第一课 什么是 Git？

Git 是目前世界上最牛逼、最先进的分布式版本控制系统

Ps:github是git保存的网站

第二课 基础理论

工作区域（Working Directory）就是你平时存放项目代码的地方。

暂存区域（Stage）用于临时存放你的改动，事实上它只是一个文件，保存即将提交的文件列表信息。暂存区在江湖中有个外号叫“索引”（Index）

Git 仓库（Repository）就是安全存放数据的位置，这里边有你提交的所有版本的数据。其中，HEAD 指向最新放入仓库的版本（这第三棵树，确切的说，应该是 Git 仓库中 HEAD 指向的版本）。

工作流程：

在工作目录中添加、修改文件；  
将需要进行版本管理的文件放入暂存区域；  
将暂存区域的文件提交到 Git 仓库。  
因此，Git 管理的文件有三种状态：已修改（modified）、已暂存（staged）和已提交（committed）

git init (初始化 Git)

git add <文件名> (把文件从工作区域放入暂存区域)

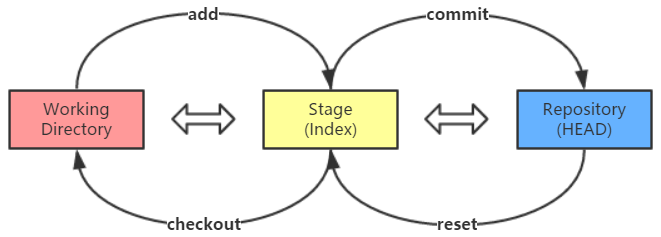
git commit -m "备注说明" （将暂存区域里的东西提交到 Git 仓库中）

第三课 git状态

git status 查看git当前状态

git commit -am "备注说明" Git 会自动帮你将工作目录中所有“已跟踪”的文件先 add 到暂存区域，然后再执行 commit 命令。另外，-am 是 -a -m 的缩写，多个选项可以写在一起

第四课 回到过去



git add 命令用于把工作目录的文件放入暂存区域

git commit 命令用于把暂存区域的文件提交到 Git 仓库

git reset 命令用于把 Git 仓库的文件还原到暂存区域

git checkout 命令用于把暂存区域的文件还原到工作目录

git log 命令查看历史提交

git reset HEAD~ 向前恢复一个版本

git reset --soft HEAD~ 只移动 HEAD 的指向，但并不会将快照回滚到暂存区域，即只移动指针

git reset --hard HEAD~ 不仅移动 HEAD 的指向，将快照回滚动到暂存区域，它还将暂存区域的文件还原到工作目录。

**最后总结一下：reset 回滚快照三部曲**

**1. 移动 HEAD 的指向（--soft）**

**2. 将快照回滚到暂存区域（[--mixed]，默认）**

**3. 将暂存区域还原到工作目录（--hard）**

git reset 00c2929 指定回滚

git reflog 操作记录

第五课 版本比对

git diff 比较暂存区域与工作目录的文件内容

--- 表示该文件是旧文件

+++ 表示该文件是新文件

第六课 修改最后一次提交、删除文件和重命名文件

git commit –amend 修改最后一次提交

git commit --amend -m "新的提交说明" 一次性必修，如果中文说明异常，也可以用这个方法

git rm <文件名> 删除文件，如果要彻底删除，git reset --soft HEAD~，软回滚快照

注意：rm 命令删除的只是工作目录和暂存区域的文件（即取消跟踪，在下次提交时不纳入版本管理）

git mv <旧文件名> <新文件名> 重命名文件

注：Windows 使用 ren 命令修改文件名，Linux 是使用 mv 命令

让git忽略文件的方法：

创建一个名为 .gitignore 的文件，

第七课 创建和切换分支

git branch <分支名称> 创建分支

git log –decorate 以精简版模式查看日志情况

git log --decorate –oneline 一行显示一个快照信息

git checkout <分支名> 切换分支

git log --oneline --decorate --graph --all

-graph 选项表示让 Git 绘制分支图，--all 表示显示所有分支

第八课 合并和删除分支

git merge <分支名> 合并分支

git branch –d <分支名> 删除分支

第九课 S1E9匿名分支和checkout命令

git checkout HEAD~ 匿名分支,可以做实验用，切换到其他分支会丢弃所有分支文件

checkout与reset的区别

checkout 命令有两种功能：

1：从历史快照（或者暂存区域）中拷贝文件到工作目录

2：切换分支

checkout更安全

第十课 github