git使用指南

1.Git简介

1.1 Git是什么，有什么用

Git (/gɪt/)是一个**开源的分布式版本控制系统**，用于敏捷高效地处理任何或小或大的项目。

举个简单的例子，假如某个团队需要共同完成一个代码项目，那么一个人如何知道整个团队最新的代码是谁写的？更新了哪里？口头交流是低效的，而且每次修改完代码还得把最新版代码发给所有人以同步版本。何况对比和查看新旧代码也不方便。此时我们使用git来更好地**云端备份文件**、**整合更新内容**、**回溯过去的版本**、**对比查看代码更新细节**等。

下面的内容顺序看完即可轻易理解，跳着看容易看乱。

本文章仅用于介绍git工具，示例不代表实验中真正使用git的过程，实验的流程和具体的命令使用会在实验课课件中详细给出（以防同学们直接对着实验仓库运行本文代码）

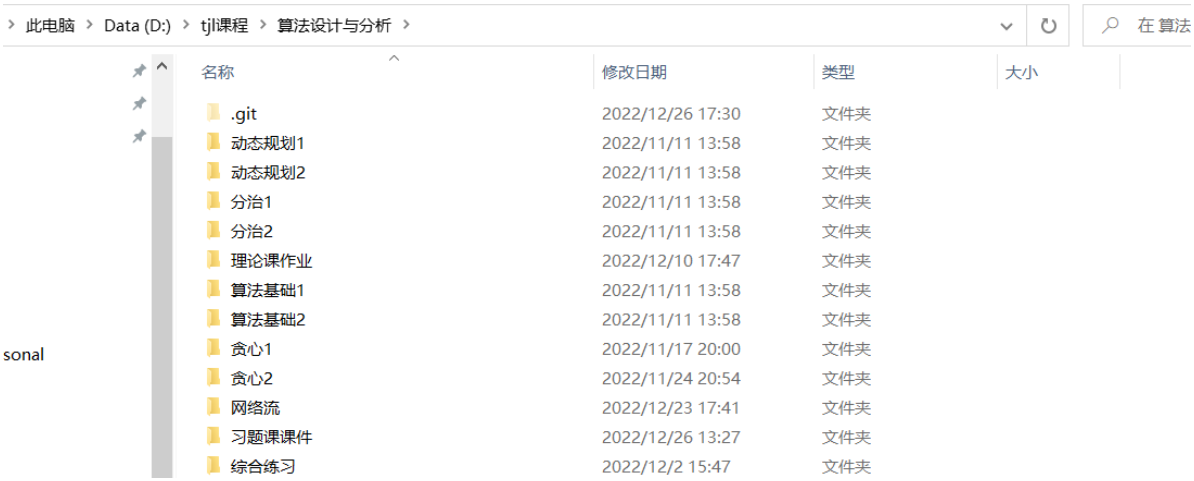
1.2 Git是如何工作的

要明白Git是如何工作的，首先需要介绍几个名词，为方便理解，仍然举刚刚的例子（一个团队写一个代码项目。假设项目里有 `wuziqi.c` 和 `cod.c` 文件，团队有我和G两个人）

本地仓库（local）：首先这些代码文件在本地肯定是要有的，不然你怎么编辑代码呢。假设 `wuziqi.c` 和 `cod.c` 都存放在一个叫 repository 的目录中，这个目录就是一个仓库（大伙可以把目录理解为文件夹，这个文件夹在你自己电脑上），称为**本地仓库**。

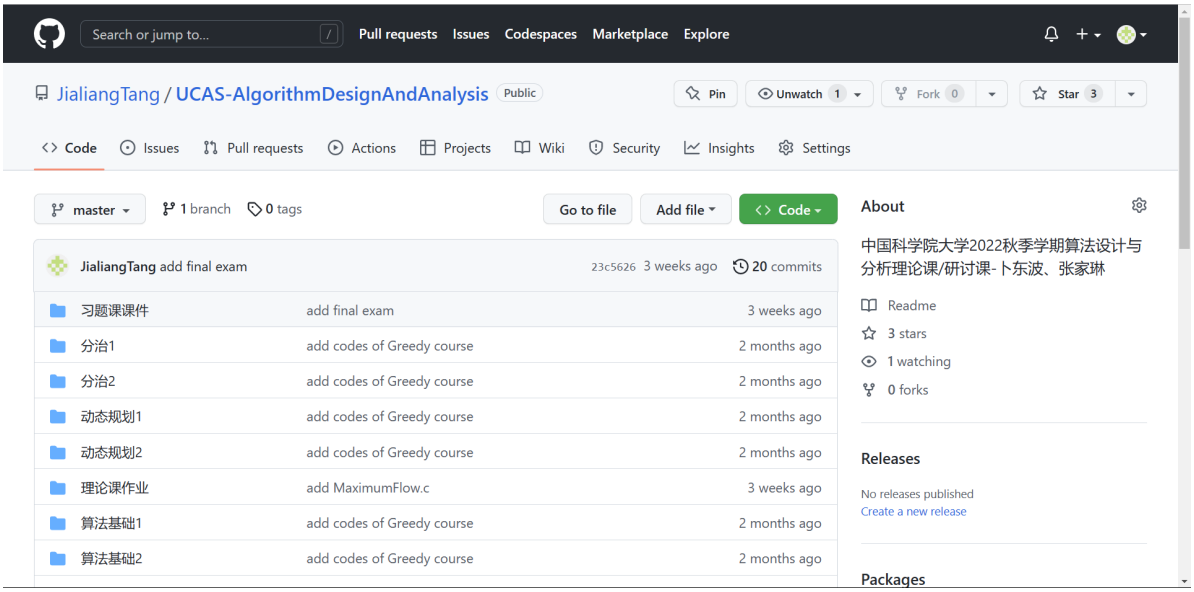
需要注意的是，一般来说本地仓库中还会有一个隐藏目录`.git`，这个目录包含了仓库的配置文件、版本控制信息等必要信息，**可以无视，但是别动。**

本地仓库的内容 = `.git`目录 + 所有需要管理的文件（可以是任何文件或者目录），比如在我电脑上的一个本地仓库：



远程上游仓库（upstream）：现在，假设W老师要求我们写一个五子棋程序，他在云端放上了原始代码框架，那么我们需要做的是把框架弄到本地来。首先我们需要将这个仓库设置为我们的远程上游仓库（git remote add upstream），然后在每次实验前拉取框架代码更新（git pull upstream master）。其中**pull命令**是专门负责拉取远程仓库内容的命令，它会把框架拉取到本地仓库里。

远程仓库（remote）：既然要统一管理代码，那就需要一个云端的仓库存放代码，这个仓库就是**远程仓库**，远程仓库是我和G都可以在网上访问的，可以直接在网上点击浏览或者下载远程仓库里的文件。有很多网站提供远程仓库服务，如github，gitee等，计组实验中，我们用的是SERVE云平台。**注意：事实上，pull命令对远程仓库也奏效。**例如，我在github中的某个远程仓库界面如下：



好了，接下来我开始写代码，写完了，我想把代码同步给G，那么我就需要把代码上传到远程仓库，核心思想就是保证远程仓库始终是最新代码，每次写完就同步到远程仓库。这个上传过程涉及到的命令和名词如下：

工作区：就是你的本地仓库。

暂存区：把对工作区文件的修改暂存、收集。实现了多份文件修改作为一次commit。

add命令：采用add命令将工作区最新代码加入到暂存区

commit命令： 将暂存区的修改信息提交，注意此时还没有更新远程仓库。提交信息将会显示在你的远程仓库活动记录中。比如，github会有这样的显示：

网络流 add MaximumFlow.c 3 weeks ago

其中网络流是我本地仓库中的一个目录，中间的文字是最近一次commit的信息，右边是更新时间。

push命令： 将暂存区内容提交到远程仓库，完成远程仓库的更新。

命令使用示例：

```
git add wuziqi.c cod.c //将修改过的wuziqi.c和cod.c加入暂存区
git commit -m "add new codes of wuziqi.c and cod.c" //将暂存区修改信息提交，-m参数后面双引号里面是修改信息，内容可以自己随便写，但建议写本次修改的说明。不加-m会自动弹出vim让你写修改信息。
git push origin master //将暂存区的修改内容提交到远程仓库，完成更新
```

为了方便起见，你可以直接用

```
git add .
```

来一次性将该目录下所有修改文件加入暂存区，而不需要再打那么多文件名了。

注：常见的操作是add 多个文件之后commit一次；commit多次之后push一次。你也可以就单纯add一次之后commit一次，commit一次之后push一次，就像上面那三行代码演示的那样。

1.3 Git 如何实现项目文件共享

然后，G想改一改代码。他首先要做的不是直接打开本地的代码文件进行修改，而是用git pull origin master拉取远程仓库最新文件，然后G的本地文件就被更新为最新文件了，她在这些文件的基础上修改即可。修改完记得add commit push一顿操作提交到远程仓库，方便我下次再改，如此往复，就是一个最简单的开发流程。

本课程中，由于老师会不断更新实验框架，大家每次写代码之前，第一件事是需要git pull upstream master获取最新实验框架，而由于整个仓库是私有的，没有其它人会修改你的代码，所以同学们每次不需要拉取远程仓库最新文件了。**除非**：你在本地写烂了，那可以删了本地的文件重新git pull origin master以获取远程仓库的代码，这就是将远程仓库视作你代码的备份，当然，你也可以直接在远程仓库网站上点击下载。

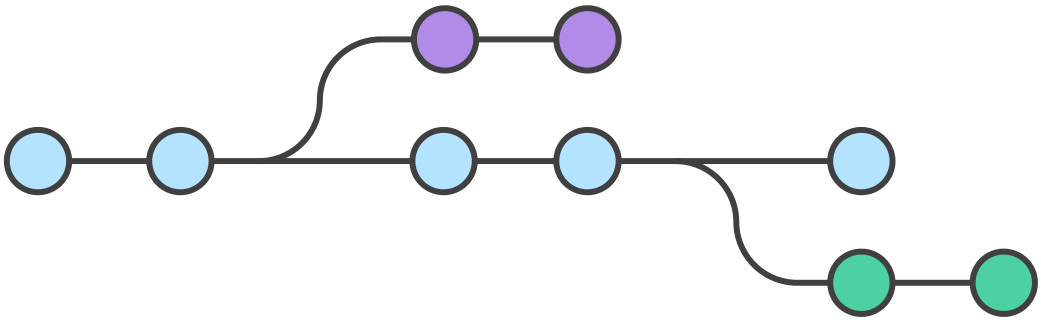
本课程的具体仓库结构示意图见后文。

1.4 Git 使用进阶——分支、标签和回退

1.4.1 分支

Git的精髓在于**分支（branch）**，分支的作用是为**较大规模的尝试性修改**建立新分支，对修改结果满意后再合并到主分支，而且在比 commit 更大的粒度上管理代码的修改记录，方便撤销一系

列针对同一功能的 commit 记录。



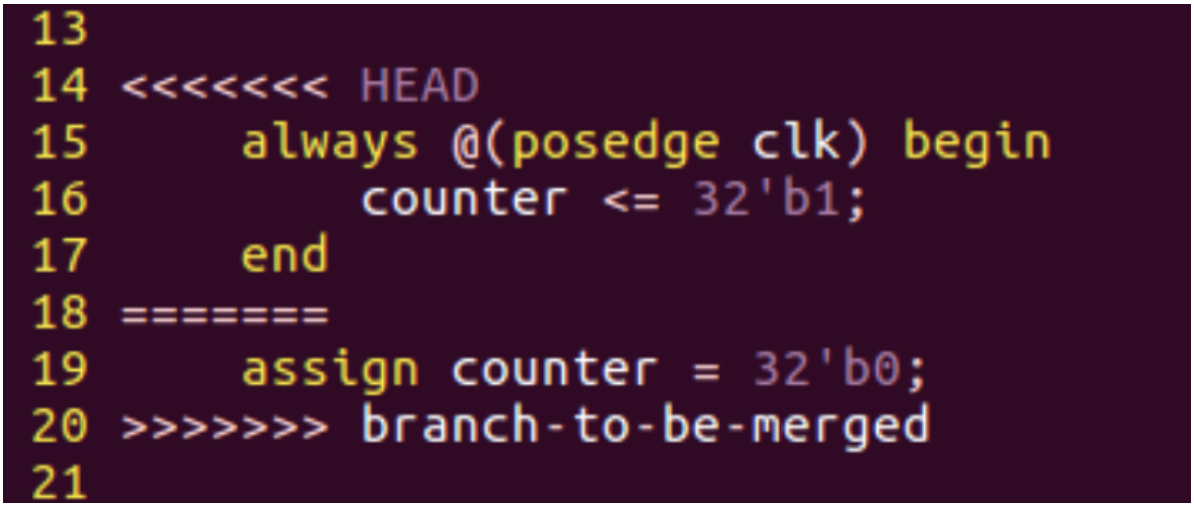
关于分支的操作较为复杂，可以参考如下链接：[Git 分支管理 | 菜鸟教程 (runoob.com)](), 里面讲得很细，可以尝试复现，熟悉分支管理。

常用分支命令：

```
git branch //查看分支
git branch <branch-name> //基于当前所在分支创建分支
git checkout <branch-name> //跳转到某分支，加-b就是创建并跳转
git branch -D <branch-name> //（强制）删除分支
git merge //合并分支
```

再说明一些菜鸟教程没有给出的信息：

1.`git status`可查看冲突文件。冲突片段具有如图所示的格式，上半部分是当前分支的修改，下半部分是要合并分支的修改。



2. `git merge --abort` 可以放弃此次合并。

注：本课程中，你完全可以不用分支，但是一定会用到下面要讲的标签。

1.4.2 标签

我们会给仓库历史中的某一个commit打上**标签（tag）**以表示重要。人们会使用这个功能来标记发布结点（v1.0、v2.0等等）

常用标签命令：

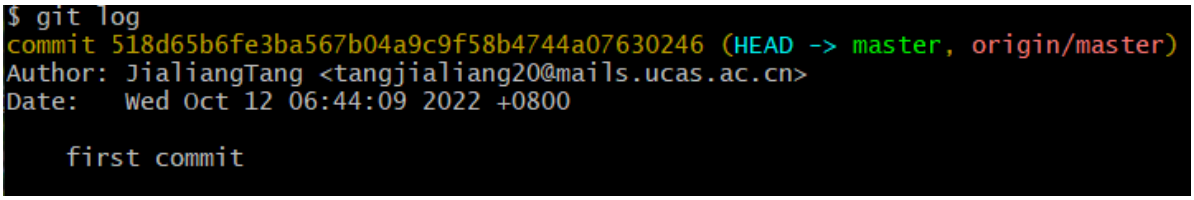
```
git tag //列出标签
git tag -a <tag-name> [commit hash] -m <message> //基于当前commit或某一确定commit创建标签
git show <tag-name> //显示该标签及其对应的commit信息
```

本课程需要你们为每次实验最终文件添加标签，具体命令和操作步骤将会在各个实验的课件中给出。

1.4.3 回退/撤销

如果当前代码的修改不尽人意，git所提供的撤销和回退操作可以快速丢弃修改，返回到指定的代码版本。这也是版本管理的重要意义之一。

`git log` 可查看 commit 的 hash值用于指定要处理的commit，就是下面黄色一长串那个。

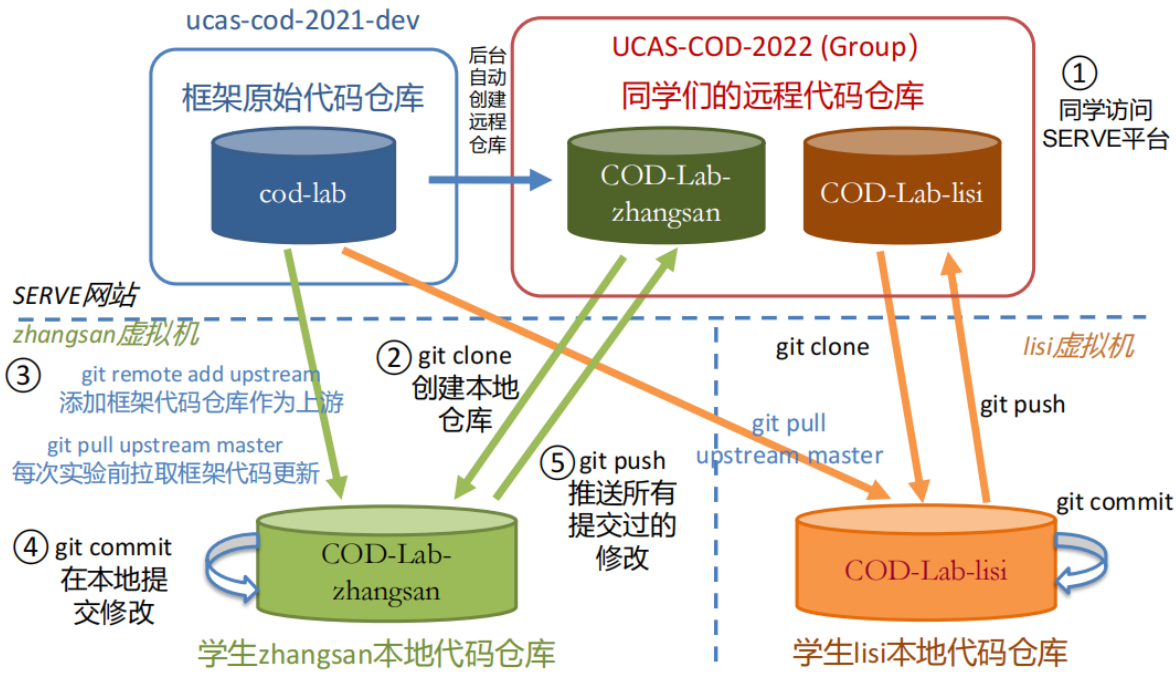


用`git checkout -- <file1> <dir1> ...`可以撤销未进入暂存区的修改。

用`git reset <file1> ...`可以将文件移出暂存区。

用`git reset <commit hash>`回退到指定的commit版本。注意，回退路径上的所有修改都会移入暂存区，命令选项`--hard`可以直接抛弃这些修改。

2. 本课程所用到的的仓库结构



如图，对应前文讲述的各仓库和命令来理解。

3.Git 常用查询命令和基本开发流程（补充）

常用命令如下：

git log

- 查看当前分支的提交记录（按q退出）
- 选项 --graph显示commit tree
- 选项 -p 显示每个文件的diff

```
* 3441368aae7d7e13d78166793c551dd14d0f89da (HEAD -> master, tag: v1.1, tag: v1.0) change test
* d8286747185f181ab557262b820a9a1c694dba39 (tag: v1.3) add test
```

git status

- 显示当前分支名、暂存区中的文件、已修改的文件、未跟踪的文件等当前环境信息

git diff

- 查看未添加到暂存区中的修改内容
- 选项 --cached 则查看暂存区中文件的修改内容
- git difftool 默认调用 vimdiff 进行并排对比

开发流程如下：

`git add <file1> <file2> …` 添加修改和新建文件到暂存区

`git commit` 将暂存区中的修改信息提交

- 不加参数 `-m` 会自动启动编辑器撰写提交信息
  - 指定编辑器（以Vim为例）：`git config --global core.editor vim`
- 一站式快捷命令：`git commit -a -m 'title'`

`git push <repo> <branch>` 推送最新修改到远程仓库

- 添加仓库 `git remote add <name> <url.git>`
- 分支名要与本地仓库当前分支保持一致

参考资料：

Git 教程 | 菜鸟教程 (runoob.com)

ngc7331/UCAS-CS-Guide: 一份面向UCAS本科计算机科学与技术专业同学的基础指南 (github.com)

作者：唐嘉良 中国科学院大学 计算机科学与技术 2006班

联系方式：QQ2366807198，微信tjl2366807198