

Git版本控制和Xcode结合使用

Shirley
2015/08/25

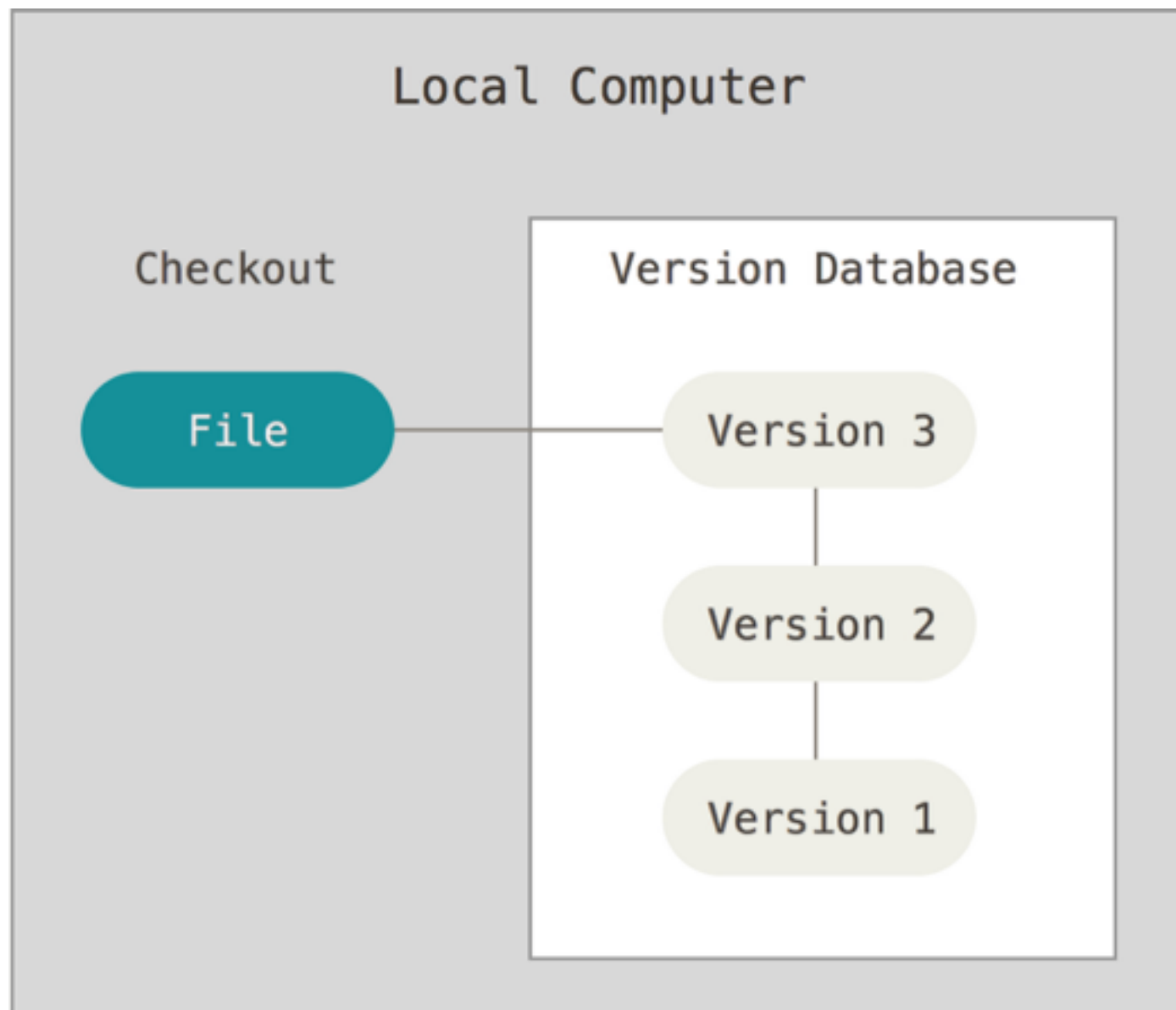
为什么要使用版本控制

什么是版本控制：版本控制是一种记录一个或若干文件内容变化，以便将来查阅特定版本修订情况的系统

版本控制系统本质：是检测文件的改变并将其保存留作以后参考使用

早期本地版本控制系统

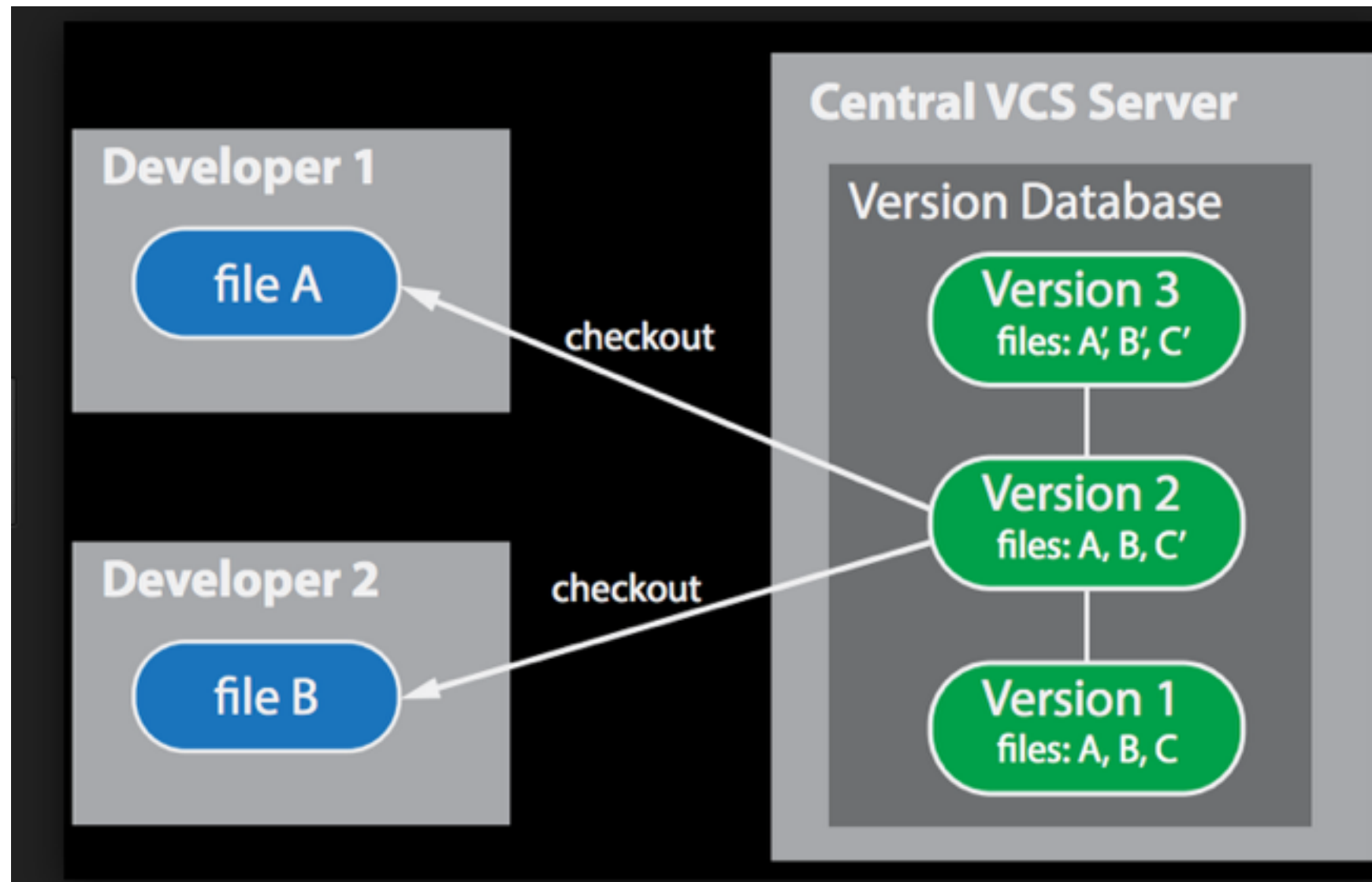
单个人；本地系统



中心版本控制系统

特点：多个人合作

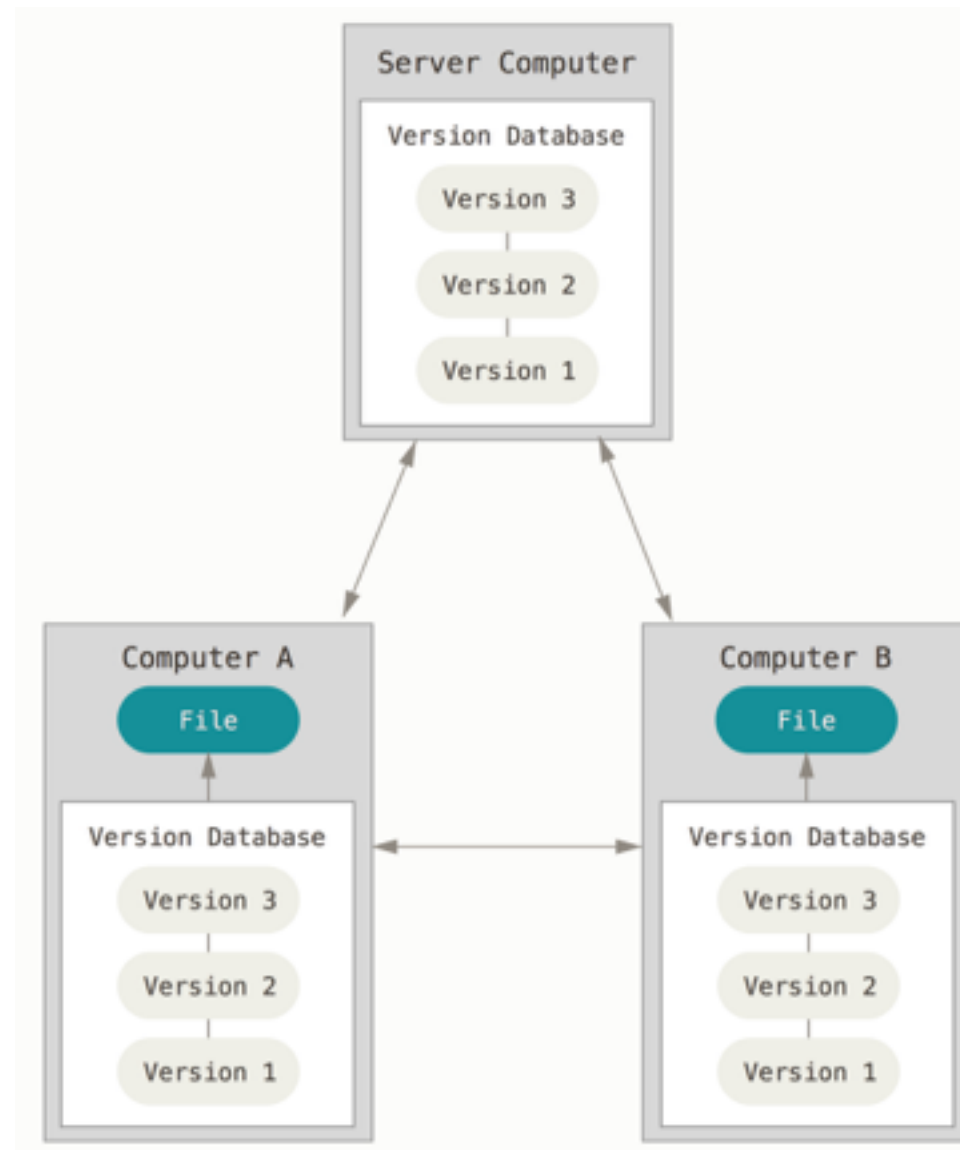
缺点：Server Down；硬盘损坏



分布式版本控制系统

特点:

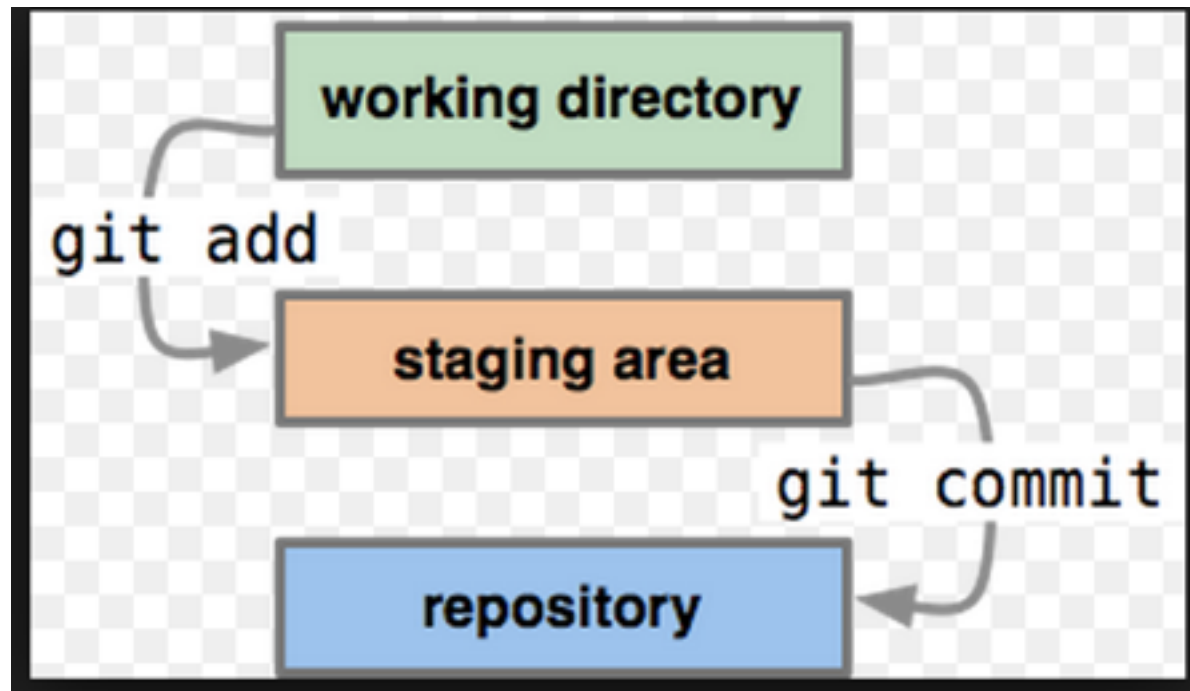
1. 每个客户端并不只是获取最新的代码，包含和服务端一样的版本
2. 服务器故障时，从任意一个client恢复



什么是git

- Git是一个开源的分布式版本控制系统，用以有效、高速的处理从很小到非常大的项目版本管理。
- Git 是 Linus Torvalds 为了帮助管理 Linux 内核开发而开发的一个开放源码的版本控制软件。

git工作原理(本地)



git工作原理(本地)

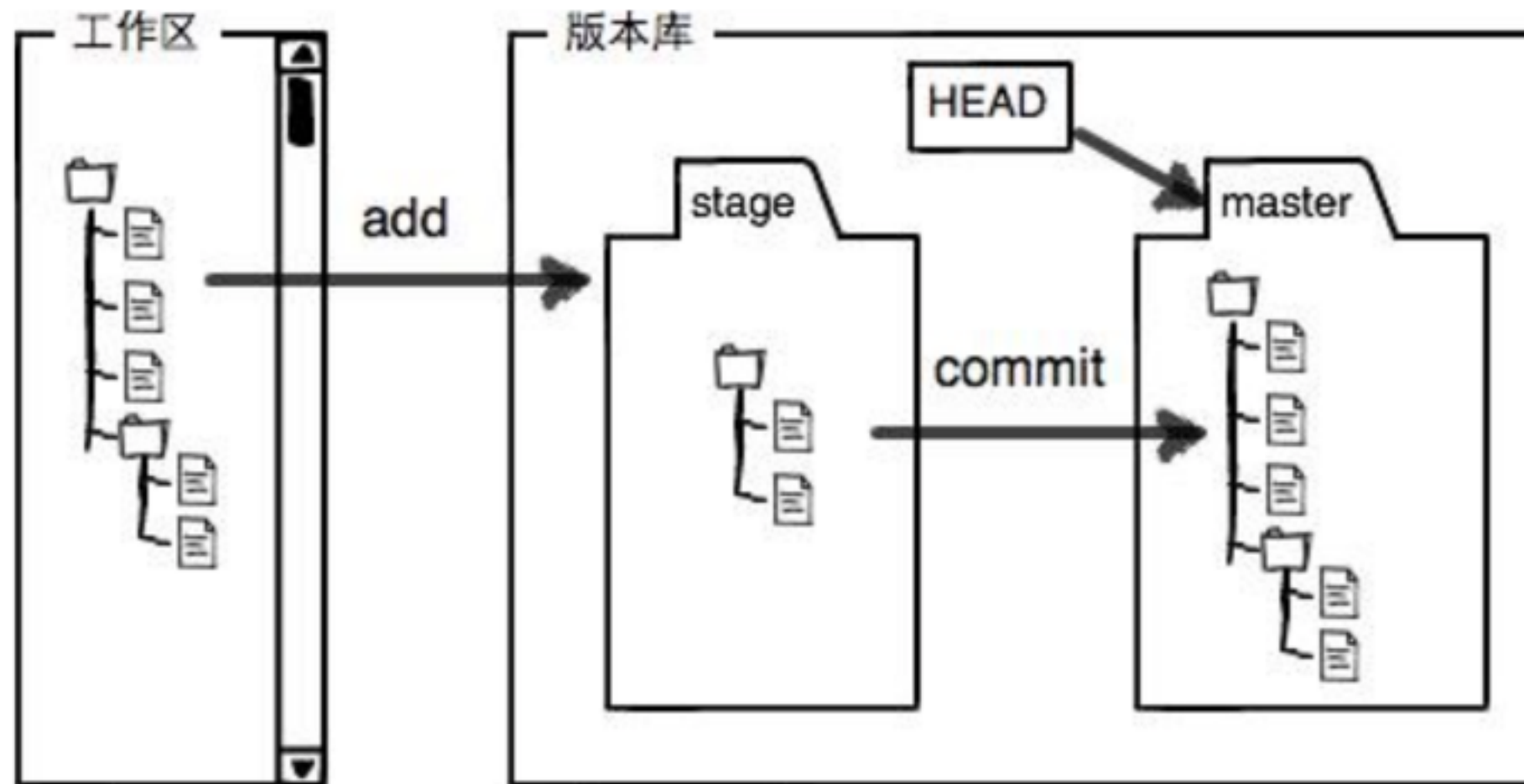
工作区(**Working Directory**): 仓库文件夹里除.git目录以外的内容

版本库(**Repository**): .git目录, 用于存储记录版本信息

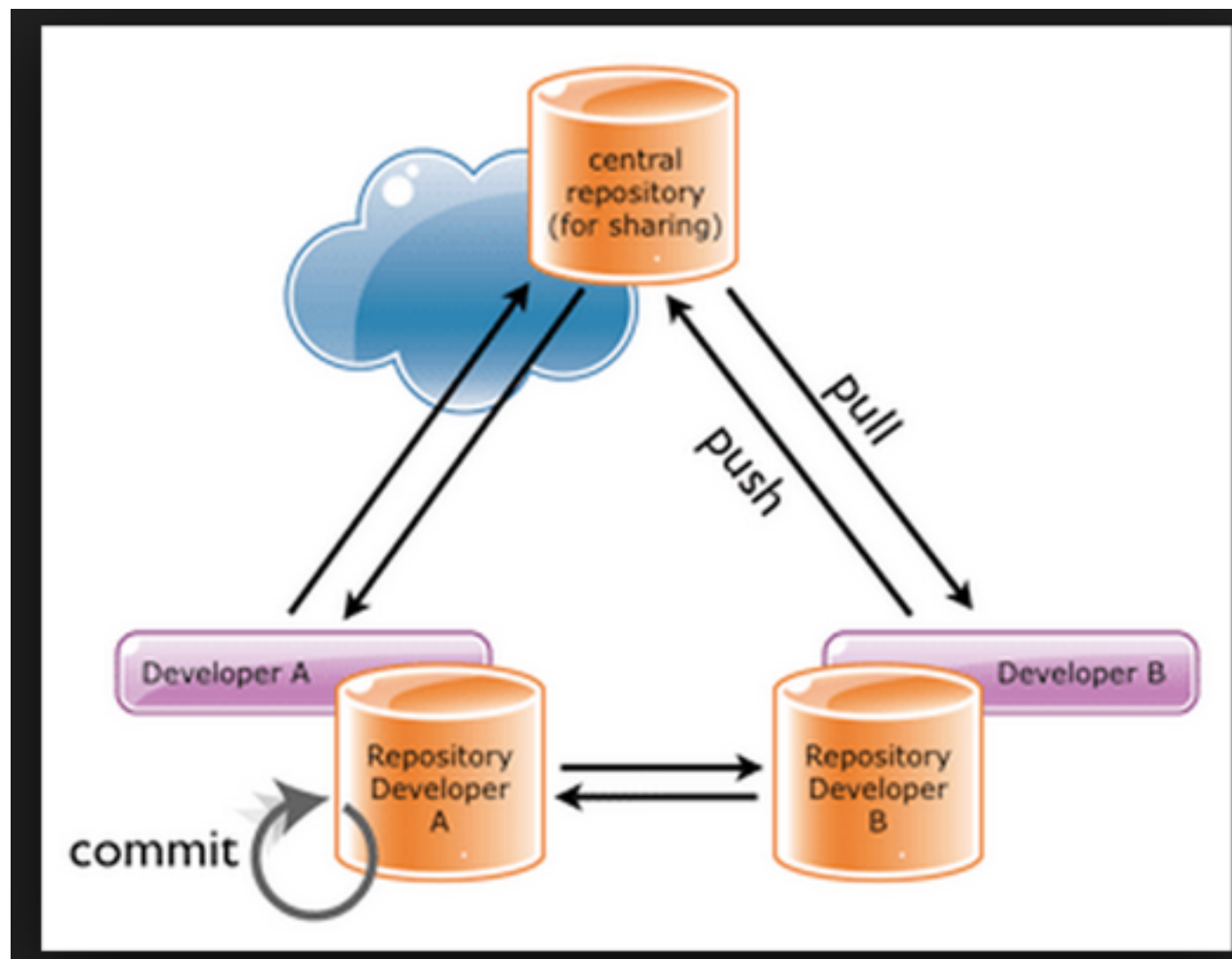
暂缓区(**stage**): 执行commit动作, 缓冲区不会存在原来add的文件

分支(**master**): git自动创建的第一个分支

HEAD指针: 用于指向当前分支

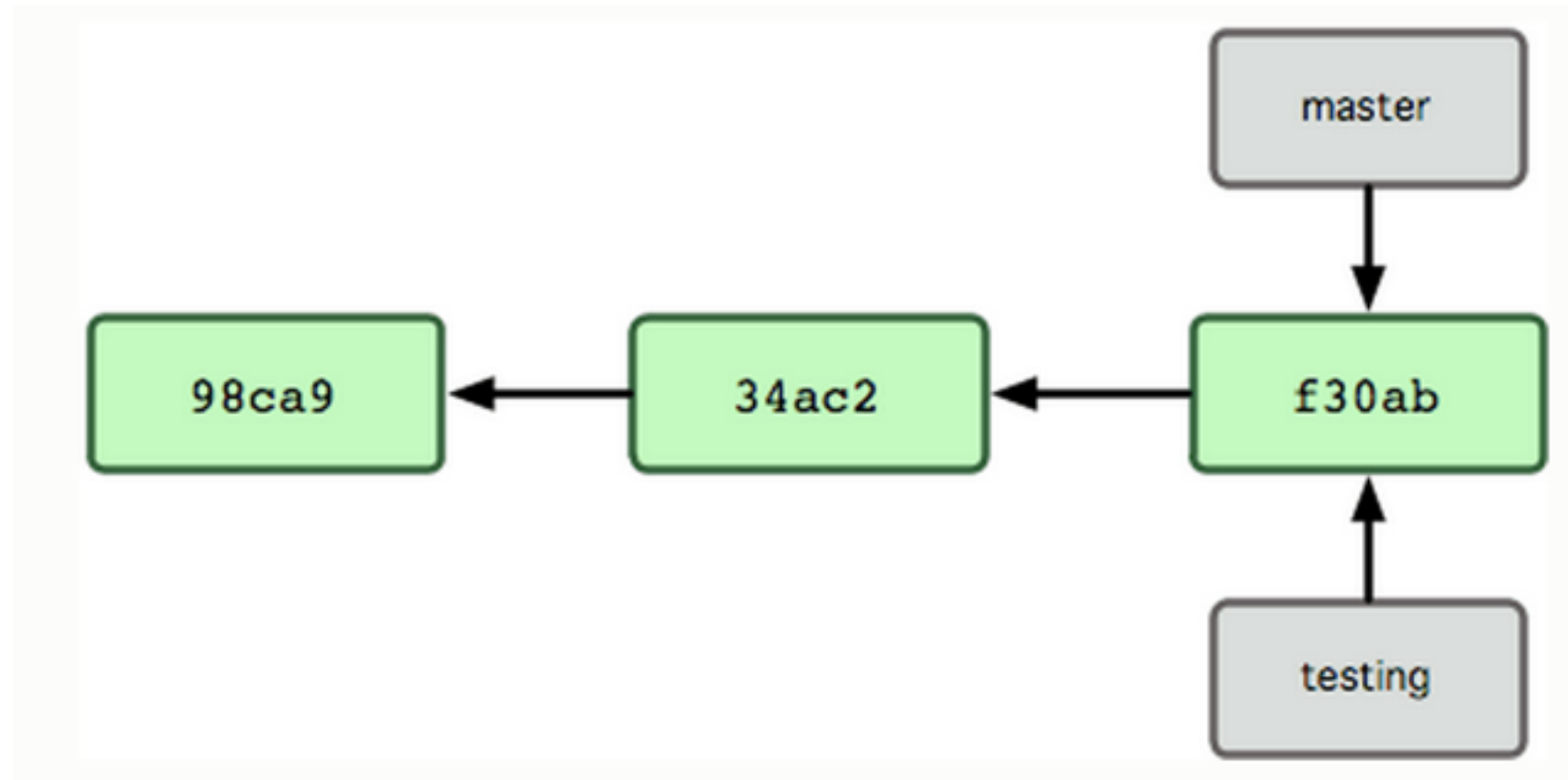


git工作原理(本地和远程)



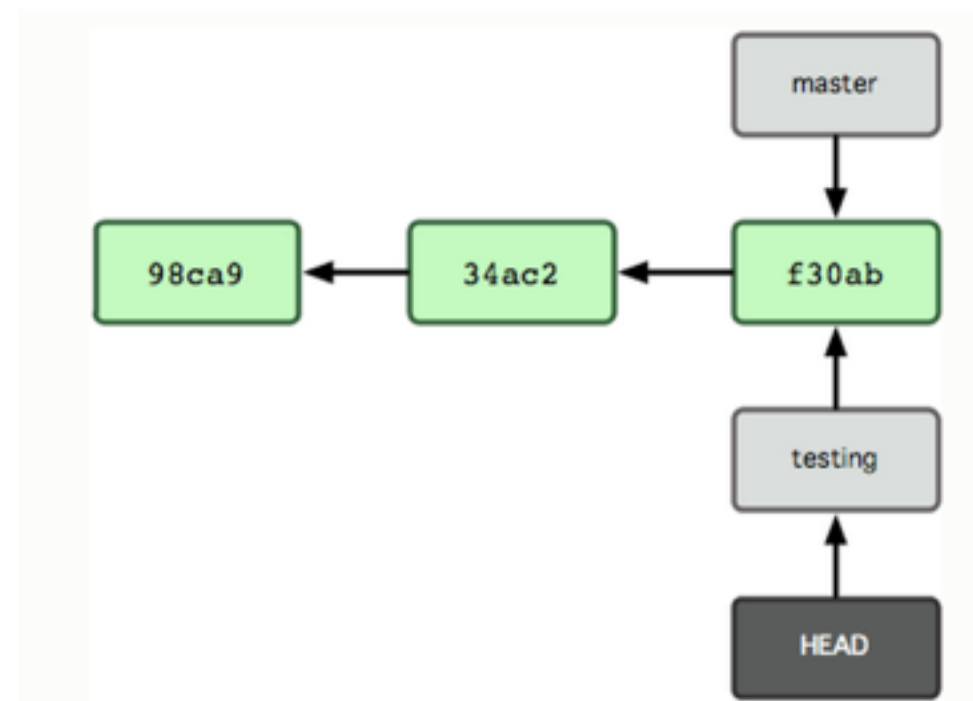
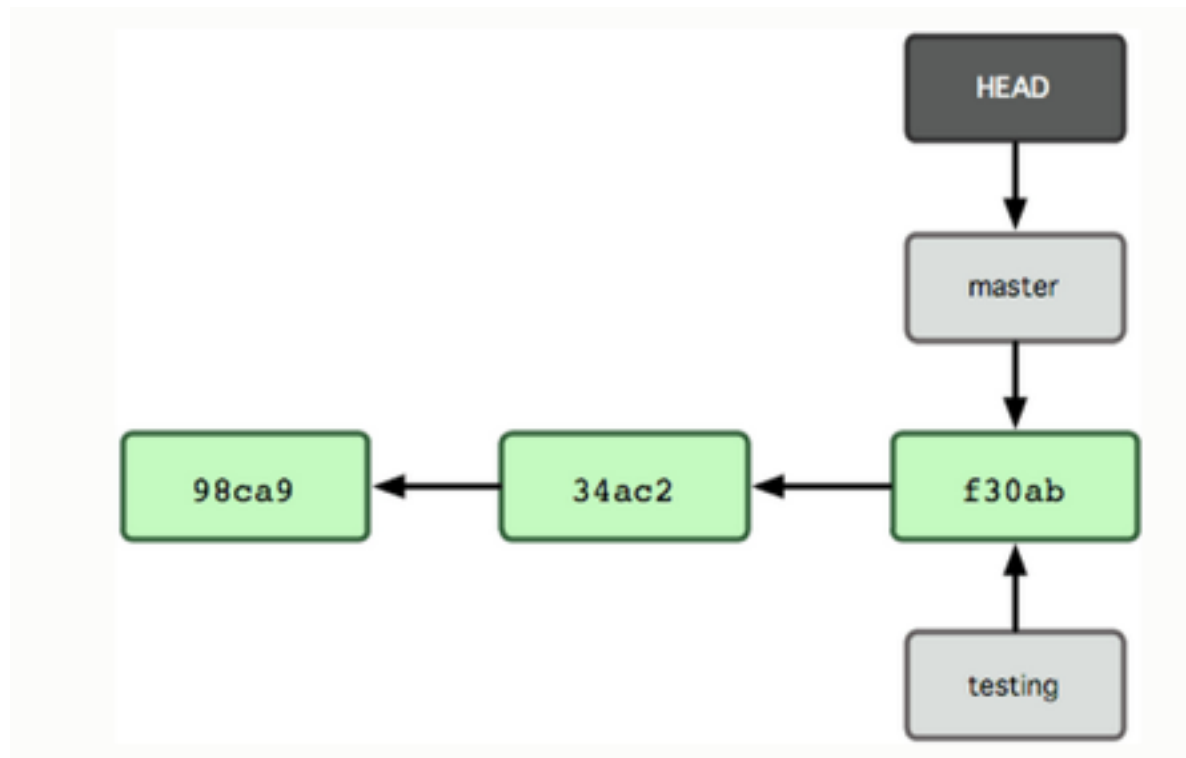
git分支(了解)

1. 几乎每一种版本控制系统都支持分支。
2. 使用分支从开发主分支(master)上分离，不影响主线的代码。



git分支(了解)

如何知道你当前在哪个分支上工作的呢?
—> HEAD指针



git常用命令(了解)

查看命令:

1. `git --help`: 查看git的所有命令
2. `git clone -help`: 查看git clone命令的细节
3. `git config -l`: 查看当前所有配置

创建本地代码库 & 配置个人信息:

1. `git init`: 初始化本地仓库
2. `git config user.name xxxx`: 配置用户名字(xxxx为配置的用户名)
3. `git config user.email xxxx@gmail.com`: 配置邮箱

一次性配置完成用户名和邮箱的配置

1. `git config --global user.name xxxx`: 一次性配置用户名字(xxxx为配置的用户名)
2. `git config --global user.email xxxx@gmail.com`: 一次性配置邮箱

git常用命令(了解)

git代码提交相关命令：

1. git status: 查看代码库状态
2. git add xxxx: 将文件xxxx提交到缓存区
3. git commit -m “提交注释描述”： 将修改的代码提交到本地仓库Repository
4. git add .: 将当前文件夹下的所有新建或修改的文件一次性添加到缓存区
5. git diff: 查看文件最新修改的文件
6. git diff xxxx: 查看xxxx文件修改的地方
7. git checkout xxxx: 撤销对xxxx文件的修改

git日志log相关命令：

1. git log: 查看所有版本库日志
2. git log xxxx: 查看xxxx文件的版本库日志

git版本切换相关命令：

1. git reflog: 查看当前分支引用纪录
2. git reset xxxx: xxxx文件从缓存区恢复到工作区(修改保留)
3. git reset --hard: 恢复最近一次提交过的状态，即放弃上次提交后的所有本次修改
4. git reset --hard 8146023: 返回到8146023那个版本(使用git reflog命令查看)

git常用命令(了解)

git分支branch相关命令：

1. git branch: 查看当前所处的分支
2. git branch xxxx: 创建xxxx分支
3. git checkout xxxx: 切换到xxxx分支
4. git branch -d xxxx: 删除xxxx分支
5. git merge xxxx: 将xxxx分支的代码合并到当前分支

git标签tag相关命令：

1. git tag: 查看版本
2. git tag xxxx: 在本地创建xxxx的版本
3. git tag -d xxxx: 删除xxxx的版本

git远程仓库

远程仓库搭建：

1. github (优秀第三方库): <https://www.github.com>
2. oschina (国内访问速度快些): <http://git.oschina.net>

忽略不需要加入版本控制的文件或者文件夹：

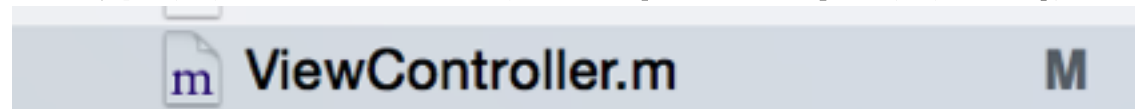
1. 链接: <https://github.com/search?utf8=✓&q=gitignore>
2. 将.gitignore文件拷贝到和.git文件夹同目录下
3. `git add .gitignore`: 添加到缓存去
4. `git commit -m "添加ignore文件"`: 添加到本地仓库中

如何在Xcode中使用git(mac本地) (掌握)

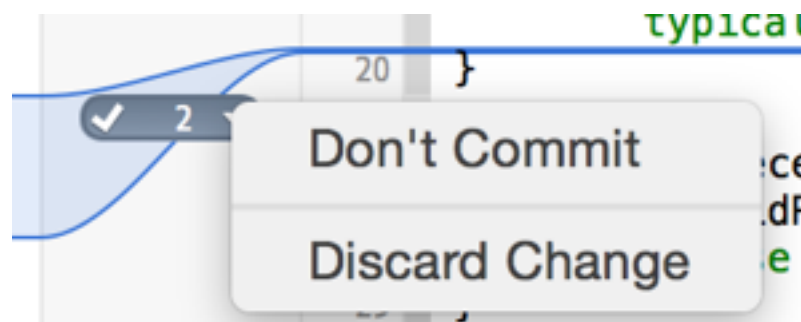
目标： 使用xcode创建一个git管理的工程

步骤：

- 1.创建xcode工程，勾选“Create git repository on My Mac”选项
——> Xcode 默认会在新项目创建时保存一个项目初始状态的版本
2. 添加/修改代码
3. Source Control > Commit提交修改版本
4. 左侧是文件的当前版本(没有提交的), 右侧是文件的最近提交的版本

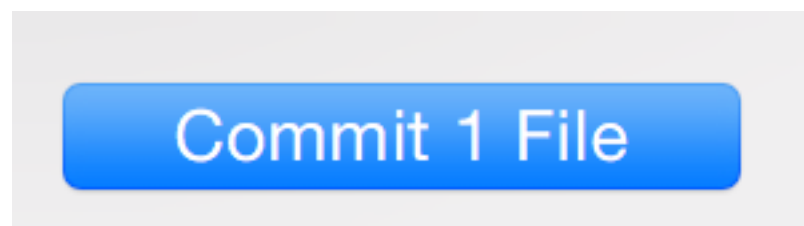


Xcode中使用git细节(mac本地)



Don't Commit: 对勾符号会换成禁止符号，对应的改动就不会提交到 repository 了

Discard Change: 取消修改的部分



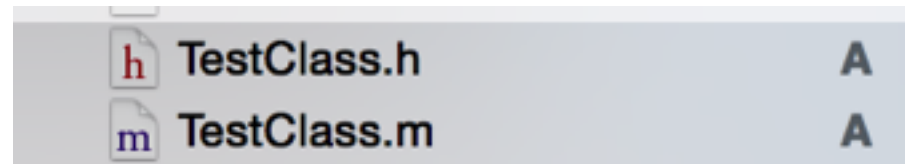
添加commit (提交的描述信息)

提交修改的变化

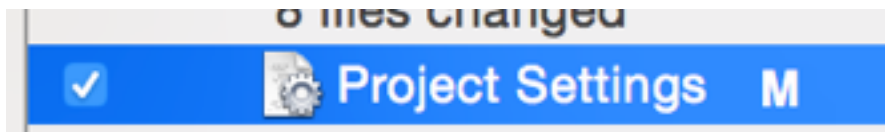
Source Control > History: 显示提交的历史纪录

Xcode中使用git细节(mac本地)

新创建类文件



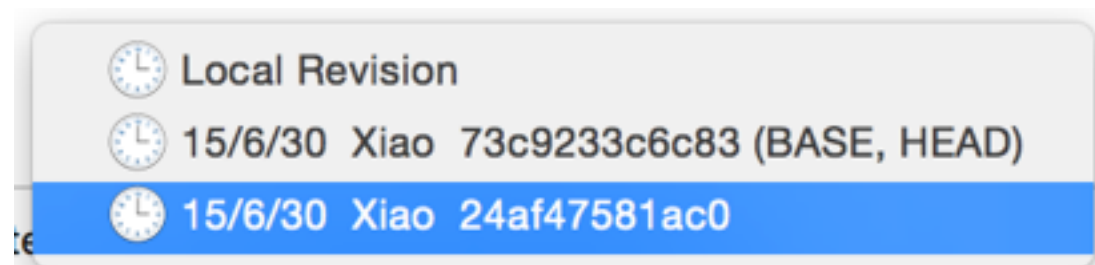
项目的配置文件：在添加新类时由 Xcode 自动修改的



版本对比：

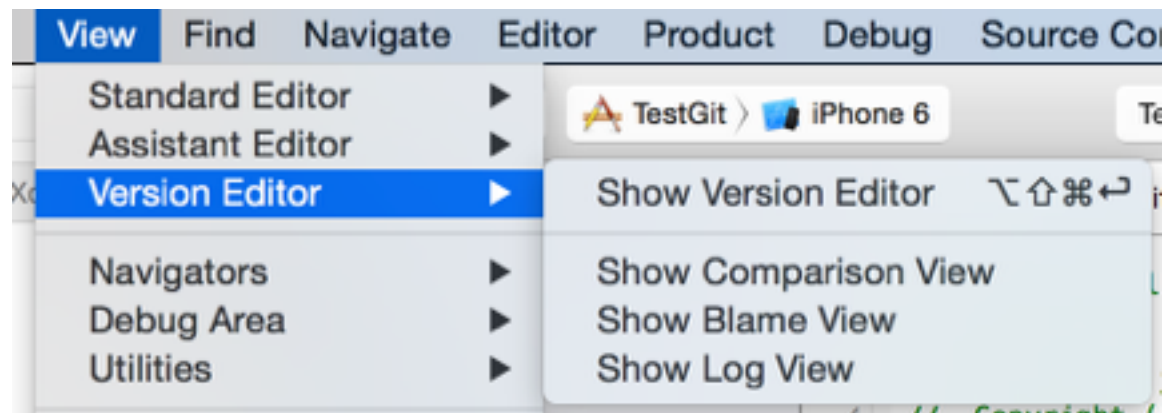


View > Version Editor > Show Version Editor

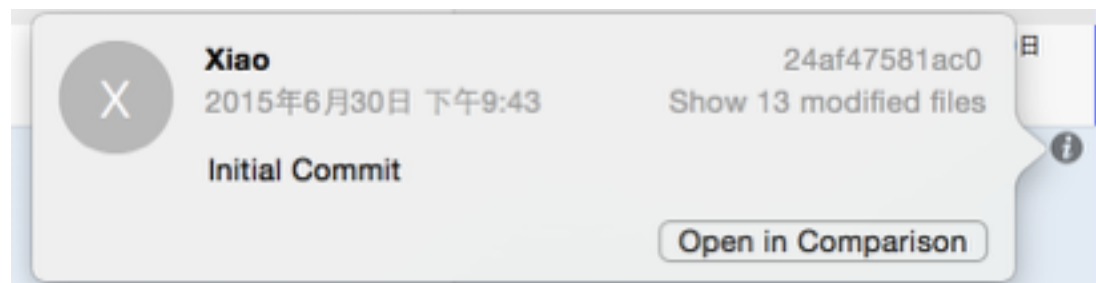


- 查看谁提交：

View > Version Editor > Show Blame View



- 查看某次提交前后代码对比



- 显示log视图：

View > Version Editor > Show Log View

xcode上的git分支

情景： 开发新特性，不想破坏目前稳定主分支的程序

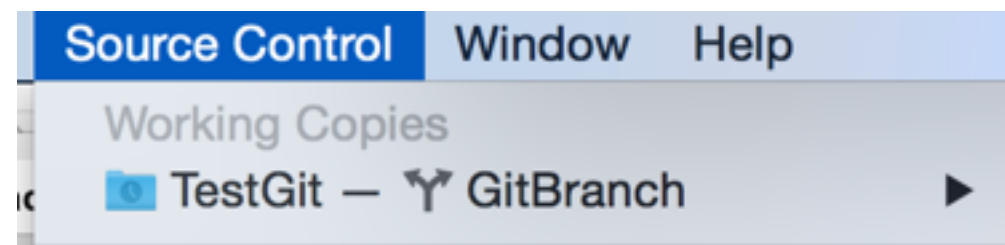
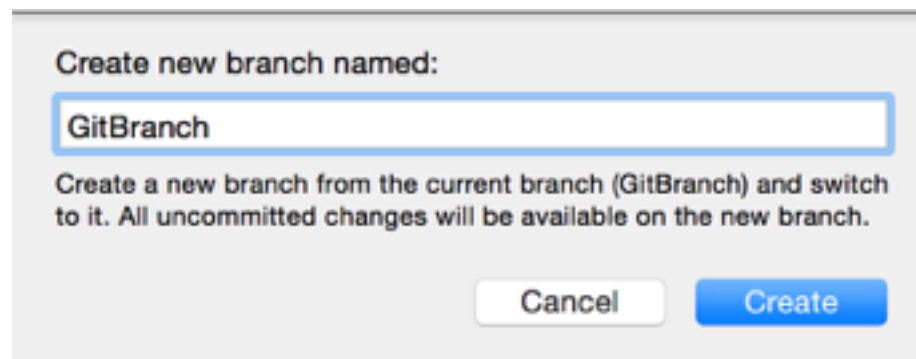
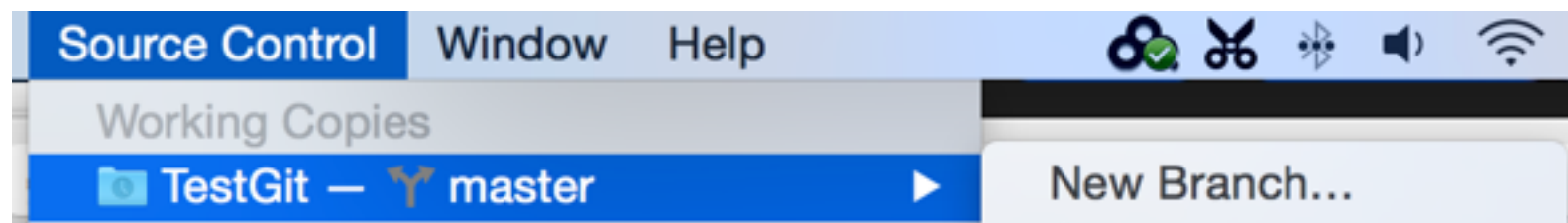
git 总会默认创建一个分支，名为 **master**。

备注：

1. 提交到 App Store的最终产品，一定是 master 分支的版本。
2. 任何处于其它分支的代码，都必须先合并到 master 分支，之后才能正式发布

在xcode上创建分支步骤：

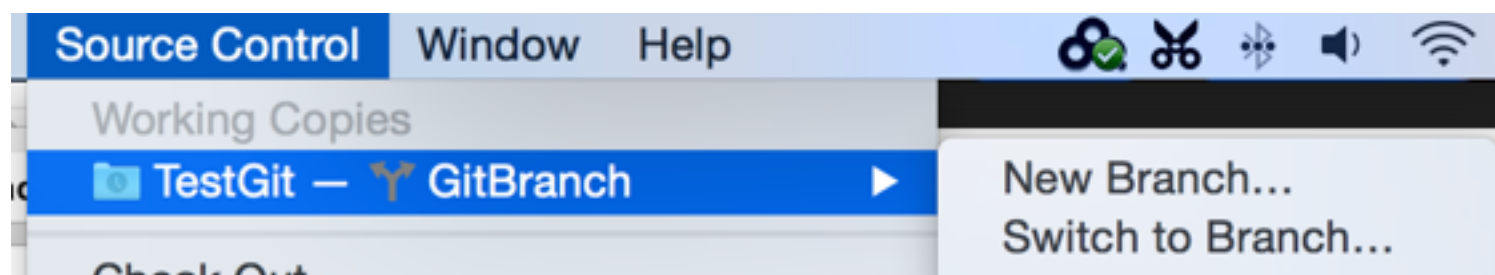
1.Source Control > master > New Branch



提交代码git分支

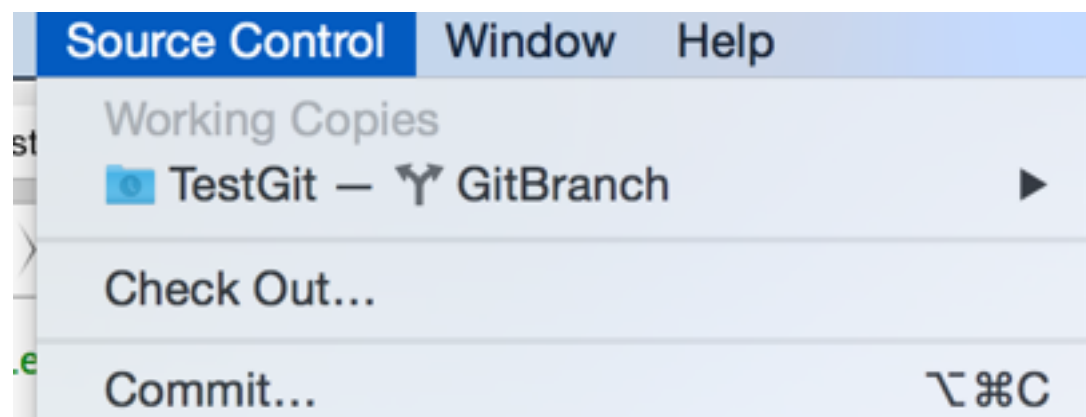
步骤：

1. 切换branch



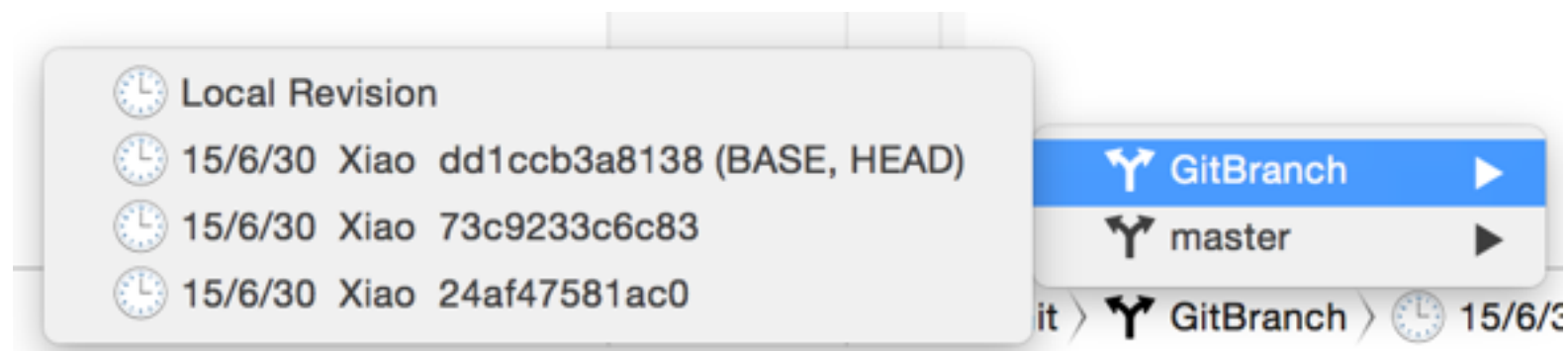
2. 修改代码

3. 提交代码Commit



4. 提交代码Commit

View > Version Editor > Show Version Editor

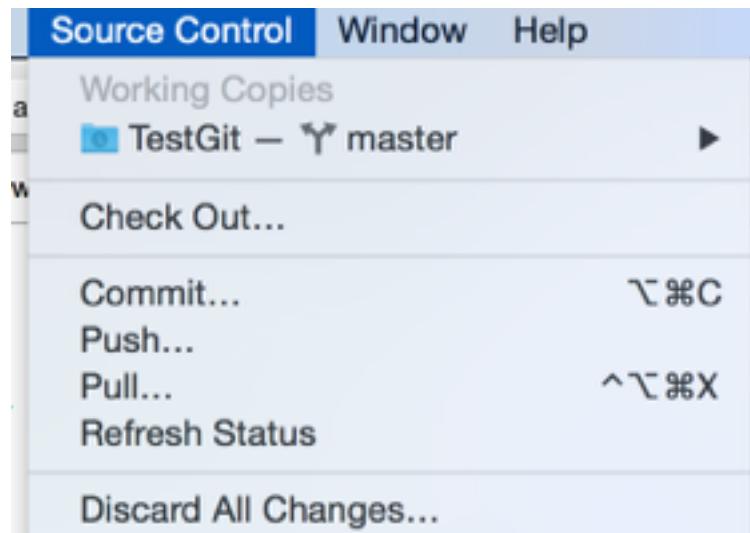


合并两个分支变化到主分支

合并步骤：

1. 首先在合并之前，该分支上做的修改必须先提交到本地仓库
2. 要确定你现在处在主分支上
3. 合并两个不同的分支到一个分支上，你有两种选择
 - 1) 从分支上合并**Merge From Branch**: 你可以选择在分支上做过的任何修改来合并到当前工作的分支上。
 - 2) 合并到分支上**Merge Into Branch**: 你可以选择在当前工作的分支上做过的任何修改合并到分支上。
 - 3) **注意**: 当你当前活跃的分支是主分支的话，第二个选择是不可用的。

放弃提交的修改



备注：

- 下班前提交push当天的代码
- 上班前pull最新代码

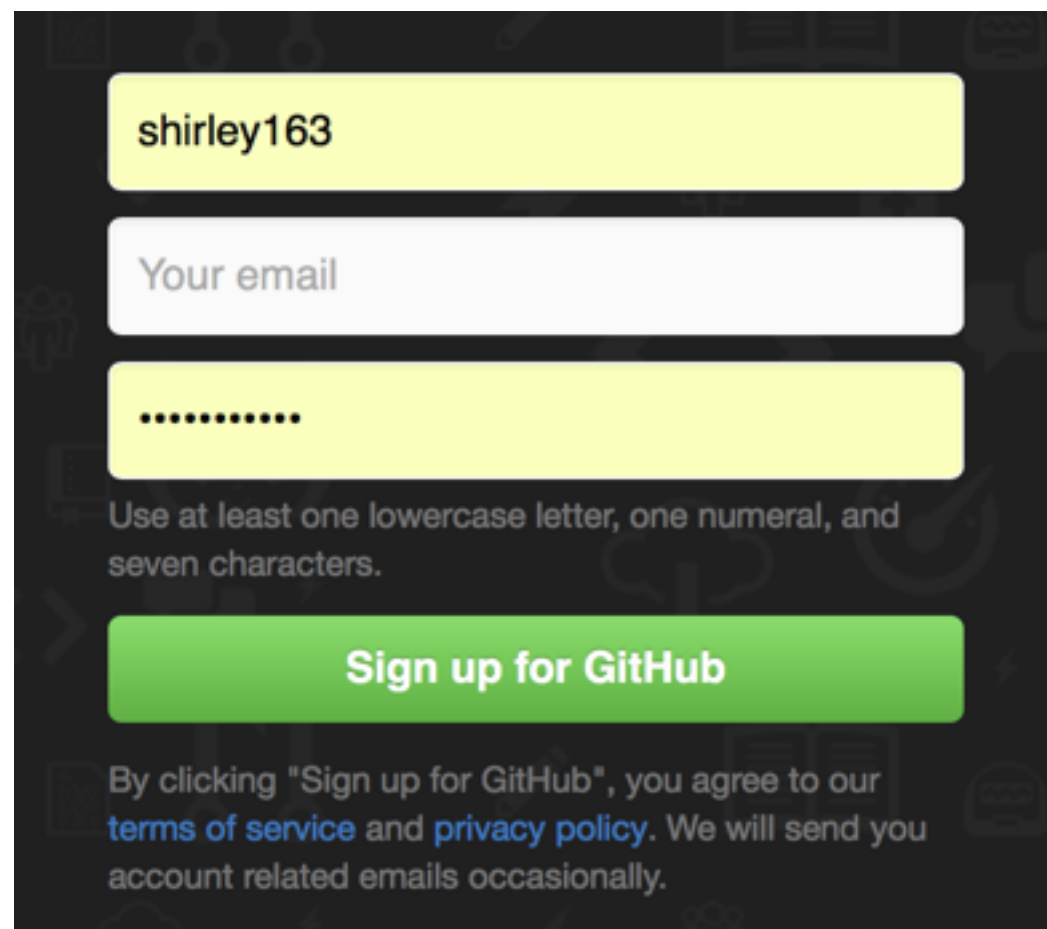
xcode/SVN:

<http://www.cnblogs.com/tsengyuen/archive/2011/03/26/1996615.html>

Xcode中使用git细节(GitHub远程仓库)

步骤:

1. 创建一个账号: www.github.com



The image shows a GitHub sign-up form on a dark background. It features three input fields: a yellow field containing 'shirley163', a white field with the placeholder 'Your email', and another yellow field with a masked password '.....'. Below the password field is a text requirement: 'Use at least one lowercase letter, one numeral, and seven characters.' A prominent green button labeled 'Sign up for GitHub' is positioned below the text. At the bottom, a small disclaimer states: 'By clicking "Sign up for GitHub", you agree to our [terms of service](#) and [privacy policy](#). We will send you account related emails occasionally.'


4. 创建一个远程repository
5. 填写repository信息
6. 点击”Create repository”

Your repositories 0

+ New repository


Owner


Repository name

 shirley163 ▾ /

Great repository names are short and memorable. Need inspiration? How about **furry-octo-archer**.


Description (optional)

☒  **Public**
Anyone can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

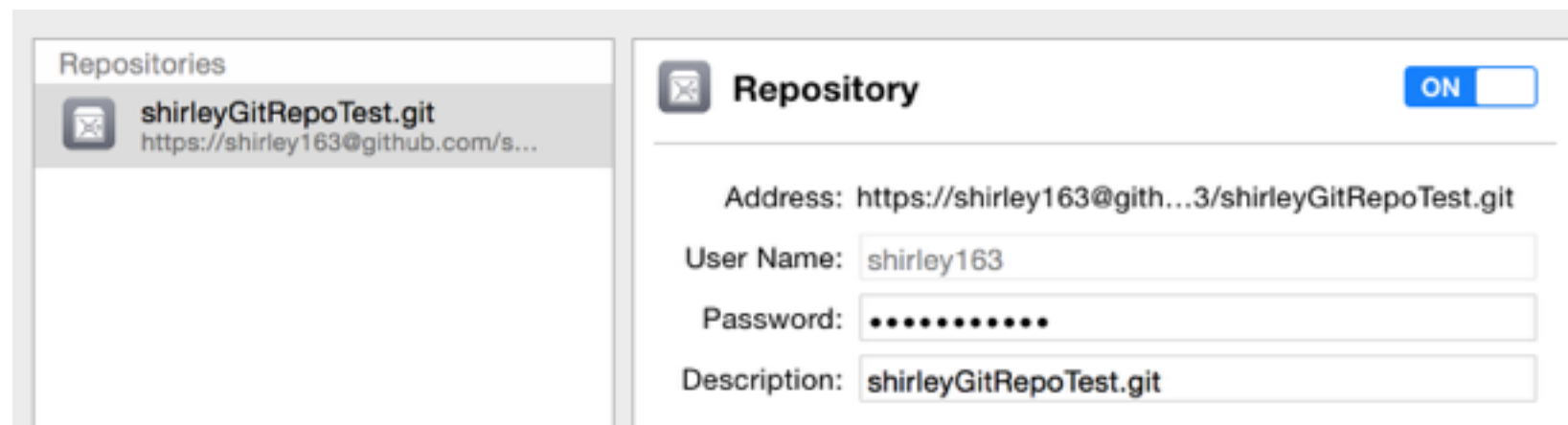
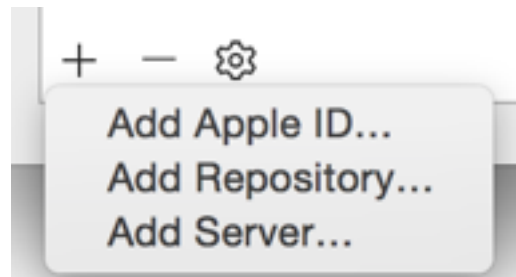
☐ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** ▾

Add a license: **None** ▾ 

Create repository

7. xcode > Preferences添加repository
8. 将github界面上的https url粘贴到Address
9. 并填写账号和密码



10. 创建xcodes，选择Check out an existing project”
11. Check out github上的repository
12. 选择刚才check out的repository



Check out an existing project

Start working on something from an SCM repository.

Check Out

Select the directory in which to check out the project.

Directory: shirleyGitRepoTest

Tags: git


Where: 桌面

Cancel Check Out

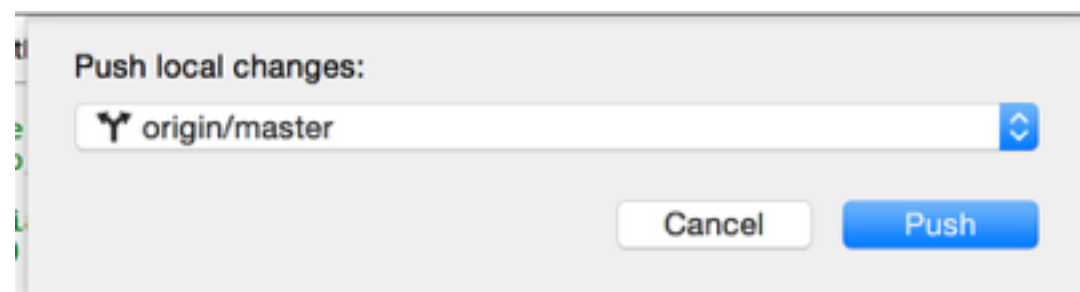
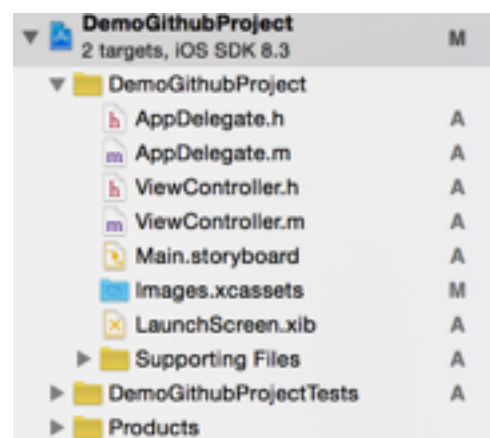


Check Out Successful.

Choose an item:

	Recents	Favorites	Repositories
	shirleyGitRepoTest https://github.com/shirley163/shirleyGitRepoTest.git		

13. 在刚才check out的文件夹下，创建xcode项目
14. 添加/修改代码，Source Control > Commit
15. push本地代码



16. 邮箱确认

You don't have any verified emails. We recommend [verifying](#) at least one email.

Email verification helps our support team verify ownership if you lose account access and allows you to receive all the notifications you ask for.

shirleyfighting163@163.com

Primary

Public

Unverified

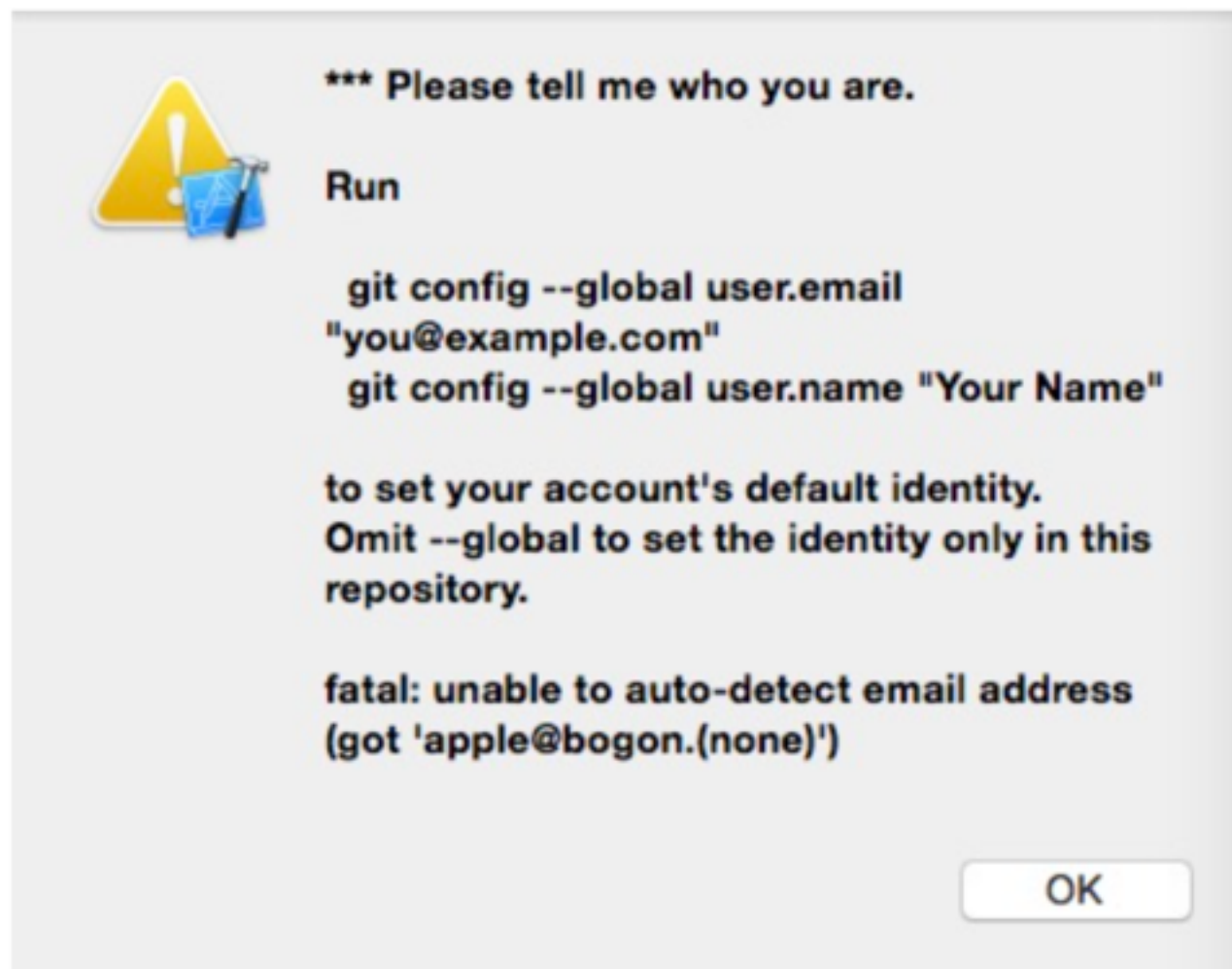
Send verification email



Please confirm verification of shirleyfighting163@163.com.

shirleyfighting163@163.com

Confirm



解决方案：

1. 全局配置邮箱信息, 在终端输入: `git config --global you@gmail.com`
2. 只是针对某个项目做设置: