



南京大學
NANJING UNIVERSITY

软件工程实验六: 项目协同开发管理与工具集成环境实验

SE-2021-autumn Lab6

朱庭伟

tingweizhu33@smail.nju.edu.cn

2021 年 10 月 21 日

实验目的

- 了解协同开发与持续集成过程
- 学会使用项目协同开发管理工具 git/github
- 了解持续集成并使用 jenkins 自动构建项目

协同开发

协同

■ 多版本，多人，多设备

- 多版本：管理一个项目的多个版本及其修改记录
- 多人：多人合作开发同一个项目
- 多设备：在不同设备上持续开发同一个项目

I 分布式版本控制系统

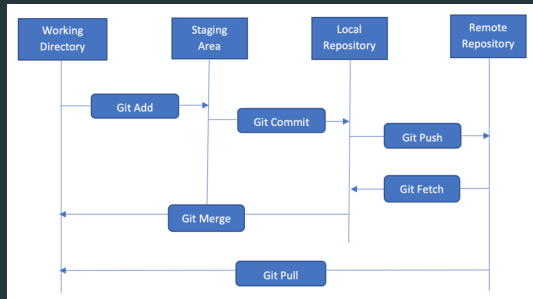


HEAD

- 指向 commit 对象的可变指针

Workflow

- 工作区 (Working Directory)
- 暂存区 (Stage/Index)
- 版本库 (Repository)



版本控制

初始化

```
$ git init
```



提交

```
$ git add .  
$ git commit -m "update"
```



check 一下

```
$ git status  
$ git diff  
$ git log
```



回退

```
$ git reset
```



分支管理

创建分支

■ `$ git checkout -b`

合并分支

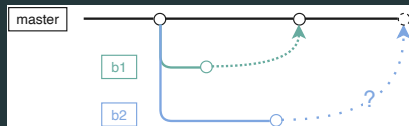
■ `$ git merge`

解决冲突

■ 两个分支对同一文件各有新的提交时

check 一下

■ `$ git log --graph`

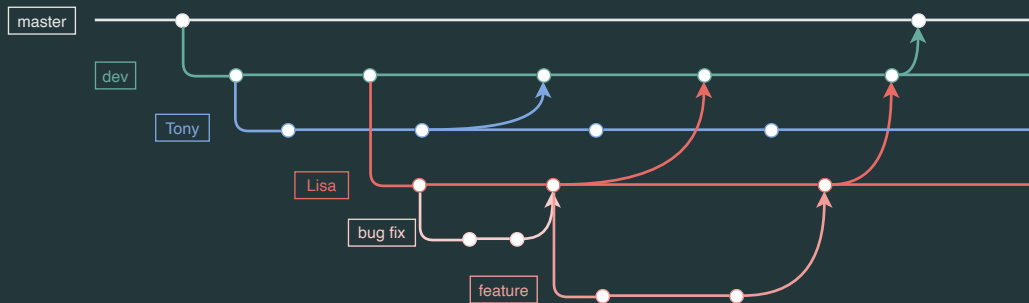


- 若 b1 和 b2 修改了不同文件，可以自动合并
- 若 b1 和 b2 修改了相同文件，需要解决冲突

```
<<<<<<< HEAD
Hello b1
=====
Hello b2
>>>>>> b2
```

```
* abf6c69 (HEAD -> master) Merge branch 'b2'
| \
| * 4bbb1f2 (b2) add b2
* | 0503fa3 (b1) add b1
|/
* 849a047 first commit
```

开发策略

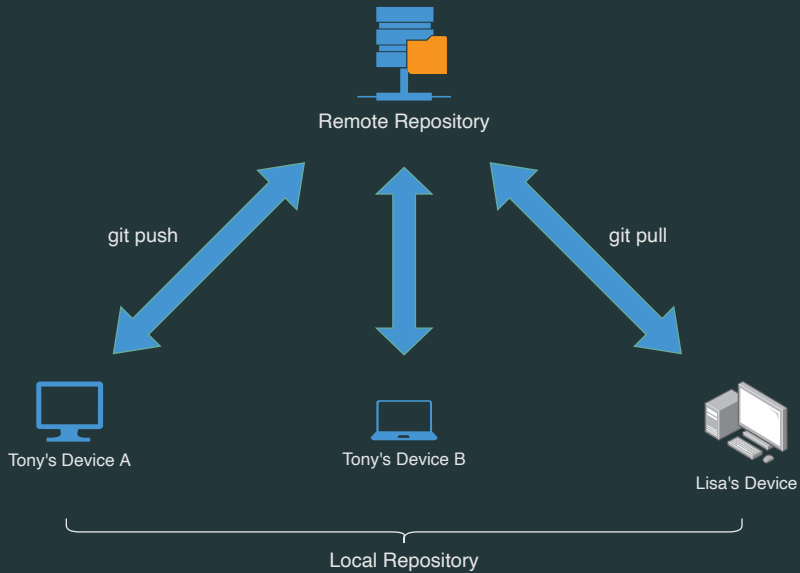


标签



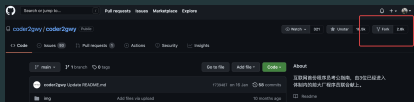
```
$ git tag v1.0 as6ge4
```

远程仓库

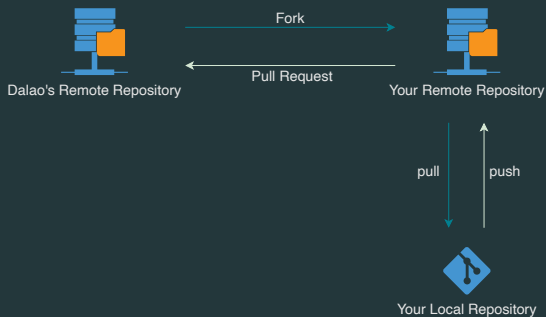
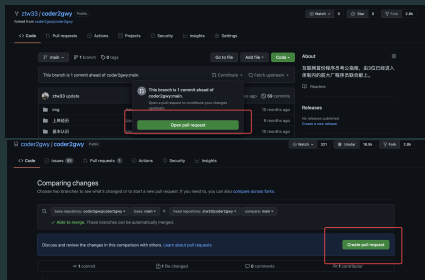


向开源社区贡献你的代码!


Fork



Pull Request



实验内容 (可参考的教程: git 官方文档 [1], 廖雪峰的教程 [2])

- 安装 git, 在本地将你的开源项目目录初始化为 git 仓库 (如已有 .git 文件夹请先删除)
- 在本地尝试修改、提交、回退等过程, 在报告中展示你的操作, 并使用 `git diff`, `git log`, `git status` 等命令展示操作前后的区别
- 根据实验三针对几个页面进行微调的任务, 在本地为每个子任务创建一个分支并在各分支上进行开发, 最终将所有修改合并到 master 分支上; 如有冲突请尝试解决。在报告中展示你的操作, 并使用 `git log --graph` 命令展示分支合并图
- 给你的某个稳定版本的代码打上标签
- 注册 github 账号, 在账号中创建远程仓库 (权限请设置为 public); 把本地的所有分支和标签推送到远端
- 使用 pull request 提交自己的代码和报告 (作业提交说明里细 )
- 在报告中回答以下问题:
 - 使用 git 的好处?
 - 使用远程仓库 (如 github/gitee 等) 的好处?
 - 在开发中使用分支的好处? 你在实际开发中有哪些体会和经验?
- 如果你额外学习并实践了关于 git/github 的其他进阶操作 (如 merge 和 rebase 的区别、reset 和 revert 的区别、stash, cherry-pick 的使用等), 可在报告中展示

持续集成

■ 提交 → 测试 (第一轮) → 构建 → 测试 (第二轮) → 部署 → 回滚

- 提交：开发者向代码仓库提交代码
- 测试 (第一轮)：代码仓库中对 commit 操作配置了钩子 (hook)，提交代码后跑自动化测试，主要是单元测试
- 构建：将源码转换为可以运行的实际代码 (如安装依赖、配置资源等)
- 测试 (第二轮)：全面测试 (单元测试，集成测试，端对端测试)
- 部署：将可以部署的代码版本的所有文件打包存档，发到生产服务器
- 回滚：当前版本发生问题，回滚到上一个版本的构建结果

Jenkins 是一个独立的开源软件项目，是基于 Java 开发的一种持续集成工具，用于监控持续重复的工作，旨在提供一个开放易用的软件平台，使软件的持续集成变成可能。

主要用于：

- 持续、自动地构建/测试软件项目
- 监控一些定时执行的任务

⚠ (不作要求, 若完成可额外增加分数)

- 在本机安装 jenkins, 并在全局工具配置和系统设置中配置好 JDK 地址、Gradle 地址、ANDROID_HOME 地址和 JAVA_HOME 地址
- 新建任务, 在源码管理中填写自己项目的 github 地址, 对项目进行一次构建
- 修改代码再次推送到 github 仓库中, 再次对项目进行构建

实验提交要求

提交内容：实验三的代码 & 实验报告

提交仓库 (及作业提交流程): <https://github.com/ztw33/NJU-SE2021-autumn-Lab6>

报告格式：Markdown

推荐的 md 文件编辑器: `vscode` / `typora`
简单的小教程 [4]

参考 i



Git.

Git reference.

<https://git-scm.com/docs>.



廖雪峰.

Git 教程.

<https://www.liaoxuefeng.com/wiki/896043488029600>.



阮一峰.

持续集成是什么?

<https://www.ruanyifeng.com/blog/2015/09/continuous-integration.html>.



简书.

献给写作者的 markdown 新手指南.

<https://www.jianshu.com/p/q81RER>.