基础

2020年6月17日　　23:36

1. git下载和安装

2. git配置

   添加全局配置

$ git config --global user.name "用户名"
$ git config --global user.email "邮箱名"

   ```
   xl@DESKTOP-M6LF3LK MINGW64 ~/Desktop
   $ git config --global user.name "xivLi"

   xl@DESKTOP-M6LF3LK MINGW64 ~/Desktop
   $ git config --global user.name
   xivLi
   ```

   ```
   xl@DESKTOP-M6LF3LK MINGW64 ~/Desktop
   $ git config --global user.email "13373276585@163.com"

   xl@DESKTOP-M6LF3LK MINGW64 ~/Desktop
   $ git config --global user.email
   13373276585@163.com
   ```

3. git仓库初始化

   切换到仓库文件目录，执行git init

$ git init

   ```
   xl@DESKTOP-M6LF3LK MINGW64 ~/Desktop/gitdir
   $ git init
   Initialized empty Git repository in C:/Users/xl/Desktop/gitdir/.git/
   ```

4. 常用指令

   a. 将文件添加到暂存区：git add

$ git add 文件名
$ git add 文件名1 文件名2 文件名3 ...
$ git add .

   b. 将文件中暂存区移除：git rm或git restore

$ git rm --cached 文件名	
$ git restore --staged 文件名	新

   c. 将文件从暂存区提交到本地仓库：git commit

$ git commit -m "说明性文字"	

   d. 比较文件差异：git diff

$ git diff 文件名	修改了文件后通过该指令，将现有文件和暂存区的文件进行比较，不指定文件名，将比较所有文件
$ git diff HEAD 文件名	将现有文件和本地仓库的文件进行比较
$ git diff HEAD^ 文件名	将现有文件和某一历史版本的本地仓库的文件进行比较

5. 分支管理

   a. 列出当前仓库所有分支

$ git branch -v	

b. 创建分支

$ git branch 分支名	

c. 切换分支

$ git checkout 分支名	

d. 合并分支

$ git checkout 主分支名	
$ git merge 副分支名	

6. 版本回退

a. 查看版本

$ git log	
$ git log --pretty=oneline	推荐
$ git log --oneline	

b. 回退

$ git reset --hard 版本编号	可以前进后退，与git reflog指令配合使用
$ git reset --hard HEAD^^^	只能回退到某一历史版本，这里回退三步，与git log指令配合使用
$ git reset --hard HEAD~3	只能回退，这里回退3步，与git log指令配合使用

c. 撤销回退

$ git reflog	通过查看历史操作得到某一个版本编号
$ git reset --hard 版本编号	

7. 远端仓库

a. 克隆

$ git clone https://github.com/xxx/xxx.git	将远端仓库克隆到当前目录

b. 提交到远端仓库

$ git add 文件名	
$ git commit -m "说明性文字"	
$ git push 远端仓库地址或代号 分支名	如：git push origin master

c. 获取远端仓库最新版本

$ git pull origin master = git fetch origin master + git merge origin/master	pull=fetch+merge origin是远端仓库别名 master是远端仓库分支

d. 冲突解决

当两人均对远端仓库的同一文件做了不同修改，且其中一人已将修改推送到远端仓库时，另一人想要推送，则需要先解决冲突：

第一步：

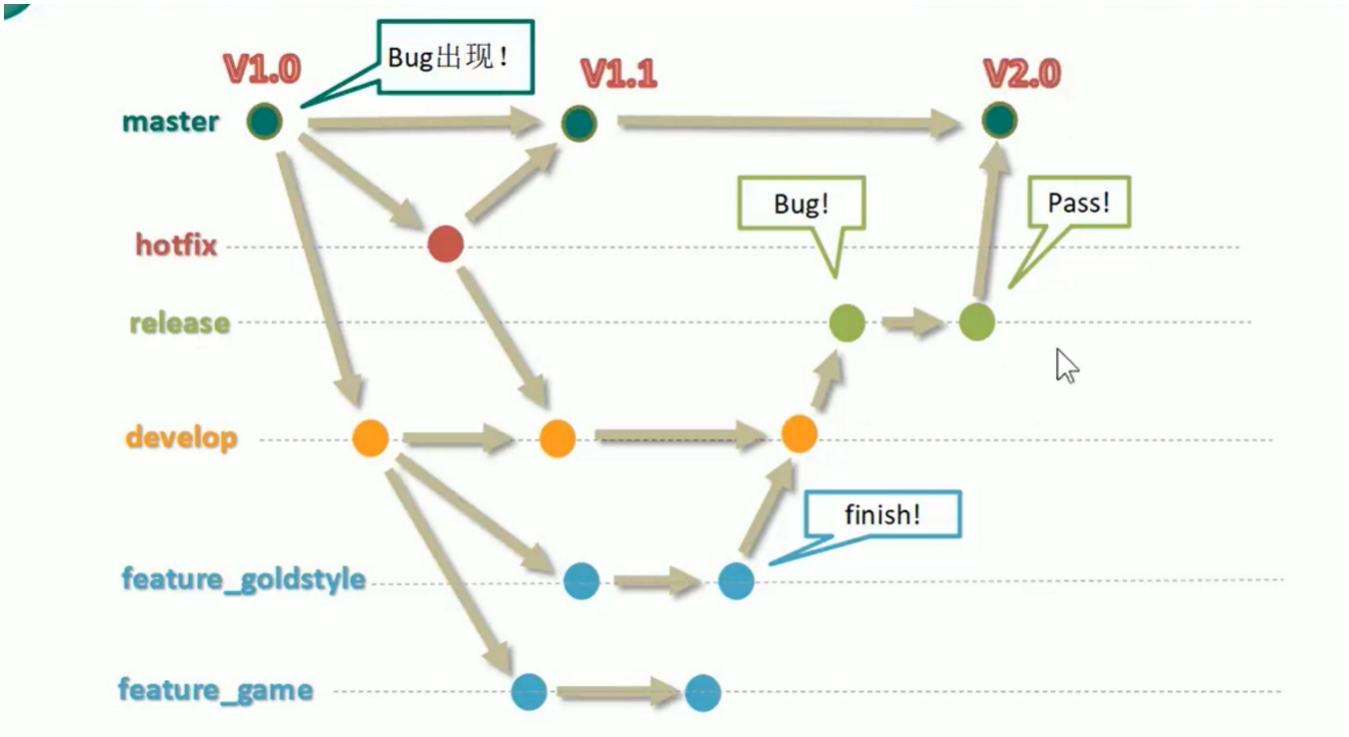
$ git pull origin master	获取远端仓库的master分支最新版本，若有冲突则手动解决冲突

解决办法：进入冲突文件，结合队友的修改将冲突文件修改成最终提交的样子

执行	git add 冲突文件名
再执行	git commit -m "说明文字"

第二步：

$ git push origin master	将本地仓库的master分支推送到远端仓库


V1.0
Bug出现！
V1.1
V2.0
master
hotfix
Bug!
Pass!
release
develop
finish!
feature_goldstyle
feature_game