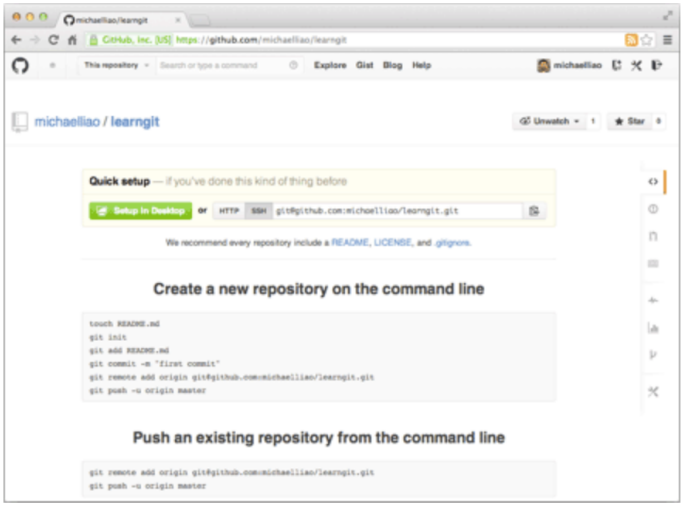
# git教程(4) 远程仓库

一 添加远程仓库
1 现在的情景是，你已经在本地创建了一个git仓库，又想在 github 创建一个git仓库，并让 2 个仓库进行远程同步。
这样一来，GitHub上的仓库不仅可作为备份，还可以让其他人通过该仓库去协同办公！！

1.1 先在GitHub上创建一个仓库。“create repository“

在Repository name填入`learngit`，其他保持默认设置，点击“Create repository”按钮，就成功地创建了一个新的Git仓库：




1.2 接着把已有的本地仓库与刚刚在github创建的仓库进行关联。【git remote add origin git@github.com:自己的git账号名/learngit.git】
在本地的 laerngit仓库【即目录】下运行命令：
git remote add origin git@github.com:自己的git账号名/learngit.git【”同一个本地仓库绑定多个，覆盖？？“】

```
$ git remote add origin git@github.com:michaelliao/learngit.git
```

注意：要替换成自己GitHub账号名，否则就将自己的本地仓库关联到“博主的远程仓库”上了！！即使关联了，我们也无法推东西过去，因为我们的 SSH Key公钥不在“博主”的账号列表中！！！

1.3 添加后，远程库的名字【origin】就是 origin，这是 git 的默认叫法！！可以改，但是为了约定熟成就这样叫着。

1.4 接着把本地仓库的所有东西推送到远程库上。【git push -u origin master （为啥要有 -u？？？ origin上的master分支？？！）】

```
$ git push -u origin master
Counting objects: 20, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (15/15), done.
Writing objects: 100% (20/20), 1.64 KiB | 560.00 KiB/s, done.
Total 20 (delta 5), reused 0 (delta 0)
remote: Resolving deltas: 100% (5/5), done.
To github.com:michaelliao/learngit.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

注意：由于远程库是空的【需加上 -u 参数！！绑定远程的master分支。“set-upstream？？“】，我们 第一次推送master分支时，加上-u参数，表示 git不但会把本地的master分支内容推送到远程新的master分支，还会将 2个 master分支 进行一个关联！！ ---> 以后推拉就可以简写，如 git push || pull。

1.5 从现在起，只要本地做了修改、提交，就可以直接 git push origin master【将最新修改内容同步到远程库上！！】

2 SHH警告【？？？】

当你第一次使用Git的`clone`或者`push`命令连接GitHub时，会得到一个警告：

```
The authenticity of host 'github.com (xx.xx.xx.xx)' can't be established.
RSA key fingerprint is xx.xx.xx.xx.xx.
Are you sure you want to continue connecting (yes/no)?
```

这是因为Git使用SSH连接，而SSH连接在第一次验证GitHub服务器的Key时，需要你确认GitHub的Key的指纹信息是否真的来自GitHub的服务器，输入`yes`回车即可。

Git会输出一个警告，告诉你已经把GitHub的Key添加到本机的一个信任列表里了：

```
Warning: Permanently added 'github.com' (RSA) to the list of known hosts.
```

这个警告只会出现一次，后面的操作就不会有任何警告了。

如果你实在担心有人冒充GitHub服务器，输入`yes`前可以对照GitHub的RSA Key的指纹信息是否与SSH连接给出的一致。

3 小结

3.1 关联一个远程库命令：git remote add origin git@server_name:path【用户名？？】/repo_name【仓库名】.git

3.2 第一次将本地的master分支内容推送到远程的master分支【不要忘了带上 -u参数！！是因为第一次推送、所以要加上 -u参数？？？不带会输出啥？？？】： git push -u origin master！！

3.3 此后，就可以 不用带上-u参数了，直接 git push origin master 推送最新的修改内容！！【不过一般做法是本地仓库也建立一个master分支，每次开发前，都将 远程master内容pull到本地，预计将用于开发的分支进行merge，接着才是该非master分支上进行开发，修改完

了在 add、commit、push到远程，向远程master发起合并！！！】

3.4 分布式版本控制系统最大好处之一 —— 没有联网【集中式版本控制如SVN没有网是不能干活的】也可以工作【等到有网了，再把本地进行提交、推送即可！！】，不用考虑远程库的存在！！

二 从远程库克隆
1 克隆远程库
1.1 远程库准备好后，git clone命令去将远程库克隆到本地、形成本地仓库！！
如 git clone git@github.com:账号名/仓库名.git【与
git remote add origin git@xxx 很相似！！】

```
$ git clone git@github.com:michaelliao/gitskills.git
Cloning into 'gitskills'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 3
Receiving objects: 100% (3/3), done.
```

2 如果多人协作开发，那么每个人各自从 远程克隆一份即可！！

3 git支持多种协议。
默认是 git:// 使用SSH，但也可以使用 https【https除了速度慢以外，最大麻烦就是每次推送都必须输入口令（毕竟安全嘛？？！）】等其他协议。

4 小结
4.1 要克隆一个远程仓库，必须知道其地址，然后
git clone git@gihub.com:user_name/repo_name.git 接口

4.2 git支持多种协议，包括https，但通过 ssh 支持的原生git速度最快！！

完