第一部分
Git

1. Version Control(VS)

   中文叫版本控制系统，出现的目的就是为了方便大规模**协作**。
   - 帮助共享代码
   - 记录每个人修改了什么，代码库究竟发生了什么
   - 比互相用email附件，用微信文件传输更有条理

   常用的Version Control
   - Git，SVN，mercurial…

2. Git

   安装: https://git-scm.com/book/en/v2/Getting-Started-Installing-Git (Mac用户装XCode就好)
   Git是一个distributed system（分布式系统），与centralized的SVN对立。

   独立工作

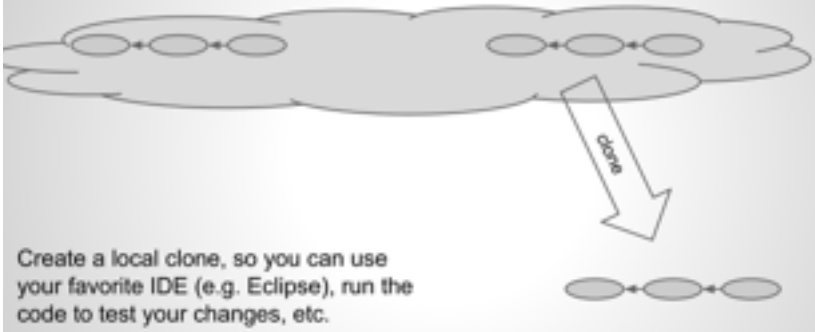
   Commands:
   - git init
   - git status
   - git add
   - git commit -m <message>
   - git log
   - git reset <filename> (unstage)
   - git checkout -- <filename> (undo)
   - git rm <filename>

   问题：what is staging?

   合作

   流程：Fork->Clone->本地操作->Push->Pull Request->Merge

Common Workflow

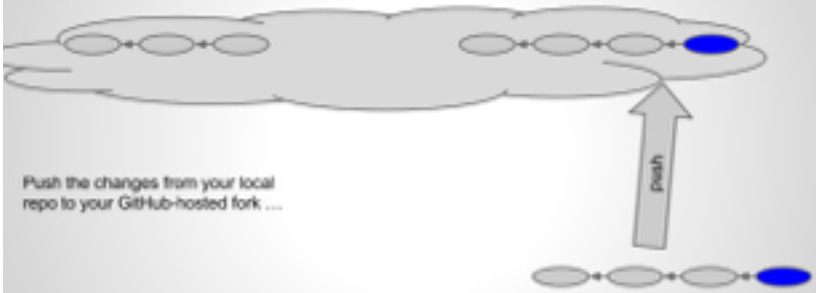


Create a local clone, so you can use your favorite IDE (e.g. Eclipse), run the code to test your changes, etc.

How can you contribute changes?

However, you can't run/test your java code up on GitHub...

Common Workflow



Push the changes from your local repo to your GitHub-hosted fork ....

Common Workflow

Create a *pull-request* ...

Common Workflow

One of the original project's maintainers *merges your pull-request.*

Commands:

git remote add origin <URL>
git push origin master
git pull origin master
git diff HEAD
git diff -- staged

分支 (branching)

创建分支的一大好处在于促进协同。详细的来说，不同的人可以在不同分支上做不同的修改，互不干扰。最终在每个人完成自己的任务后**合并（merge）**代码。比如小明在A分支做前端，大明在B分支做后段。两人协同工作，同步进行，效率更高。当然如果合作不好，不同的分支也很可能会在最终合并的时候产生**分歧（conflict）**。

每一个local copy上面都有一个主干的分支，通常叫做master。最终我们都要把所有的修改都整合到这个branch上面，最终再pull或者pull request。其他分支的名字可以由我们自己确定，但是主干分支一般只能叫做master。

Commands:

新建分支：git branch <branch-name>
切换到某分支：git checkout <branch-name>
整合修改（在master上进行）：git merge <branch-name>
删除分支：git branch -d <branch-name>

冲突解决 (Conflict Resolve)

Conflict有两种，一种是不同分支造成的，另一种是不同人在同一分支工作造成的。
这里说的Conflict，指的是不同人commit的分歧，而非不同分支的分歧。
Conflict是真正使用VC**最容易出现**，也是**最难解决**的问题！考试方面这个也是重点。
Conflict出现的**根本原因**有两方面：

1. 团队合作没有商量好
2. 没有及时pull/push

Conflict的解决需要：

1. 两个人商量好如何修改
2. 其中一个人修改好，并且**及时**push
3. 另一个人**及时**pull

***版本的穿梭**

非考试必要，但是很有趣，参见“廖雪峰”。

实用的东西：Github Desktop App、Sourcetree… 总之是脱离了command line，回到了界面窗口化

结语：Git功能很强大，使用需谨慎。光知道理论不够，还得多实践。即便理论背的滚啦烂熟，用的不多，照样蒙逼。