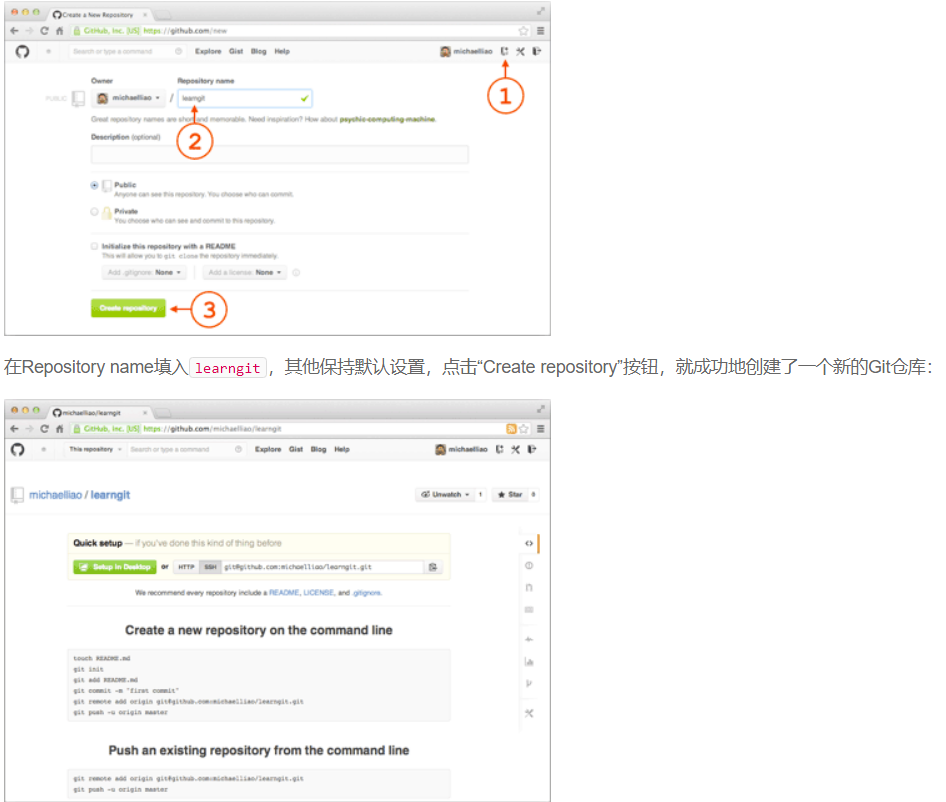
1、关于远程库的搭建、使用：

（ps：项目名称最好与本地库保持一致）

1. 创建远程库：
2. 远程库与本地库连接：

git remote add origin [git@gihub.com:Zmq1294550676/<](mailto:git@gihub.com:Zmq1294550676/%3c)项目名>.git

（origin：为远程库的名字，可以修改为其他的）

1. 推送方式：

第一次：git push -u origin master

以后使用简化命令：git push origin master

2、克隆远程库：(修改代码（Debug）使用的方式)

命令：git clone git@github.com:Zmq1294550676/gitskills.git

3、提交文件到远程库：

使用git push origin master :

（1）必须先提交本地文件

命令：git add <file\_name> 添加文件，使用\*\*表示所有

git commit –m “描述”

（2）再进行push，推送到远程库：git push origin 分支名

4、Git分支合并：

当Git无法自动合并分支时，就必须首先解决冲突。解决冲突后，再提交，合并完成。解决冲突就是把Git合并失败的文件手动编辑为我们希望的内容，再提交。

用git log --graph --pretty=oneline --abbrev-commit命令可以看到分支合并图。

# 常用命令：

1. 分支操作：

* 创建分支：git branch 分支名
* 查看分支：本地分支：git branch

远程分支：git branch -r

* 转到对应分支：git checkout 分支名

git swithc 分支名

* 创建并切换到新分支：git checkout -b 分支名

git swithc –c 分支名 **（新git推荐使用）**

* 删除分支：git branch **–d** 分支名

（强行删除）git branch **–D** 分支名

* 合并（分支1）到当前所在的分支：git merge 分支1

1. 查看Git本地仓库的状态：git status
2. 查看分支历史：

（简化版，图形）git log --graph --pretty=oneline --abbrev-commit

（或者，详细版） git log

1. **多人协作：**

* 查看远程仓库的信息：git remote

（详细版）git remote -v

**远程仓库默认名称：origin**

* **创建分支让其他人开发：**其他人克隆新分支进行开发，再push到新分支。

**命令：**git checkout -b dev 远程仓库名/新分支名

**例如：**git checkout -b dev origin/dev

* **拉取远程仓库：命令：**
* **拉取默认分支内容：**git pull
* **拉取特定分支内容：先设定要获取的分支，再用git pull**

**设定分支：**git branch --set-upstream-to=远程仓库名/分支名 本地分支名

**例如：**git branch --set-upstream-to=origin/dev dev

1. 标签：默认为HEAD

* 添加标签：git tag 标签名
* 创建带有备注文字的标签：

git tag -a 标签名 -m "说明文字" 提交的ID （即：commit ID）

* 查看标签：git tag
* 查看对应标签的说明：git show 标签名
* 推送标签的远程：
* 推送特定标签：git push 分支名 标签名

**例如：**git push origin V1.0

* 一次性推送所有标签：git push 分支名 –-tags
* 删除**本地**标签：git tag –d 标签名
* 删除远程标签：git push 分支名 :refs/tags/标签名

**例如：**git push origin :refs/tags/V1.0