# 在Linux上搭建Git环境(基于Github)

* **第一步：注册帐号，创建仓库。**

要想使用github第一步当然是注册github账号了， github官网地址：<https://github.com/>。 之后就可以创建仓库了（免费用户只能建公共仓库），Create a New Repository，填好名称后Create，之后会出现一些仓库的配置信息，这也是一个git的简单教程。

* **第二步：Github 安装**

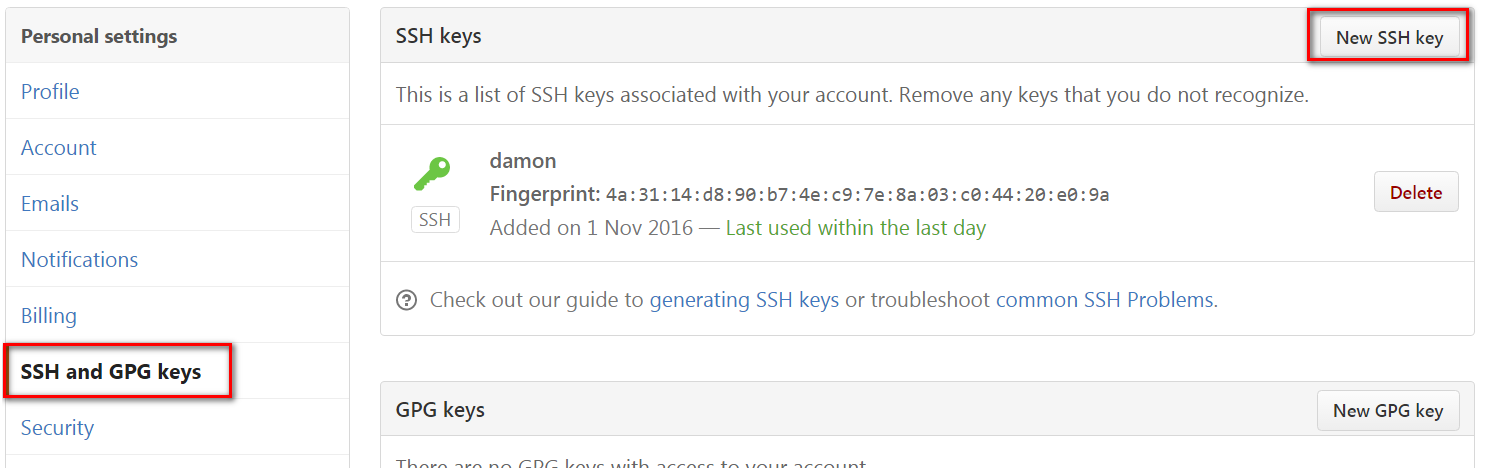
[下载 git Linux 版](https://git-scm.com/downloads)

* **配置Git**

首先在本地创建ssh key；

$ ssh-keygen -t rsa -C "youremail"

后面的youremail改为你在github上注册的邮箱(ldmmdl@163.com)，之后会要求确认路径和输入密码，我们这使用默认的一路回车就行。成功的话会在~/下生成.ssh文件夹，进去，打开id\_rsa.pub，复制里面的key。回到github上，进入 Account Settings（账户配置），左边选择SSH Keys，Add SSH Key,title随便填，粘贴在你电脑上生成的key。



为了验证是否成功，在git bash下输入：

$ ssh -T git@github.com

如果是第一次的会提示是否continue，输入yes就会看到：You've successfully authenticated, but GitHub does not provide shell access 。这就表示已成功连上github。

接下来我们要做的就是把本地仓库传到github上去，在此之前还需要设置username和email，因为github每次commit都会记录他们。

$ git config --global user.name "your name"

$ git config --global user.email "your\_email@youremail.com"

要添加远程地址，首先，创建新文件夹，打开，然后执行 git init 以创建新的 git 仓库。

然后输入以下命令添加远程地址。

$ git remote add origin git@github.com:yourName/yourRepo.git

后面的yourName和yourRepo表示你再github的用户名和刚才新建的仓库，加完之后进入.git，打开config，这里会多出一个remote "origin"内容，这就是刚才添加的远程地址，也可以直接修改config来配置远程地址。

* **创建分支**

$ git remote update //更新本地分支表

$ git branch –r //查看服务器端（github）上面已经有的branch

$ git checkout –b new\_branch //创建新的分支

$ git push origin new\_branch //将新的分支推送到github仓库

$ git push origin :branch\_name //删除服务器端的分支

$ git checkout –d branch\_name //删除本地的分支

* **至此，环境搭建完成。接下来进行如此操作即可。**

## 在GIT hub上面clone一个已经存在的仓库

1. 按照上面的步骤安装好环境。
2. 利用git clone [git@github.com:Damonnnn/Developer-Career.git](mailto:git@github.com:Damonnnn/Developer-Career.git) （注意一定要以ssh方式）
3. 直接git checkout branch\_name切换这个仓库已有的branch。
4. 修改，提交即可

# GIT常用命令集合

|  |  |
| --- | --- |
| 命令 | 解释 |
| git init | 创建一个空仓库。 |
| git clone https://github.com/YOUR-USERNAME/YOUR-REPOSITORY  eg: git clone git@github.com:Damonnnn/Developer-Career.git | 从远端克隆一个仓库。（可以不用git init也可以）。注意一定要以ssh的方式clone，否则每次都需要输入密码 |
| git remote update | 更新本地分支表。 |
| git branch –r | 查看服务器端（github）上面已经有的branch。 |
| git checkout –b new\_branch | 创建新的分支。 |
| git checkout –d branch\_name | 删除本地的分支。 |
| git checkout --track origin/ branch\_name | 在本地创建branch\_name分支并track远端同名的分支。其实就是建立映射，类似与master映射origin/master |
| git checkout branch | 切换分支 |
| git push origin :branch\_name | 删除服务器端的分支。 |
| git push origin new\_branch | 将新的分支推送到github仓库。 |
| git add –A | 全部添加改动过的文件。 |
| git commit –m “description” | 提交改动。 |
| git push origin branch\_name | 将改动推送到远端。 |
| git status | 查看当前状态 |
| git pull | 更新本地仓库至最新 |
| git pull origin branch\_name | 从远端更新自本地仓库，好像比上面的厉害。 |
| git log | 查看当前状态 |
| git fetch origin  git reset –hard origin/branch\_name | 放弃本地改动。重新在服务器抓取最新的code |
| git log  git tag 1.0.0 1b2e1d63ff | 利用git log查看commit ID，即1b2e1d63ff。  利用git tag创建一个标签1.0.0. |
|  |  |

# Git ignore 规则

 ignore files的三种方法

以下涉及的ignore文件均为如下格式：

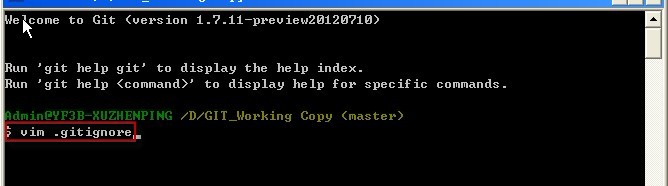
|  |
| --- |
| # 以'#'开始的行，被视为注释.  # 忽略掉所有文件名是 foo.txt的文件.  foo.txt  # 忽略所有生成的 html文件,  \*.html  # foo.html是手工维护的，所以例外.  !foo.html  # 忽略所有.o和 .a文件.  \*.[oa] |

【方式一】

在仓库目录下新建一个名为**.gitignore**的文件（因为是点开头，没有文件名，没办法直接在windows目录下直接创建，必须通过右键[Git](http://lib.csdn.net/base/git) Bash，按照[Linux](http://lib.csdn.net/base/linux)的方式来新建.gitignore文件）。如下图所示。

.gitignore文件对其所在的目录及所在目录的全部子目录均有效。通过将.gitignore文件添加到仓库，其他开发者更新该文件到本地仓库，以共享同一套忽略规则。





【方式二】

       通过配置.git/info/exclude文件来忽略文件。这种方式对仓库全局有效，只能对自己本地仓库有作用，其他人没办法通过这种方式来共享忽略规则，除非他人也修改其本地仓库的该文件。

【方式三】

       通过.git/config配置文件的core. Excludesfile选项，指定一个忽略规则文件（完整路径），如下图所示。忽略规则在文件e:/gitignore.txt中（当然该文件名可以任意取）。

       该方式的作用域是也全局的。



【例子】

# 忽略\*.o和\*.a文件

 \*.[oa]

# 忽略\*.b和\*.B文件，my.b除外

\*.[bB]

!my.b

# 忽略dbg文件和dbg目录

dbg

# 只忽略dbg目录，不忽略dbg文件

dbg/

# 只忽略dbg文件，不忽略dbg目录

dbg

!dbg/

# 只忽略当前目录下的dbg文件和目录，子目录的dbg不在忽略范围内

/dbg