**创建工作区+初始化仓库：**

（1）新建文件夹，进入该目录。 该目录即为工作区

（2）初始化一个Git仓库，使用git init命令。

（3）添加文件到Git仓库，分两步：

使用命令git add <file>添加至暂存区。（一次只能添加一个文件，可多次添加）

使用命令git commit -m <message>提交修改

git add -u：将已追踪文件的修改、删除，添加到暂存区

git add .：保存文件的新建和修改，（不包括删除）添加到暂存区。

git add -A：将文件的修改，文件的删除，文件的新建，添加到暂存区

**如何变换版本号：**

HEAD指针指向的版本就是当前版本

使用命令git reset --hard commit\_id。

1. 回退到上一个版本，git reset --hard head^。 退两个版本号则为head^^。

以此类推，当回退版本号过多时，可用head~100这种方式表示

（2） 用git log可以查看提交历史，以便确定要回退到哪个版本。

（3） 回退到4G时代想重回5G时代，用git reflog查看命令历史，以便确定要回到未来的哪个版本。

**Git提交原理**

工作区的修改ADD到缓存区



将缓存区的修改COMMIT到版本库



每次做的修改，如果不用git add到暂存区，那就不会加入到commit中

这过程实质上是对“修改”进行的ADD，COMMIT

**删除：**

彻底删除版本库及本地文件：

（1）RM test.txt (2)git status 查看文件状态，哪些文件被删除了 （3）git RM test.txt

（4）git commit -m “remove”

git checkout -- test.txt

**修改的撤销：**

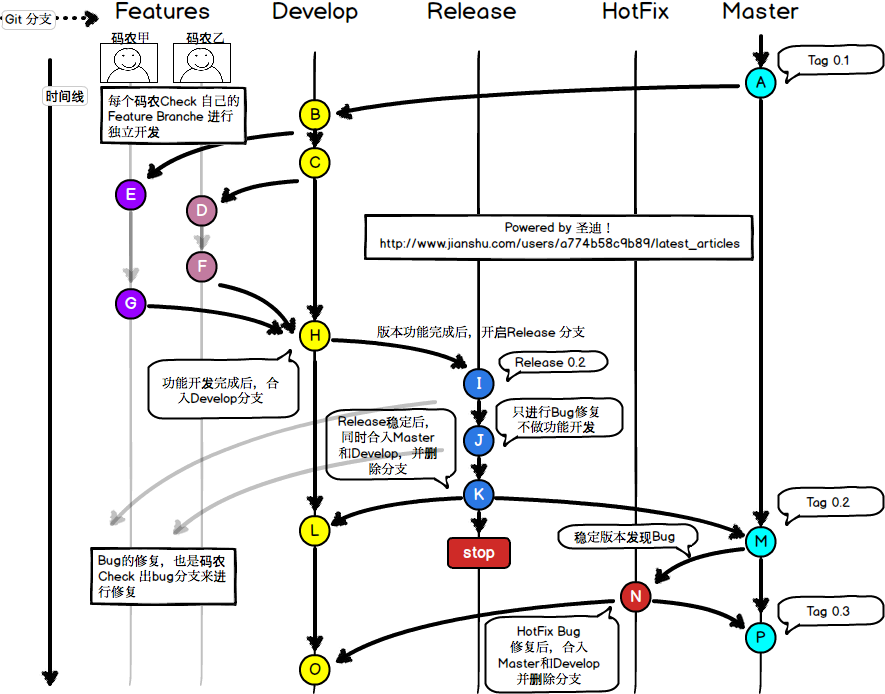
场景1：当你改乱了工作区某个文件的内容，想直接丢弃工作区的修改时，用命令git checkout -- file。

场景2：当你不但改乱了工作区某个文件的内容，还添加到了暂存区时，想丢弃修改，分两步，第一步用命令git reset HEAD <file>，就回到了场景1，第二步按场景1操作。

场景3：已经提交了不合适的修改到版本库时，想要撤销本次提交，参考[版本回退](https://www.liaoxuefeng.com/wiki/896043488029600/897013573512192)一节，不过前提是没有推送到远程库。

场景4：将错误的内容推送到了远程库，删库跑路。

**分支：**



以dev为例，列举

1. 查看分支: git branch

当前所处分支会以 \* 标识

1. 创建dev分支：git branch dev

切换到dev分支: git checkout dev

创建并切换到dev分支： git checkout -b dev

1. 合并dev分支到master分支： git merge dev

需提前切换master分支为当前分支

1. 删除dev分支：git branch -d dev
2. 合并分支后再删除分支但不删除其存在记录：dev merge –-no -ff -m dev
3. 冲突：并无详细记载

倘若处于冲突状态（master/merging）并向退出该状态回到冲突前：

git reset --hard HEAD

1. 手头活没干完，要停下临时去干别的活（常见于修bug）
   1. 先将没干完的活保存：git stash
   2. 该干嘛干嘛去
   3. Git stash list查看莲ma区都有哪些
   4. 恢复
      1. 恢复后删除莲ma区内的相应数据：git stash pop
      2. 恢复后不删除莲ma区内的数据：git stash apply

之后手动删除莲ma区内的数据：git stash drop

1. 针对一个没有提交过的分支，使用强制删除：git branch -D dev

**多人协作篇：**

（1）查看远程库信息，使用git remote -v；本地新建的分支如果不推送到远程，对其他人就是不可见的；

（2）从本地推送分支，使用git push origin branch-name，如果推送失败，先用git pull抓取远程的新提交；

（3）在本地创建和远程分支对应的分支，使用git checkout -b branch-name origin/branch-name，本地和远程分支的名称最好一致；

（4）建立本地分支和远程分支的关联，使用git branch --set-upstream branch-name origin/branch-name；

（5）从远程抓取分支，使用git pull，如果有冲突，要先处理冲突

**标签：**

1. 新建标签： git tag tagname [ commit id ]
   1. 一般默认将标签加在当前HEAD指向位置
   2. 加上commit id则将标签加在该位置
   3. 可以创建带有说明的标签，用-a指定标签名，-m指定说明文字：

git tag -a v0.1 -m "version 0.1 released" 1094adb

* 1. Show tag可查看所有标签
  2. git show tagname可以查看标签信息以及说明文字

注意：标签总是和某个commit挂钩。如果这个commit既出现在master分支，又出现在dev分支，那么在这两个分支上都可以看到这个标签。

1. 删除本地标签：git tag -d tagname
2. 删除已推送标签：
   1. Git tag -d tagname
   2. Git push origin :refs/tags/tagname
3. 推送标签：
   1. 推送一个标签：git push origin tagname
   2. 推送所有标签：git push origin -tags

**自定义Git部分自行调理：https://www.liaoxuefeng.com/wiki/896043488029600/900785521032192**