**Git原理和分支**

电子与信息工程学院 2017级微电子科学与工程 刘嘉琦 17311037

**原理：**

Checkout

缓存区

本地仓库

reset

工作区间

Fetch

远程仓库

add

commit

push

**工作区间:** 是我们创建的工程文件，在系统的编辑器可以直接看到；

**缓存区:** 相当于是一个工作区间和本地仓库的中转站，能够用他处理冲突，暂时存储文件，只能通过GUI或窗口显示；

**本地仓库:** 是连接本地代码跟远程代码的枢纽，也是只能在shell 窗口显示，代码可以暂时存储在本地仓库而不提交到网络上。

**远程仓库:** 一个远程的负责存储我们代码的服务器，最经典/出名的是Github ，在远程存储文件，可以与他人通过远程的服务器来共享代码，是多人共同完成项目的关键。

在本地创建本地仓库或者增加文件时，如命令行输入git init命令后首先在本地仓库产生更改，如果输入git add则会将文件保存到缓存区，再次执行git commit -m “message” 即可将代码从缓存区保存到本地的仓库之中。push -u origin branch\_name 则可以将本地仓库的代码同步到远程仓库了。如果将远程仓库的代码更新到本地仓库，要更新你的本地仓库至最新改动，执行git pull以在你的工作目录中获取（fetch）合并（merge）远端的改动。

HEAD

**分支：**

master

31ac2

98ca5

21ab1

testing

Git中的分支是指向 commit 对象的可变指针，每次提交的时候这个指针都会自动向前移动，这样使得每一次指针都会指向最近一次提交对象的分支上。

新建分支在git中是新建了一个指针，指向当前commit的对象，它保存着一个名为 HEAD 的特别指针,这个指针相当于是分支的别名，指向你正在工作中的本地分支。

每次运行 git branch 命令，仅仅是建立了一个新的分支，并没有改变HEAD指针的指向，所以还需要通过checkout命令来切换到另一个分支当中，此时也就切换了HEAD指针的指向，使得HEAD指向另一个分支当中。

切换分支的神奇之处在于，每次进入现有的分支，进行一波更改之后，可以切到另一个分支来查看以前历史版本的代码或者文件，仿佛灵活穿梭时空一样，历史版本和现有版本同时存在，而且由于git巧妙地设计，使得这种看似复杂的方案占用的空间并不多。

而如果想要将几个分支同步到同一个版本，就需要进行分支的合并，而git并不是简单粗暴的直接把指针移动，而是先把这几个分支合并后的结果做一个新的快照，然后自动创建一个指向它的提交对象到本地仓库之中，

（新系统Visio没安装好，框图是使用的word画的，画风非常丑陋请见谅@-@）