理解了这些内容，你就会完全掌握Git远程操作。

- git clone
- git remote
- git fetch
- git pull
- git push

本文针对初级用户，从最简单的讲起，但是需要读者对Git的基本用法有所了解。同时，本文覆盖了上面5个命令的几乎所有的常用用法，所以对于熟练用户也有参考价值。



一、git clone

远程操作的第一步，通常是从远程主机克隆一个版本库，这时就要用到git clone命令。

```
1 | $ git clone <版本库的网址>
```

比如，克隆jQuery的版本库。

```
1 | $ git clone https://github.com/jquery/jquery.git
```

该命令会在本地主机生成一个目录，与远程主机的版本库同名。如果要指定不同的目录名，可以将目录名作为git clone命令的第二个参数。

```
1 | $ git clone <版本库的网址> <本地目录名>
```

git clone支持多种协议，除了HTTP(s)以外，还支持SSH、Git、本地文件协议等，下面是一些例子。

```
1 | $ git clone http[s]://example.com/path/to/repo.git/
2 | $ git clone ssh://example.com/path/to/repo.git/
3 | $ git clone git://example.com/path/to/repo.git/
4 | $ git clone /opt/git/project.git
5 | $ git clone file:///opt/git/project.git
6 | $ git clone ftp[s]://example.com/path/to/repo.git/
7 | $ git clone rsync://example.com/path/to/repo.git/
```

SSH协议还有另一种写法。

```
1 | $ git clone [user@]example.com:path/to/repo.git/
```

通常来说，Git协议下载速度最快，SSH协议用于需要用户认证的场合。各种协议优劣的详细讨论请参考[官方文档](#)。

二、git remote

为了便于管理，Git要求每个远程主机都必须指定一个主机名。git remote命令就用于管理主机名。

不带选项的时候，git remote命令列出所有远程主机。

```
1 | $ git remote
2 | origin
```

使用-v选项，可以参看远程主机的网址。

```
1 | $ git remote -v
2 | origin git@github.com:jquery/jquery.git (fetch)
3 | origin git@github.com:jquery/jquery.git (push)
```

上面命令表示，当前只有一台远程主机，叫做origin，以及它的网址。

克隆版本库的时候，所使用的远程主机自动被Git命名为origin。如果想用其他的主机名，需要用git clone命令的-o选项指定。

```
1 | $ git clone -o jQuery https://github.com/jquery/jquery.git
2 | $ git remote
```

```
3 | jQuery
```

上面命令表示，克隆的时候，指定远程主机叫做jQuery。

git remote show命令加上主机名，可以查看该主机的详细信息。

```
1 | $ git remote show <主机名>
```

git remote add命令用于添加远程主机。

```
1 | $ git remote add <主机名> <网址>
```

git remote rm命令用于删除远程主机。

```
1 | $ git remote rm <主机名>
```

git remote rename命令用于远程主机的改名。

```
1 | $ git remote rename <原主机名> <新主机名>
```

三、git fetch

一旦远程主机的版本库有了更新（Git术语叫做commit），需要将这些更新取回本地，这时就要用到git fetch命令。

```
1 | $ git fetch <远程主机名>
```

上面命令将某个远程主机的更新，全部取回本地。

默认情况下，git fetch取回所有分支（branch）的更新。如果只想取回特定分支的更新，可以指定分支名。

```
1 | $ git fetch <远程主机名> <分支名>
```

比如，取回origin主机的master分支。

```
1 | $ git fetch origin master
```

所取回的更新，在本地主机上要用”远程主机名/分支名”的形式读取。比如origin主机的master，就要用origin/master读取。

git branch命令的-r选项，可以用来查看远程分支，-a选项查看所有分支。

```
1 | $ git branch -r
2 | origin/master
3 |
4 | $ git branch -a
5 | * master
6 | remotes/origin/master
```

上面命令表示，本地主机的当前分支是master，远程分支是origin/master。

取回远程主机的更新以后，可以在它的基础上，使用git checkout命令创建一个新的分支。

```
1 | $ git checkout -b newBranch origin/master
```

上面命令表示，在origin/master的基础上，创建一个新分支。

此外，也可以使用git merge命令或者git rebase命令，在本地分支上合并远程分支。

```
1 | $ git merge origin/master
2 | # 或者
3 | $ git rebase origin/master
```

上面命令表示在当前分支上，合并origin/master。

四、git pull

git pull命令的作用是，取回远程主机某个分支的更新，再与本地的指定分支合并。它的完整格式稍稍有点复杂。

```
1 | $ git pull <远程主机名> <远程分支名>:<本地分支名>
```

比如，取回origin主机的next分支，与本地的master分支合并，需要写成下面这样。

```
1 | $ git pull origin next:master
```

如果远程分支是与当前分支合并，则冒号后面的部分可以省略。

```
1 | $ git pull origin next
```

上面命令表示，取回origin/next分支，再与当前分支合并。实质上，这等同于先做git fetch，再做git merge。

```
1 | $ git fetch origin
2 | $ git merge origin/next
```

在某些场合，Git会自动在本地分支与远程分支之间，建立一种追踪关系（tracking）。比如，在git clone的时候，所有本地分支默认与远程主机的同名分支，建立追踪关系，也就是说，本地的master分支自动“追踪”origin/master分支。

Git也允许手动建立追踪关系。

```
1 | git branch --set-upstream master origin/next
```

上面命令指定master分支追踪origin/next分支。

如果当前分支与远程分支存在追踪关系，git pull就可以省略远程分支名。

```
1 | $ git pull origin
```

上面命令表示，本地的当前分支自动与对应的origin主机“追踪分支”（remote-tracking branch）进行合并。

如果当前分支只有一个追踪分支，连远程主机名都可以省略。

```
1 | $ git pull
```

上面命令表示，当前分支自动与唯一一个追踪分支进行合并。

如果合并需要采用rebase模式，可以使用 -rebase选项。

```
1 | $ git pull --rebase <远程主机名> <远程分支名>:<本地分支名>
```

五、git push

git push命令用于将本地分支的更新，推送到远程主机。它的格式与git pull命令相仿。

```
1 | $ git push <远程主机名> <本地分支名>:<远程分支名>
```

注意，分支推送顺序的写法是<来源地>:<目的地>，所以git pull是<远程分支>:<本地分支>，而git push是<本地分支>:<远程分支>。

如果省略远程分支名，则表示将本地分支推送与之存在“追踪关系”的远程分支（通常两者同名），如果该远程分支不存在，则会被新建。

```
1 | $ git push origin master
```

上面命令表示，将本地的master分支推送到origin主机的master分支。如果后者不存在，则会被新建。

如果省略本地分支名，则表示删除指定的远程分支，因为这等同于推送一个空的本地分支到远程分支。

```
1 | $ git push origin :master
2 | # 等同于
3 | $ git push origin --delete master
```

上面命令表示删除origin主机的master分支。

如果当前分支与远程分支之间存在追踪关系，则本地分支和远程分支都可以省略。

```
1 | $ git push origin
```

上面命令表示，将当前分支推送到origin主机的对应分支。

如果当前分支只有一个追踪分支，那么主机名都可以省略。

```
1 | $ git push
```

如果当前分支与多个主机存在追踪关系，则可以使用-u选项指定一个默认主机，这样后面就可以不加任何参数使用git push。

```
1 | $ git push -u origin master
```

上面命令将本地的master分支推送到origin主机，同时指定origin为默认主机，后面就可以不加任何参数使用git push了。

不带任何参数的git push，默认只推送当前分支，这叫做simple方式。此外，还有一种matching方式，会推送所有有对应的远程分支的本地分支。Git 2.0版本之前，默认采用matching方法，现在改为默认采用simple方式。如果要修改这个设置，可以采用git config命令。

```
1 | $ git config --global push.default matching
2 | # 或者
3 | $ git config --global push.default simple
```

还有一种情况，就是不管是否存在对应的远程分支，将本地的所有分支都推送到远程主机，这时需要使用 - all选项。

```
1 | $ git push --all origin
```

上面命令表示，将所有本地分支都推送到origin主机。

如果远程主机的版本比本地版本更新，推送时Git会报错，要求先在本地做git pull合并差异，然后再推送到远程主机。这时，如果你一定要推送，可以使用 - force选项。

```
1 | $ git push --force origin
```

上面命令使用 - force选项，结果导致在远程主机产生一个”非直进式”的合并（non-fast-forward merge）。除非你很确定要这样做，否则应该尽量避免使用 - force选项。

最后，git push不会推送标签（tag），除非使用 - tags选项。

```
1 | $ git push origin --tags
```



相关文章

- [如何创建你自己的Git服务器](#)
- [git分支初学指南](#)
- [开发者日常使用的 Git 命令](#)
- [Git详解之七：自定义Git](#)
- [Git详解之四：服务器上的Git](#)

- [Git详解之二：Git基础](#)
- [蒋鑫：为什么 Git 比 SVN 好](#)
- [Git常用命令](#)
- [Git详解之三：Git分支](#)
- [Git详解之六：Git工具](#)

发表评论

Comment form

Name*

邮箱*

网站（请以 http://开头）


评论内容*

请填写评论内容

(*) 表示必填项

[提交评论](#)

1 条评论

1.  [heu_zhujunge](#) 说道：
[2014/06/12 下午 9:48](#)

写的简单易懂

 0  0

[回复](#)

[« 一站式学习Wireshark（三）：应用Wireshark IO图形工具分析数据流
深入理解Redis主键失效原理及实现机制 »](#)

Search for:

- [本周热门文章](#)
- [本月热门文章](#)
- [热门标签](#)

0 [真正统治世界的十大算法](#)

- 1 [265行代码实现第一人称游戏引擎](#)
- 2 [浏览器CPU加速12倍](#)
- 3 [MySQL在大型网站的应用架构演变](#)
- 4 [如何避免面试中薪水要的太高或太低?](#)
- 5 [终极疯狂越野探险车和它的四岁女“指挥官”](#)
- 6 [抓包工具](#)
- 7 [站着工作未必对你的健康有好处](#)
- 8 [一站式学习Wireshark \(一\)](#)
- 9 [PHP入门、进阶与实战，外加玩转Swift!](#)

推荐关注



[Swift语言学习路径图](#)

苹果全新编程语言:快速/动态/强大/优雅，未来将取代Objective-C成为iOS主流！



[Android开发学习路径图](#)

移动互联网最时髦的职业，名企需求量极大，薪资5k-25k/月，不缺高薪缺人才！



[Cocos2d-x学习路径图](#)

全球流行的游戏开发引擎，免费/开源/跨平台/轻量级/极度，薪资6k-25k/月！



[免费中文课程：玩转Swift](#)



精心地设置了更多更加好玩儿，更加贴合实际，能够让用户们更加快速地融入到iOS应用开发中这样的实例。



[Android攻城狮的第一门课—入门篇](#)

课程的目标就是“看得懂、学得会、做得出”，为后续的学习打下夯实的基础。

最新评论（期待您也参与评论）

-  Re: [Intel/Google/Mozilla联手：浏览器CPU加速12倍](#)
ES7... 还希望能够... 唉 灯盏细辛
-  Re: [对 Linux 新手非常有用的20个命令](#)
cat那里没有用转义字符，>>被删了 沧之云
-

Re: [Objective-C开发者对Swift亮点的点评](#)

对比obj-c, swift是各种倍儿爽，不愧是现代语言。 Lex



Re: [浏览器如何赚钱：谷歌需要火狐](#)

一切都建立在Firefox 在 Chrome不断推广的前提下还能占据这么大市场份额。 whosye



Re: [程序员最头疼的事：命名](#)

我估计跟ITworld区别的原因是因为国内大部分程序猿对英文不明确而导致对命名的不重视。。。比如我，... 字来



Re: [为生活可以忍，侮辱技术行不行？](#)

是金子到哪里都会发光的，就算是为了生活，也不用一直忍。当年我也有没还完的房贷，有老婆，就是没孩子。两... 金子（化名）



Re: [有关“非计算机专业如何转行做程序员”的一点思考](#)

看了这么多评论，力劝学习计算机 三思后行。。在我看来，大多数人并不适合学习计算机，所谓的兴趣，也不过... neo



Re: [反测试无用论之三：牛逼公司的模式就一定对吗？](#)

那个放射设备的不靠谱实在不应该由测试人员负责任。那是个典型的竞争条件bug，你测一百遍没有问题，也不... ntysdd



热点文章推荐



[敢偷用我的WiFi？看我怎么治你](#)

邻居盗用我的WiFi，可以直接选择加密口令，或者...作为一名极客也可以耍耍他。



[你的电脑在偷偷连接什么网站？](#)

大家都有过网络连接变慢的经历吧？让我们一块儿揪出这些小东西吧。



[Java开发牛人十大必备网站](#)

对于Java开发牛人来说，网站的好坏取决于如何使用它们。



[100个高质量Java开发者博客](#)

收集全球范围内100个高质量Java开发者博客。

关于伯乐在线博客

在这个信息爆炸的时代，人们已然被大量、快速并且简短的信息所包围。然而，我们相信：过多“快餐”式的阅读只会令人“虚胖”，缺乏实质的内涵。伯乐在线博客团队正试图以我们微薄的力量，把优秀的原创/译文分享给读者，为“快餐”添加一些“营养”元素。

伯乐在线-博客(blog.jobbole.com) 专注于分享职业相关的博客文章、业界资讯和职业相关的优秀工具和资源。博文类别包括：程序员、设计、营销、互联网、IT技术、自由职业、创业、运营、管理、翻译和人力资源等等。期待您通过[RSS订阅](#)和[微博](#)关注我们。如果您也愿意[分享一份自己的原创/译文](#)，可以[从这里开始](#)~

联系我们

网站合作和广告投放

联系邮箱：Webmaster (at) JobBole.com

QQ： 630772296 （加好友请注明来意）

交换友情链接要求：PR>=4

网站使用问题

请直接在[这里发帖](#)询问或者反馈

欢迎关注并订阅伯乐在线博客

