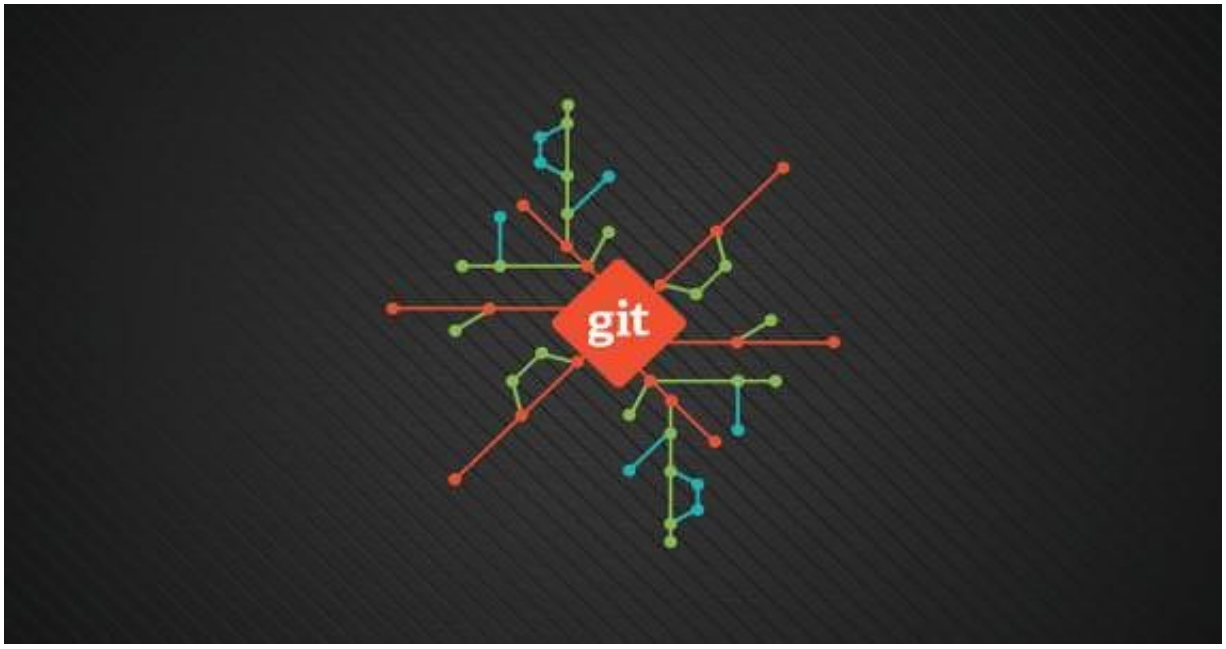
Introduction to git and github



1.简介（Overview）

·Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.
·Git is easy to learn and has a tiny footprint with lightning fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.
·git不仅仅是个版本控制系统，它也是个内容管理系统(CMS),工作管理系统等。

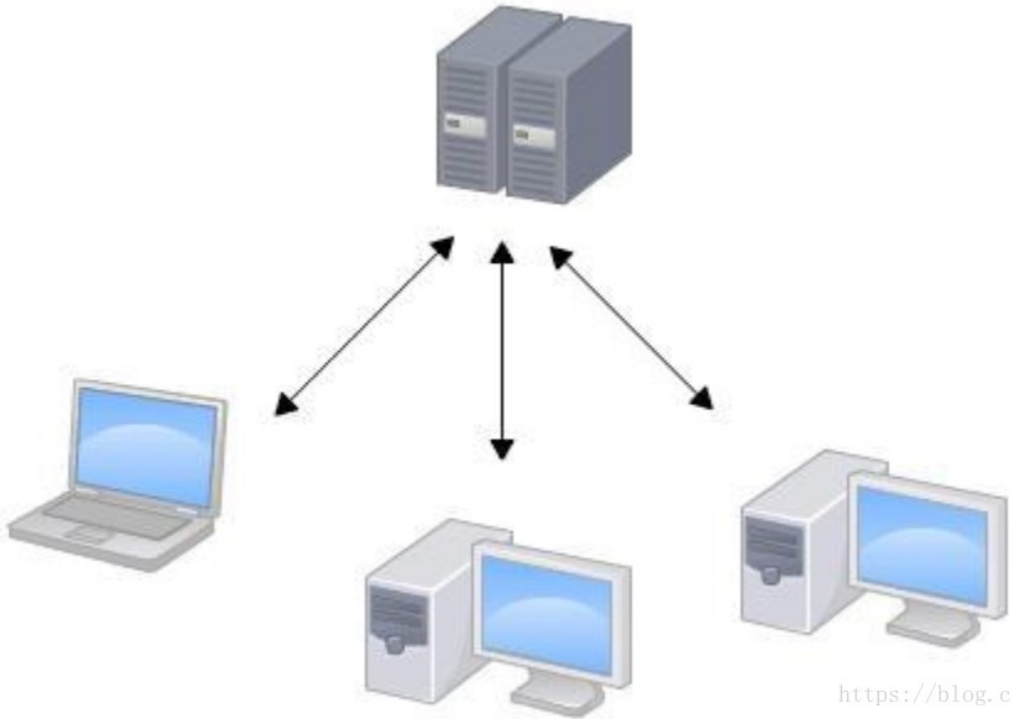
2.版本控制系统(version control system)

2.1 版本控制系统是干什么的？

工程设计领域中使用版本控制管理工程蓝图的设计过程。 在 IT 开发过程中也可以使用版本控制思想管理代码的版本迭代.

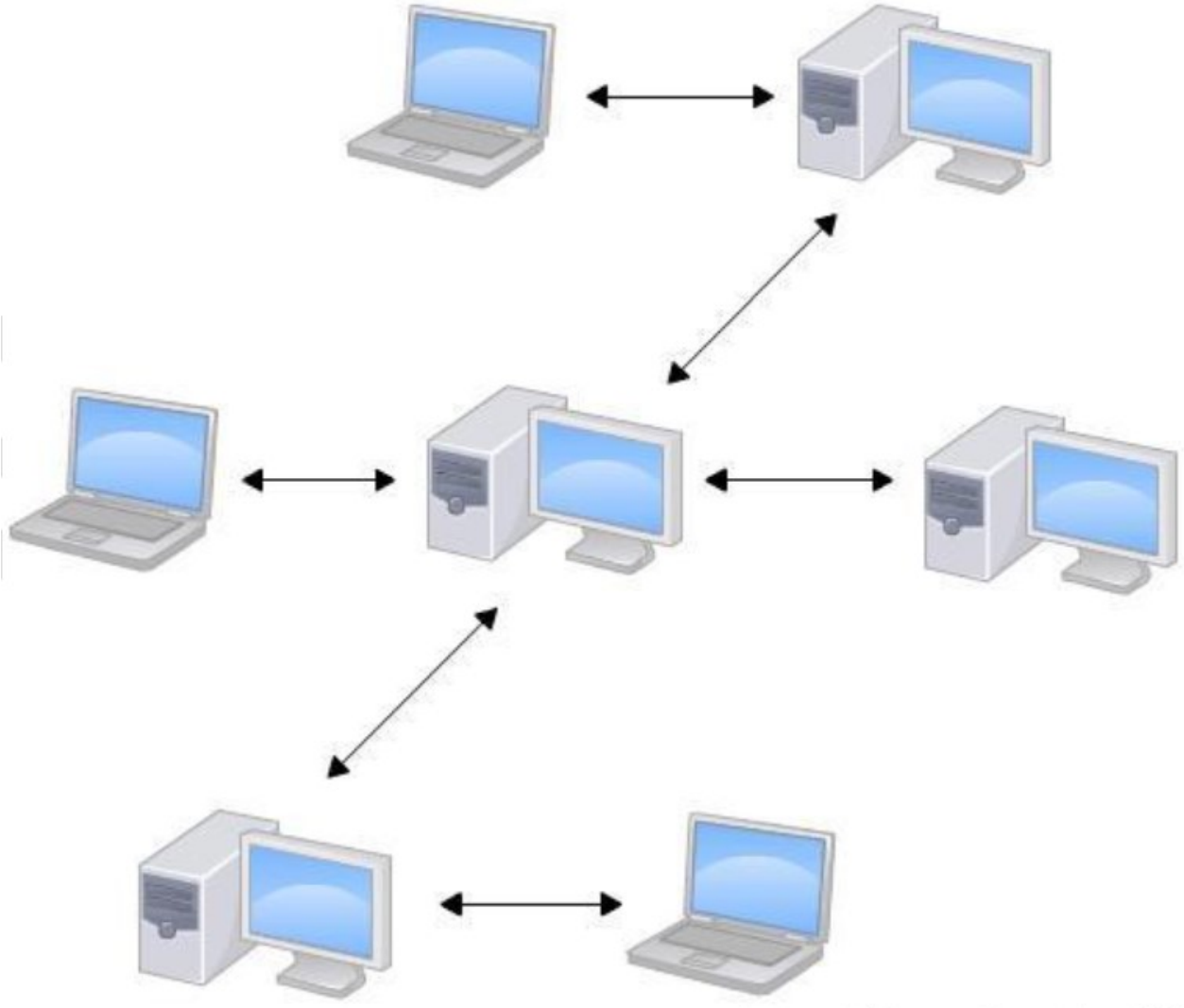
2.2版本控制工具

1.集中式版本控制系统（Centralized Version Control System，CVCS)
CVS,SVN,VSS and so on



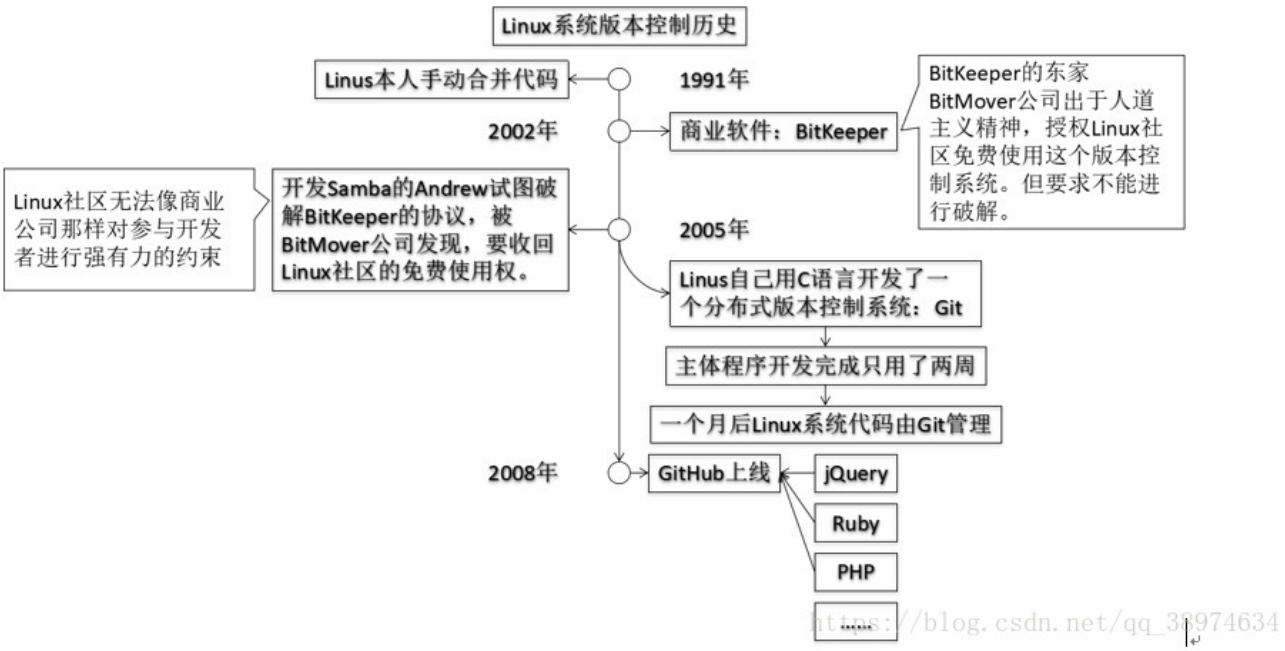
2.分布式版本控制系统（Distributed Version Control System，DVCS)

Git,Mercurial、 Bazaar、 Darcs and so on



3.Git的简介

3.1 Git的历史



3.2 官网

git的官网 ：https://git-scm.com/
github的官网 ：https://github.com/
gitlab的官网 ：https://about.gitlab.com/

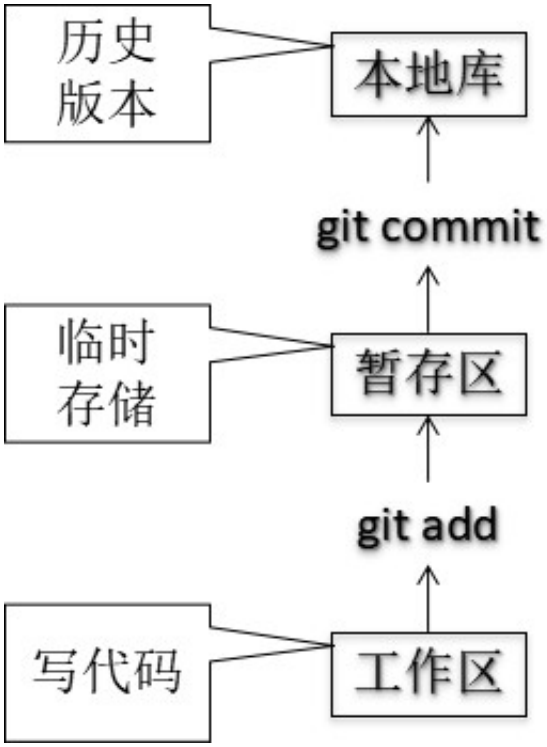
3.3 Git的优势

1 大部分操作在本地完成，不需要联网
2 完整性保证
3 尽可能添加数据而不是删除或修改数据
4 分支操作非常快捷流畅
5 与 Linux 命令全面兼容

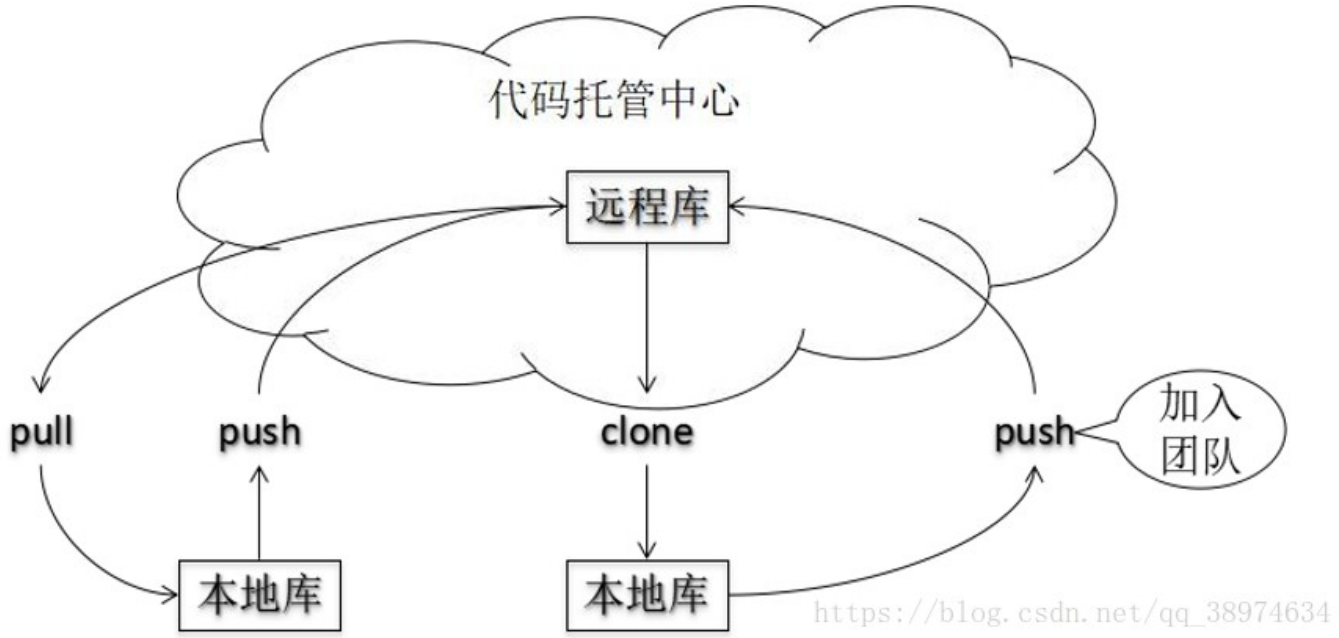
3.4 Git的安装

自行安装

3.5 Git的结构



3.6 Git和代码托管中心



这里简单说一下github和gitlab区别

- 外网环境下 GitHub 码云
- 局域网环境 GitLab服务器（相当于个人的github）

4.git操作

4.1本地库初始化

首先我在自己的工作区创建一个目录gitDev，专门用来存放gitDev这个项目，将gitDev比作我将要开发的项目。

`mkdir gitDev`

进入创建好的目录，pwd看下我的位置
~~接下来要开始表演子~~

4.1初始化

`git init`

```
➜  gitDev git init
初始化空的 Git 仓库于 /home/wx/gitDev/.git/
➜  gitDev git:(master)
```

`ls -la`

```
➜  gitDev git:(master) ls -la
总用量 12
drwxrwxr-x  3 wx wx 4096 12月 14 20:40 .
drwxr-xr-x 55 wx wx 4096 12月 14 20:40 ..
drwxrwxr-x  7 wx wx 4096 12月 14 20:40 .git
➜  gitDev git:(master)
```

查看，多了.git的隐藏文件，说明已经初始化成功了

4.2设置签名

```
git config --global user.name xxxx
git config --global user.email xxxx
```

4.3 提交

`git statue`

```
➜  gitDev git:(master) git status
位于分支 master

初始提交

无文件要提交（创建/拷贝文件并使用 "git add" 建立跟踪）
➜  gitDev git:(master)
```

创建一个文件 test
再看看

```
→ gitDev git:(master) git status
位于分支 master

初始提交

未跟踪的文件:
  （使用 "git add <文件>..." 以包含要提交的内容）

        test

提交为空，但是存在尚未跟踪的文件（使用 "git add" 建立跟踪）
```

`git add [file]`

```
→ gitDev git:(master) ✗ git add test
→ gitDev git:(master) ✗ git status
位于分支 master

初始提交

要提交的变更:
  （使用 "git rm --cached <文件>..." 以取消暂存）

        新文件:    test
```

这只是存在于 暂存区 需要commit提交`

`git commit -m "comment about thsi commit"`

```
→ gitDev git:(master) ✗ git commit -m "My First Commit" test
[master （根提交） e6534ef] My First Commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 test
→ gitDev git:(master) git status
位于分支 master
无文件要提交，干净的工作区
```

4.4查看log

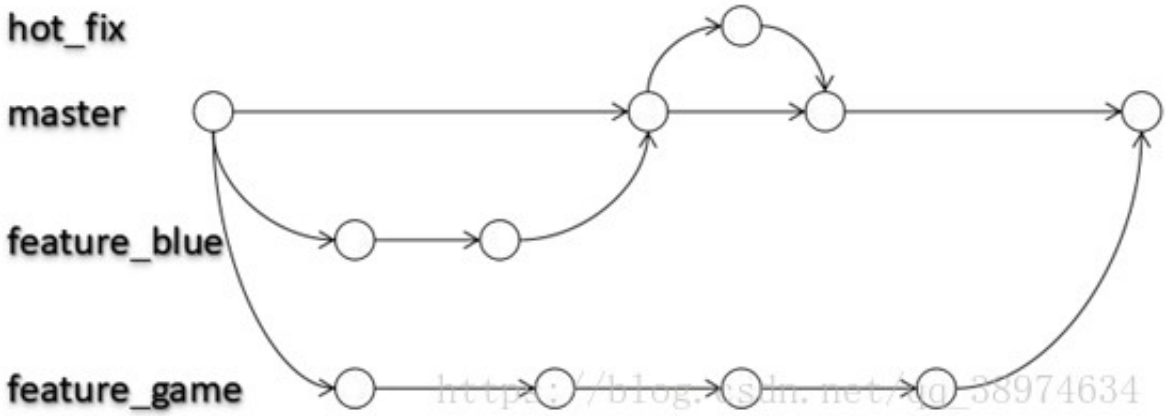
`git log --graph -- pretty="oneline"`

用最简短的图信息显示提交的历史.

`git reflog`

所有提交操作的参考记录.

4.5分支管理



分支可以理解多个功能同时推进，最后再合并在团队开发中有很大的作用

好处：

1. 同时并行推进多个功能开发，提高开发效率
2. 各个分支在开发过程中，如果某一个分支开发失败，不会对其他分支有任何影响。失败的分支删除重新始即可。

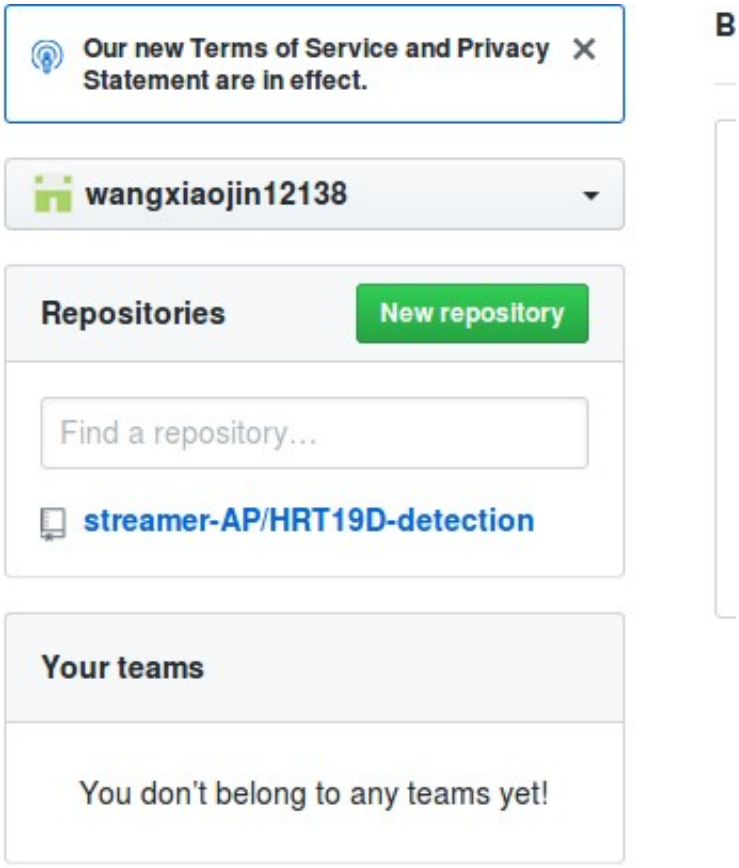
下面是分支的基本操作：

1. 创建分支
   `git branch [name]`
2. 查看分支
   `git branch -v`
3. 切换分支
   `git checkout [name]`

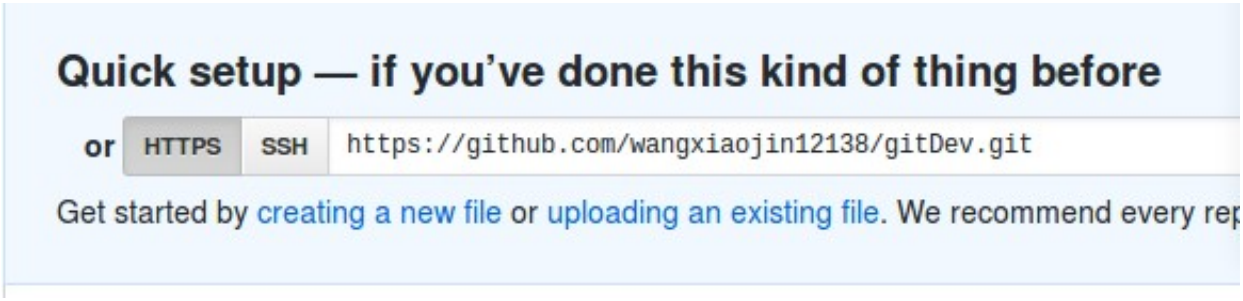
```
→  gitDev git:(master) git branch hot_fix
→  gitDev git:(master) git branch -v
  hot_fix e6534ef My First Commit
* master  e6534ef My First Commit
→  gitDev git:(master) git checkout hot_fix
切换到分支 'hot_fix'
→  gitDev git:(hot_fix) git branch -v
* hot_fix e6534ef My First Commit
  master  e6534ef My First Commit
```

5.github

5.1在github创建远程库



远程库可选**https**协议或者**ssh**协议

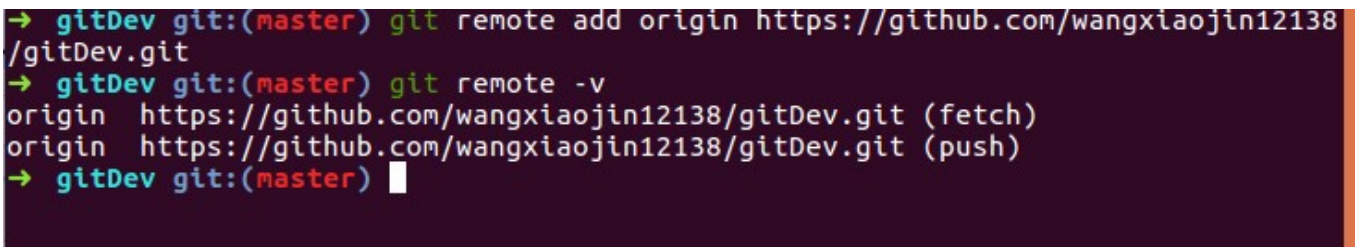


5.2在本地添加远程库

```
git remoteadd [name] [remote-address]
```

5.3查看远程库

```
git reomte -v
```

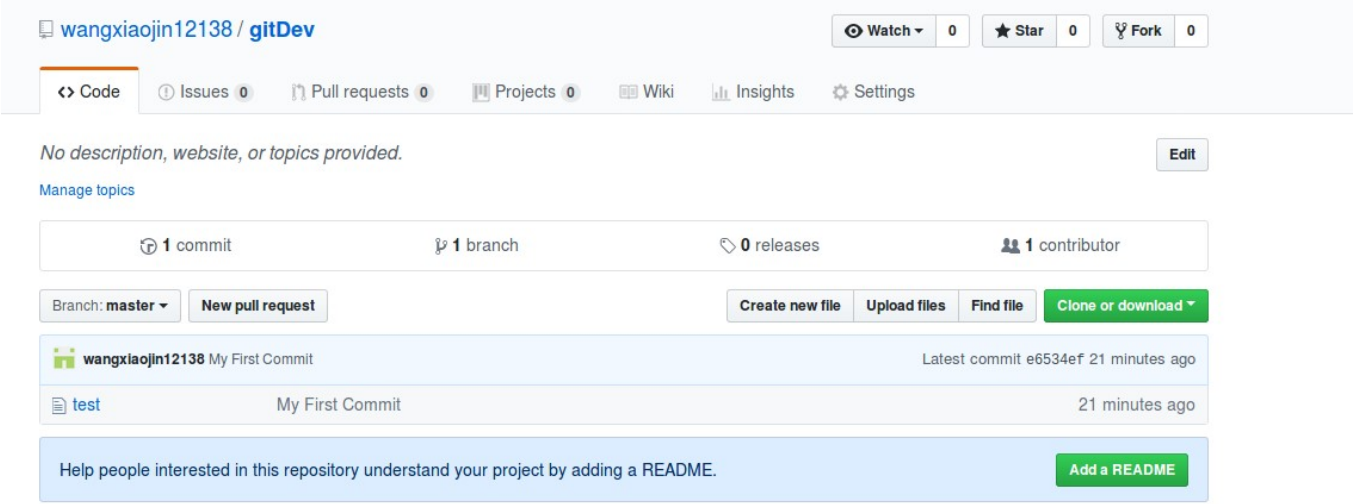


5.4将本地库**push**到远程库中

```
git push origin master
```

```
➜  gitDev git:(master) git push origin master
Username for 'https://github.com': wangxiaojin12138
Password for 'https://wangxiaojin12138@github.com':
对象计数中: 3, 完成.
写入对象中: 100% (3/3), 205 bytes | 0 bytes/s, 完成.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/wangxiaojin12138/gitDev.git
 * [new branch]      master -> master
➜  gitDev git:(master)
```

在github中查看



5.5如果要下载已经有的库

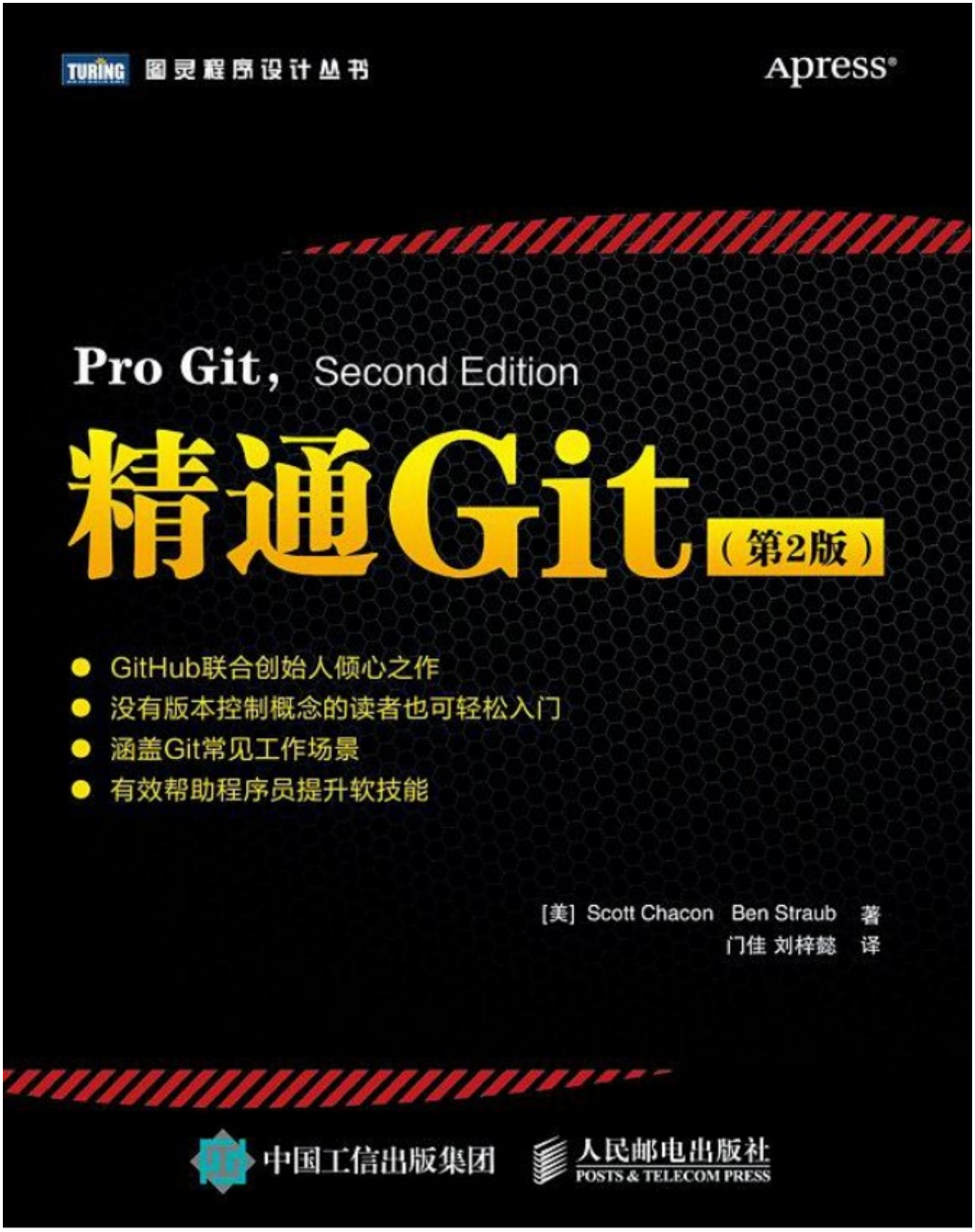
```
git clone
```

```
➜  gitDev git:(master) ✗ git clone https://github.com/wangxiaojin12138/gitDev
正克隆到 'gitDev'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
展开对象中: 100% (3/3), 完成.
检查连接... 完成。
➜  gitDev git:(master) ✗
```

6.总结

这里仅仅讲解了git和git最基本的概念和最基础的用法

想要更多的了解可以翻阅这本书



推荐教程网站

官网：https://git-scm.com/book/en/v2

菜鸟教程-github: http://www.runoob.com/w3cnote/git-guide.html

菜鸟教程-git: http://www.runoob.com/git/git-tutorial.html

廖雪峰：https://www.liaoxuefeng.com/wiki/001373951630592960 6dd18361248578c67b8067c8c017b000

HRT_19D的团队协同将通过git和github来完成
git和github作为coder的开发利器 不可不掌握