**Git 学习**

# Git介绍

- 80%的程序员必会技能

- 提高代码开发效率

- 多人协作

- 资源共享

- 备份文件（历史状态）

Git版本管理工具的作用：

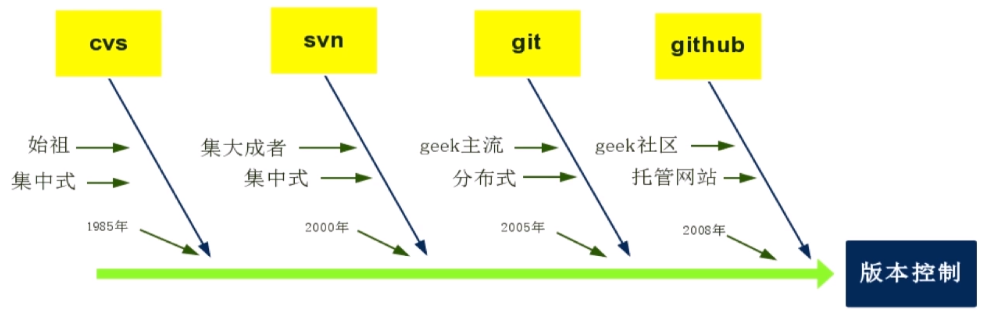
1、备份文件（类似U盘，网盘）；

2、记录历史（有各个阶段的代码历史版本）；

3、多端协作（可以在家工作，也可以在公司）；

4、团队协作（代码整合，多人共同完成一个项目）

版本控制工具发展史：



1：cvs 1985,集中式, 始祖 （网络环境）

2：svn 2000，集中式 , 集大成者 (网络环境)

3：git 2005，分布式，geek主流 (无网环境)

4：GitHub 2008，托管网站，geek社区 (程序猿托管网络)

集中式：服务器 分布式：效率高

# Git安装

1. 下载 https://git-scm.com

2. 配置用户信息

git config --global user.name “用户名”

git config –global user.email “邮箱号”

3. git client管理工具 Sourcetree： https://www.sourcetreeapp.com/

# Git 仓库（也叫版本库）

简单理解就是一个目录，该目录下的所有资源都能被Git管理，每个文件的删除与修改都能被Git追踪。这样就便于使用Git还原到文件任意状态。

以下命令都是在某个特定的仓库下执行的：

- 初始化版本库（本地仓库）

**git init**

- 添加文件到版本库

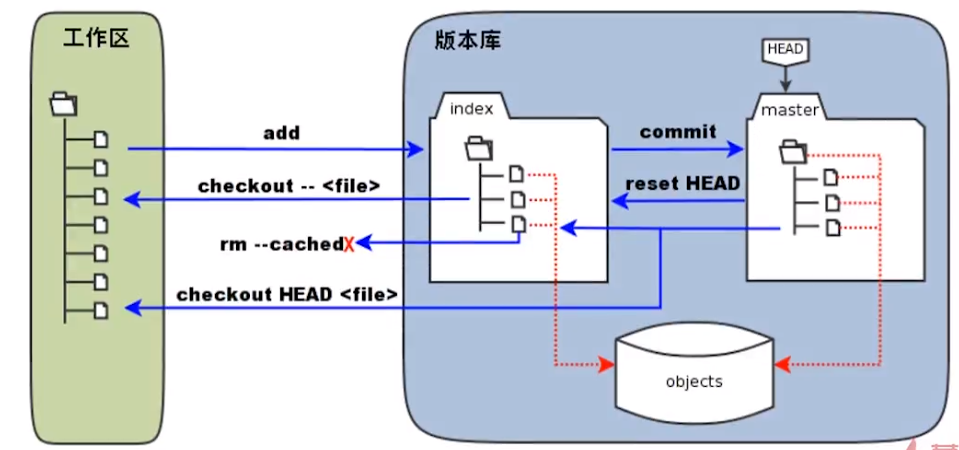
**git add [文件/目录]**

**git commit -m “描述信息”** 提交

- 查看仓库状态（主要是显示当前仓库有哪些文件变动过但未提交）

**git status**

# Git工作流



**工作区**：当前git指令操作的目录

**index区**：暂存区，可通过git add 指令将文件保存到暂存区

**master区**：版本库，可通过git commit -m 指令将暂存区的文件提交给版本库管理。在这里的文件可以用Git追踪，使恢复到任意提交的状态。

# 远程仓库

之前针对的都是本地仓库（也叫版本库的操作），但是我们也可以使用Git将资源推送到远程仓库（云空间），也可以将远程的资源同步或克隆到本地仓库，这样做的好处有以下几点：

- 可以节约硬盘空间

- 便于多人协作完成某个项目开发（共享项目资源）

- 资源共享

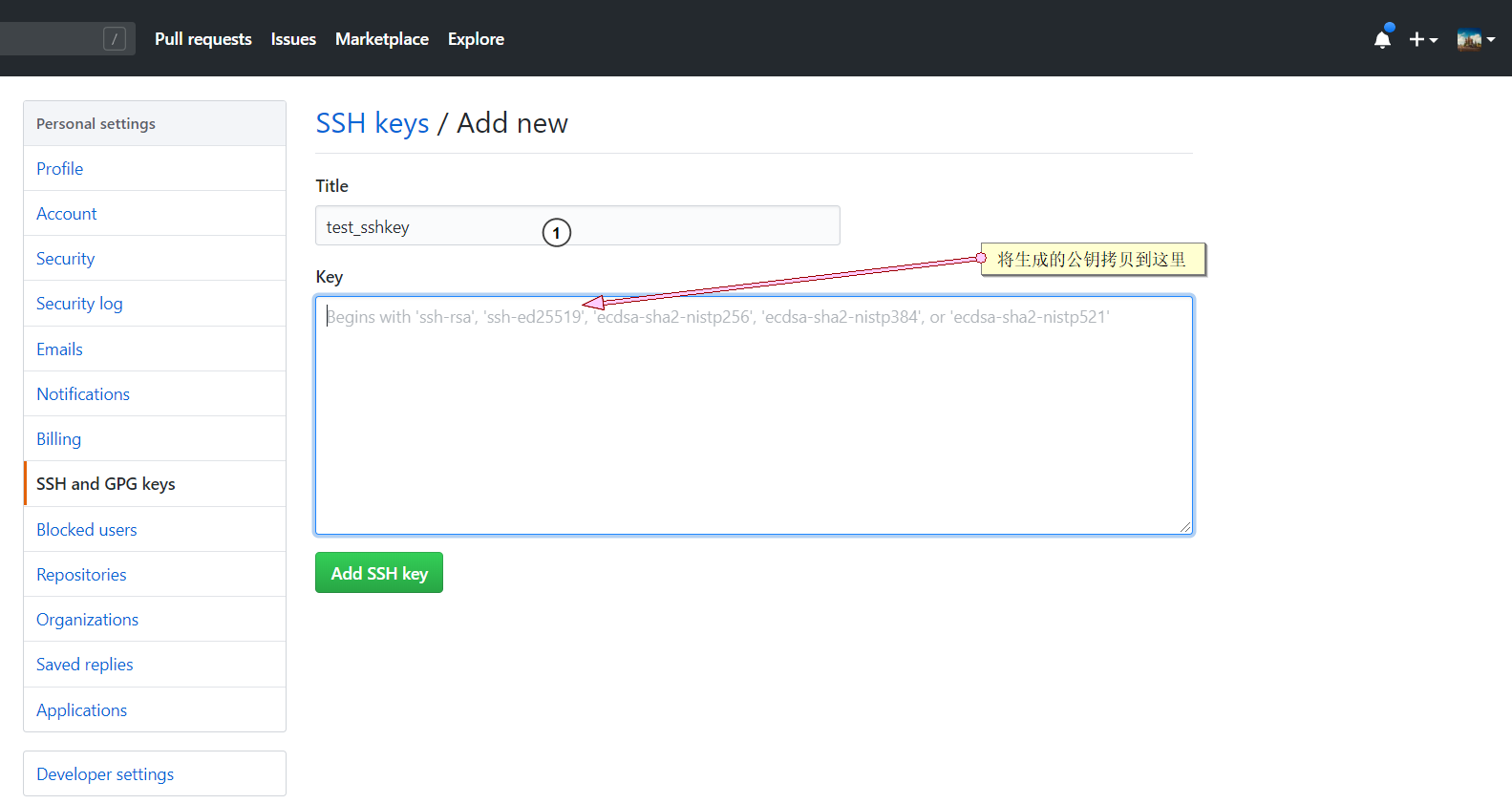
**怎么样建立远程仓库（以github为例）：**

1. 申请github账号 https://github.com/join

2．创建SSH key

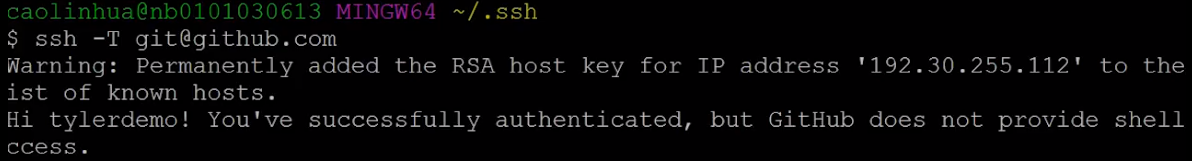
- ssh-keygen -t rsa -C “注册的github账号邮箱”

- 进入生成的SSH key的目录，将生成id\_rsa.pub（公钥）复制到github个人账户的设置的ssh位置，即如下图：



- 测试能否连上远程仓库

ssh -T git@github.com



**推送代码到远程仓库：**

1. 将本地仓库与远程某个仓库关联

*git remote add origin 远程github仓库URL*

2. 将资源推送到远程仓库

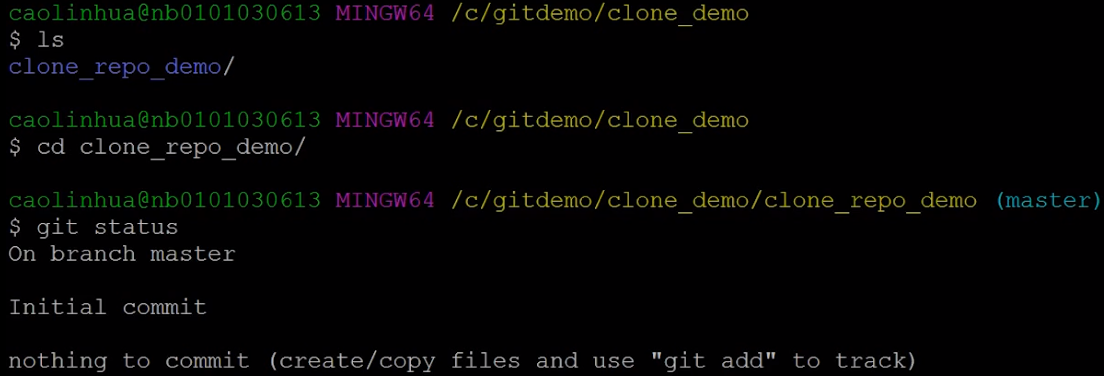
*git push -u origin master*

-u：表示以后推送都使用该远程仓库（git push）



**克隆到本地仓库：**

*git clone远程github仓库URL*



注：一定要在本地的未存在Git仓库的目录下克隆（最好是空目录）

# 分支管理

分支在多用户协作完成同一个项目时很常见，比如每个开发者可以建立一个自己的分支，这样在你没有推送代码时其他用户就不能看见你写的代码

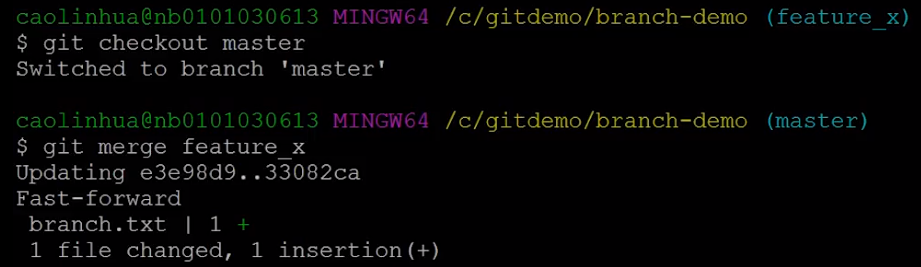
git branch 查看所有分支

git branch name 新建分支

git branch -d name 删除指定分支

git merge name 合并分支到当前所在分支

git checkout name 切换到指定分支



# 标签管理

查看所有标签：git tag

创建标签：git tag *name*

指定提交信息：git tag -a *name* -m *“你的内容”*

删除标签：git tag -d *name*

标签发布：git push origin *name*



# 总结

**工作流**：工作区 --> 暂存区 --> 版本库

**初始化**：git init --> git add --> git commit

**远程仓库**：git remote add --> git pull --> git push --> git clone

**分支管理**：git branch --> git checkout --> git merge

**标签管理**：git tag --> git push