目录

[一、创建 Git 仓库 2](#_Toc5972003)

[1.1 在本地目录中创建仓库 2](#_Toc5972004)

[1.2 克隆现有的仓库 2](#_Toc5972005)

[1.3、检查当前文件状态 2](#_Toc5972006)

[1.4 跟踪新文件或把文件放入暂存区 2](#_Toc5972007)

[1.5 提交暂存区 2](#_Toc5972008)

[1.6 移除文件 2](#_Toc5972009)

[二、远程仓库 3](#_Toc5972010)

[2.1 查看远程仓库 3](#_Toc5972011)

[2.2添加远程仓库 3](#_Toc5972012)

[2.3 获取远程仓库数据 3](#_Toc5972013)

[2.4推送到远程仓库 3](#_Toc5972014)

[2.5 远程仓库重命名 4](#_Toc5972015)

[2.6 删除远程仓库 4](#_Toc5972016)

[三、本地分支 4](#_Toc5972017)

[3.1 创建分支 4](#_Toc5972018)

[3.2 切换分支 4](#_Toc5972019)

[3.3 合并分支 4](#_Toc5972020)

[3.4 删除分支 5](#_Toc5972021)

[3.5 分支查询 5](#_Toc5972022)

[四、远程分支 5](#_Toc5972023)

[4.1 查看远程分支 5](#_Toc5972024)

[4.2 获取远程分支 6](#_Toc5972025)

[4.3跟踪远程分支 6](#_Toc5972026)

[4.4推送远程分支 6](#_Toc5972027)

# 一、创建 Git 仓库

## 1.1 在本地目录中创建仓库

在目录中输入：$ **git init**

## 1.2 克隆现有的仓库

$ **git clone** https://github.com/libgit2/libgit2 mylibgit

mylibgit：自定义的**仓库名**，可以省略。

## 1.3 检查当前文件状态

**$ git status**

## 1.4 跟踪新文件或把文件放入暂存区

**$ git add filename**

## 1.5 提交暂存区

**$ git commit -m "备注信息"**

**$ git commit -a -m "备注信息" ( -a 跳过暂存区，直接提交)**

## 1.6 撤销操作

**$ git commit --amend**

有时候我们提交完了才发现漏掉了几个文件没有添加，或者提交信息写错了。 此时，可以运行带有 --amend 选项的提交命令尝试重新提交

## 1.7 移除文件

**$ git rm filename**

# 二、远程仓库

## 2.1 查看远程仓库

**$ git remote -v**

**$ git remote show origin**

## 2.2 添加远程仓库

$ **git remote add** pb <https://github.com/paulboone/ticgit>

命令格式：git remote add <自定义仓库名> <url>

## 2.3 获取远程仓库数据

**$ git fetch** pb

这个命令会访问远程仓库，从中拉取所有你还没有的数据。 执行完成后，你将会拥有那个远程仓库中所有分支的引用，可以随时合并或查看。必须注意 git fetch 命令会将数据拉取到你的本地仓库 - 它并**不会**自动合并或修改你当前的工作。如果你有一个分支设置为跟踪一个远程分支，可以使用 git pull 命令来自动的抓取然后合并远程分支到当前分支。

## 2.4 推送到远程仓库

例：当你想要将 master 分支推送到 origin 远程仓库时：

**$ git push origin master**

## 2.5 远程仓库重命名

**$ git remote rename pb paul**

把 仓库名 pb 改为 paul。

## 2.6 删除远程仓库

**$ git remote rm paul**

# 三、本地分支

## 3.1 创建分支

**$ git branch 分支名**

此命令仅仅创建 一个新分支，并不会自动切换到新分支中去。

## 3.2 切换分支

**$ git checkout分支名**

下面命令为 合并3.1和3.2步骤， 即创建分支，并切换到该分支：

**$ git checkout -b 分支名**

## 3.3 合并分支

**$ git merge 分支名**

即 把 分支名 合并到**当前所在**的分支上。

## 3.4 删除分支

**$ git branch -d 分支名**

## 3.5 分支查询

**$** **git branch**

此命令，会得到当前所有分支的一个列表。

**$ git branch –v**

此命令查看每一个分支的最后一次提交。

**$ git branch --merged** **（两个横杠 - ）**

查看已经合并到了当前分支的分支，\*所在的为当前分支，其它分支已经合并到了该分支 是可以删除的。

**$ git branch --no-merged （no 前面是两个横杠 - ）**

查看没有合并到当前分支的分支。

# 四、远程分支

## 4.1 查看远程分支

**git ls-remote (远程仓库名)**

显式地获得远程引用的完整列表。

**git remote show (远程仓库名)**

获得远程分支的更多信息。

## 4.2 获取远程分支

**$ git fetch 远程仓库名**

当 git fetch 命令从服务器上抓取本地没有的数据时，它并不会修改工作目录中的内容。 它只会获取数据然后让你自己合并。即： fetch下来的远程分支是一个“标签”信息，无法对其编辑。**需设置相应的本地分支对其跟踪。**

## 4.3跟踪远程分支

**方法一：**

**$ git checkout -b serverfix origin/serverfix**

创建一个serverfix的本地分支，用来跟踪远程origin/serverfix分支。使其可以在本地编辑。

**$ git checkout** --**track origin/serverfix**

这个命令是上面命令的简化，即在本地建立一个**serverfix**的分支，用来跟踪**origin/serverfix。**

**方法二：**

**$ git merge origin/serverfix**

把远程分支合并到当前的本地分支。

**git pull** 相当于**git fetch 与 git merge** 两个命令的**合并**。它会查找当前分支所跟踪的服务器与分支，从服务器上抓取数据然后尝试合并入那个远程分支。

**$ git branch -u origin/serverfix**

**查看远程分支origin/serverfix所对应的是哪个本地分支对其跟踪。**

## 4.4推送远程分支

**$ git push origin serverfix**

把本地分支serverfix，推送到远程仓库 origin 的远程分支**serverfix里。**

**$ git push origin serverfix:somebranch**

来将本地的 serverfix 分支，推送到远程仓库**origin**上的 somebranch里。（somebranch：自定义分支名）

## 4.5删除远程分支

**$ git push origin** --**delete serverfix**