

一、本地git

1、版本管理工具

可以对代码或文件资料等进行版本管理

常见的版本管理工具：SVN，git

2、git常识

git的语言是unix和linux的语法基本一样，linux是在unix的基础上优化的

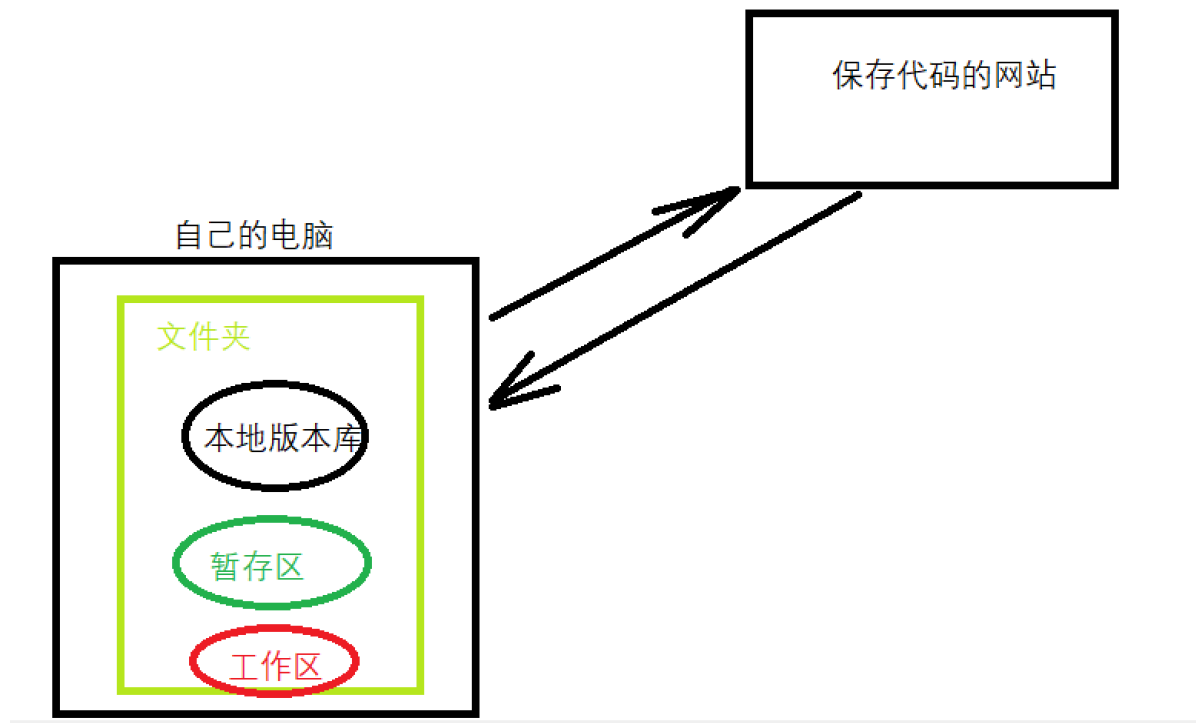
可以将文件和资料上传到网上进行保存

github 是一个国际网站，用于保存代码（上传和下载的速度会比较慢）

国内也有类似的网站，学习就操作国内的网站，码云。

3、git

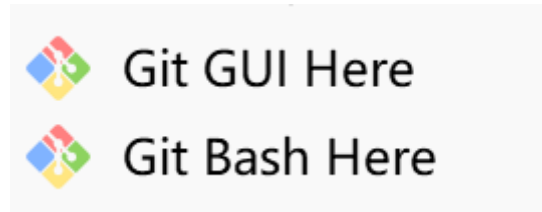
git的基本模型



4、安装git

按顺序安装：

- TortoiseGit-2.10.0.0-64bit.msi
- TortoiseGit-LanguagePack-2.10.0.0-64bit-zh_CN.msi （中文语言包）
- Git-2.32.0-64-bit.exe 操作界面
- 安装完成后，在桌面空白处点击右键，出现下图内容表示成功



5、打开git

在桌面空白处点击右键，选择git bash here 打开命令操作界面



命令界面解释：

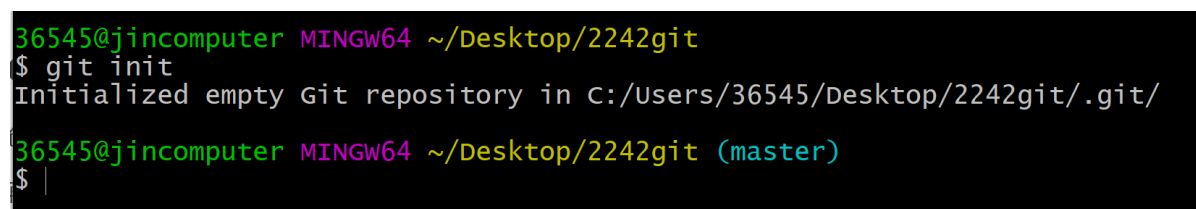


6、创建本地版本库

创建一个文件夹，进入到文件夹里面，点击右键，选择git bash here

再输入：git init 进行初始化

初始化后会自动在这个文件夹中创建一个 .git 的文件夹，这个文件夹不可以删除



7、设置用户信息

git需要表名身份后才可以进行操作

```
设置用户名: $ git config --global user.name '用户名'    #注意用户名和usr.nmae中间是有空格的
设置用户邮箱: $ git config --global user.email '邮箱'
查看当前用户名和邮箱地址: git config --global --list
```

8、版本库的操作

查看文件文件的状态:

```
git status
```

红色的是工作区的文件: 增加, 修改或删除的文件记录会在工作区中
绿色的是暂存区的文件: 暂存区的文件, 在下次提交的时候会全部提交到版本库中
非红色或绿色的文件都是在本地版本库中

提交暂存区:

```
git add 文件名
```

```
git add .    #将当前工作区的所有文件提交到暂存区
```

提交到本地版本库:

提交到本地版本库的文件必须是暂存区的, 工作区的文件无法直接提交到本地版本库

提交到本地版本库的文件或代码时, 都必须输入注释。注释更新内容是什么

```
git commit    #执行后会进入到备注窗口, 输入注释后保存退出即提交成功
```

快速提交:

```
git commit -m '备注信息'
```

9、删除文件

使用 `git rm 文件名` 删除本地版本库的文件: 会直接将删除的记录提交到暂存区

使用 `rm -rf 文件名` 命令去删除本地版本库的文件: 会将删除的记录提交到工作区

10、还原

本地版本库中的文件被修改过后，可以进行还原

将本地版本库中的文件还原会到上一个版本

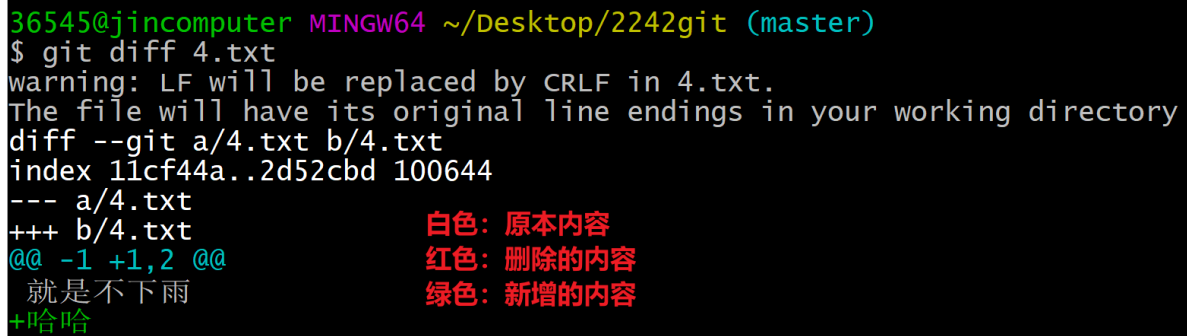
暂存区的文件发生了改变后，也可以使用命令进行还原

```
git checkout 文件名      #撤销改动
git checkout .            #撤销所有文件的改动
```

11、查看文件的修改历史

操作：将本地版本库中的文件进行修改，然后使用查看修改历史的命令来查看具体修改的内容

```
git diff 文件名
```



```
36545@jincomputer MINGW64 ~/Desktop/2242git (master)
$ git diff 4.txt
warning: LF will be replaced by CRLF in 4.txt.
The file will have its original line endings in your working directory
diff --git a/4.txt b/4.txt
index 11cf44a..2d52cbd 100644
--- a/4.txt
+++ b/4.txt
@@ -1,2 @@
-就是不下雨
+哈哈
```

白色：原本内容
红色：删除的内容
绿色：新增的内容

12、查看不同版本的修改内容

操作：将本地版本库中的文件进行修改，然后在将修改后的文件提交到本地版本库，然后使用命令查看修改的历史

```
git diff head~num  #当前版本与前num个版本对比修改内容
$ git diff head~2  #当前版本与前2个版本的代码内容对比
```

13、查看版本日志

```
git log  #完整版日志，倒序查看版本日志，最新的在前面
git log --pretty=oneline  #简化版日志
```

```
36545@jincomputer MINGW64 ~/Desktop/2242git (master) 版本号
$ git log
commit cb405aac39958f182f6de330eede446dc22c3856 (HEAD -> master)
Author: zhangsan <123@qq.com> 这个版本提交的人
Date: Thu May 12 14:28:05 2022 +0800 提交的时间

    第二次修改了4.txt 提交的备注
```

14、切换版本

将当前版本切换到对应的版本中

```
git reset --hard 版本号
```

举例：

```
$ git reset --hard cb405aac399
```

15、切换版本的记录

```
git reflog
```

```
cb405aac39958f182f6de330eede446dc22c3856 (HEAD -> master) 第二次修改了4.txt
c0dfc4b2964155f96eb7e01214f23d3d4810b9af 第一次修改4.txt
4a3ac33e3eae67dd91be2f648b38229e161779b6 新建4.txt
b70f25c383815754e21e516f5ca37b028b75fbcf 又删除2.txt
5427cb1c23e98590f9a3e01b9789844428702f7c 删除1.txt
a212640fffd457bacb0d48ddbc8eb6b887fc9b92 第二个版本
808eb645f1c3446731544789ba62cc475ec4e617 周四的第一个版本

36545@jincomputer MINGW64 ~/Desktop/2242git (master)
$ git reflog
cb405aa (HEAD -> master) HEAD@{0}: reset: moving to cb405aac399
a212640 HEAD@{1}: reset: moving to a212640fffd4
cb405aa (HEAD -> master) HEAD@{2}: commit: 第二次修改了4.txt
c0dfc4b HEAD@{3}: commit: 第一次修改4.txt
4a3ac33 HEAD@{4}: commit: 新建4.txt
b70f25c HEAD@{5}: commit: 又删除2.txt
5427cb1 HEAD@{6}: commit: 删除1.txt
a212640 HEAD@{7}: commit: 第二个版本
808eb64 HEAD@{8}: commit (initial): 周四的第一个版本

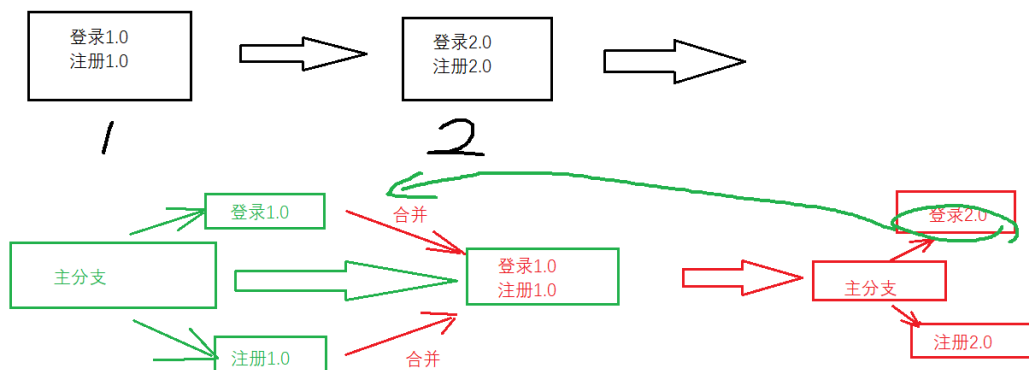
36545@jincomputer MINGW64 ~/Desktop/2242git (master)
```

二、分支

1、分支概念

在初始化git的时候，会默认创建一个主分支，叫：master

分布式开发：开发多个模块，通过创建多个分支的方式开开发，每个分支都有自己的版本。然后选择不同的分支进行合并成一个新的版本



2、创建分支

```
git branch 分支名
```

创建分支时，会将当前分支的所有内容复制到新的分支中
在新的分支中可以再次创建新的分支
如果在没有提交过本地版本库内容的分支中，再创建新的分支会报错

3、切换分支

```
git checkout 分支名
```

在切换之前要把工作区、暂存区的内容提交到本地版本库，不然在切换到的时候，会将未提交的内容带到新的分支中
如果切换的分支不存在则创建一个新的分支

将5.txt的文件提交到master 的本地版本库中，然后切换到dev分支中，查看dev分支中是否有5.txt文件

4、删除分支

```
git branch -d 分支名
```

5、查看所有分支

```
git branch
```

6、合并分支

可以将分支进行合并

合并的内容只能合并版本库中的内容，不可以合并暂存区和工作区的内容

```
git merge 分支名
```

7、合并冲突

两个分支中的相同文件名的文件，在合并时，如果文件里面的内容不一致，在合并的时候会产生冲突，如下图

```
36545@jincomputer MINGW64 ~/Desktop/2242git (master)
$ git merge dev
Auto-merging 6.txt
CONFLICT (content): Merge conflict in 6.txt
Automatic merge failed; fix conflicts and then commit the result.

36545@jincomputer MINGW64 ~/Desktop/2242git (master|MERGING)
$
```

8、解决冲突

可以进入到冲突的文件中，将冲突的内容修改，删除到不需要保留的内容，保存退出后将该文件提交到本地版本库中即可解决冲突

三、远程

1、在码云上创建一个仓库

- 在码云上注册一个账号
地址: <https://gitee.com/>
- 登录后右上角点击'+', 选择新建仓库

2、连接仓库

- https方式连接: 每次操作仓库都需要输入账号和密码
- SSH方式连接: 使用公钥的方式进行连接, 在以后操作仓库的时候就不需要每次输入账号和密码

本地生成公钥

在git中执行一个命令生成:

```
ssh-keygen -t rsa -C '自己的邮箱'
```

输入命令后一直按enter键直到生成为止

找到公钥 (id_rsa.pub) 的存放路径, 打开公钥文件, 复制里面的所有内容

打开码云--> 设置-->SSH公钥-->将复制的公钥粘贴进去-->输入密码即添加成功

仓库首选项

安全设置

SSH公钥

GPG公钥 Beta

私人令牌

登录历史

流水线 Gitee go

标题

公钥标题(key)

公钥

把你的公钥粘贴到这里, 查看 怎样生成公钥

支持以 'ssh-rsa', 'ssh-dss', 'ssh-ed25519', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', 'ecdsa-sha2-nistp521' 格式生成公钥

填入公钥

建立与码云的远程连接

输入命令创建连接:

```
git remote add 连接名 远程连接的路径 #路径需要到码云上复制SSH的连接
举例:
$ git remote add gz2242 git@gitee.com:jcszhi/gz2242git.git
```

删除连接:

```
git remote remove 连接名
```

jcsz / gz2242git

代码

Issues 0

Pull Requests 0

Wiki

快速设置— 如果你知道该怎么操作, 直接使用下面的地址

HTTPS

SSH

git@gitee.com:jcszhi/gz2242git.git

点击复制

我们强烈建议所有的git仓库都有一个 README, LICENSE, .gitignore 文件

初始化 readme 文件

Git入门? 查看 帮助, Visual Studio / TortoiseGit / Eclipse / Xcode 下如何连接本站, 如何导入仓库

2、推送

把本地的文件资料推送到远程仓库中

注意: 在推送的时候, 要先拉取一下远程仓库的代码后与本地的合并后, 在进行推送。如果不这样, 可能会导致推送冲突。(你在推送代码时, 其他人先一步推送了代码, 此时你如果没有先拉取最新的代码而直接推送的话, 就会产生推送冲突)


```
git push -u 连接名 推送的分支 #推送
```

举例:

```
$ git push -u gz2242 master
```

3、拉取

从远程仓库中拉取代码到本地

```
git pull 连接名 分支名 --allow-unrelated-histories #将远程仓库中的内容拉取到本地中，并且自动与本地中的代码合并
```

```
--allow-unrelated-histories #忽略本地代码和远程仓库的代码的差异进行合并
```