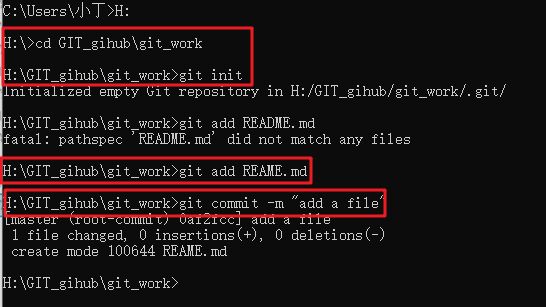
Git

**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。  
**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）。  
**Git 的工作流程一般是酱紫：**  
1. 在工作目录中添加、修改文件；  
2. 将需要进行版本管理的文件放入暂存区域；  
3. 暂存区域的文件提交到 Git 仓库。

Git 管理的文件有三种状态：已修改（modified）、已暂存（staged）和已提交（committed）



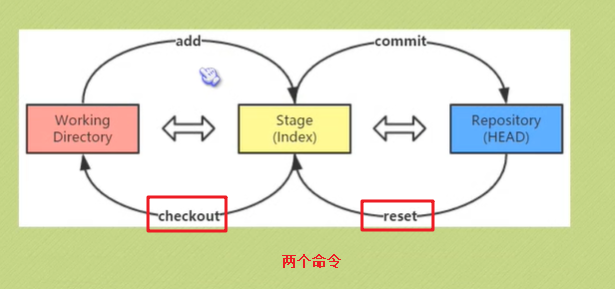
查看当前工作：

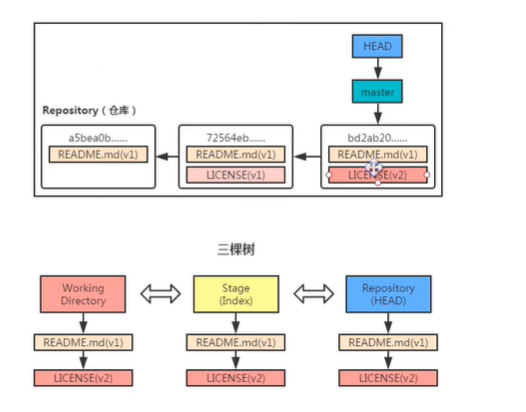
git status

提交记录

git log

回到过去版本



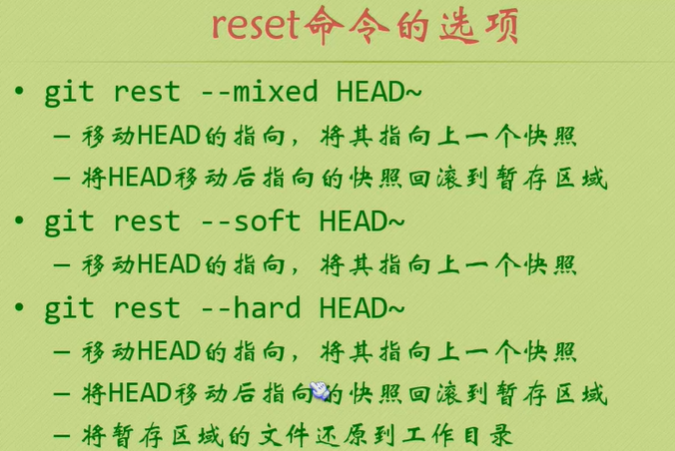


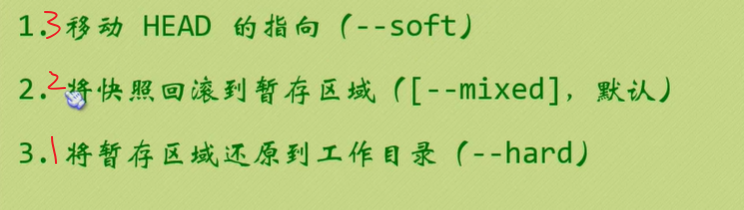
Head~ 快照 回到上一个版本 head~10 回到第十个版本

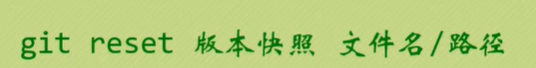
Git rest – head~

* 指向上一个快照
* 移动后的快照回滚到暂存区域

快照是数据存储的某一时刻的状态记录；备份则是数据存储的某一个时刻的副本







**比较暂存区域和工作目录**

Git diff

比较改动后不同代码

移动命令 d、u半页移动 j、k按行移动等等·

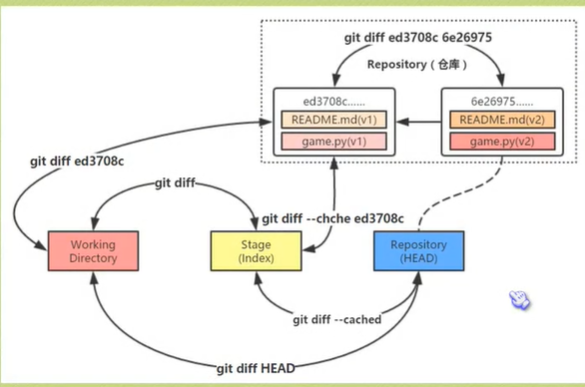
H帮助文档会有提示

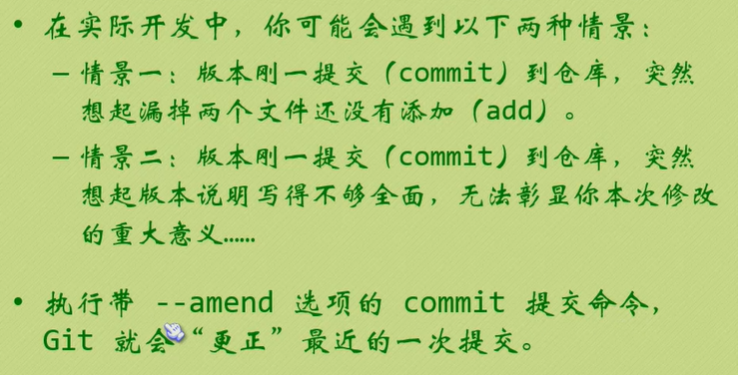
Q退出

**比较历史快照**

Git diff 版本号5位 版本号 两个版本号前五位即可

**比较当前工作目录和git仓库区别**





Git commit –amend

文件删除回复

Git checkout – 文件名

文件不小心提交后：

Git rm 文件名删除从工作区删除

回到过去

Git reset – soft HEAD~

就不会留下痕迹

Git rm -f 强制删除暂存和工作区域对应文件

Git rm –cached 文件名 删除暂存区域文件

**重命名**

Git mv game.py(旧文件) wordgame.py(新文件)

# Git分支

**创建分支**

Git branch feature(分支名)

Git log –decorate --oneline 一行显示一个快照 --graph --all 图形化显示从下到上

（git log）

切换分支:Git checkout 分支名

**合并删除分支**

Git merge 分支