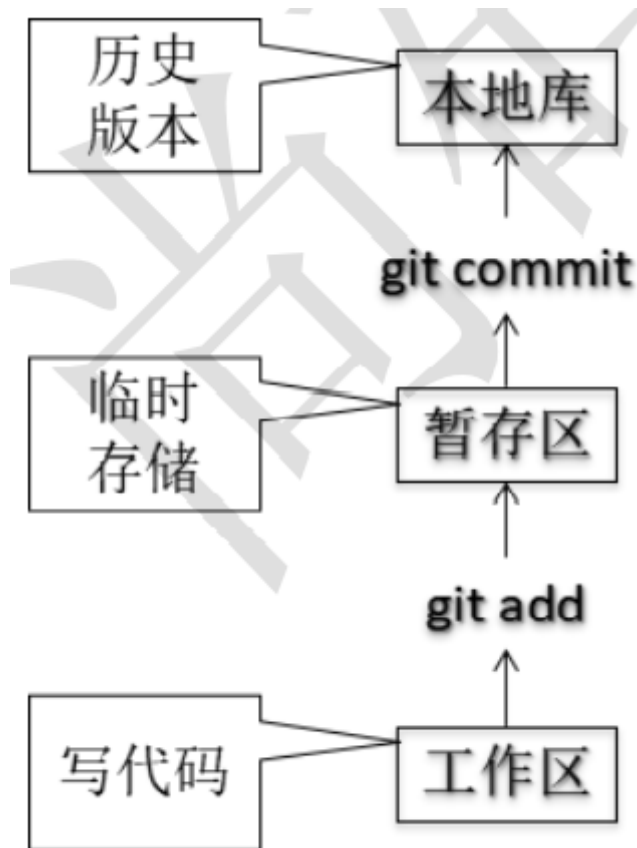


1.Git结构

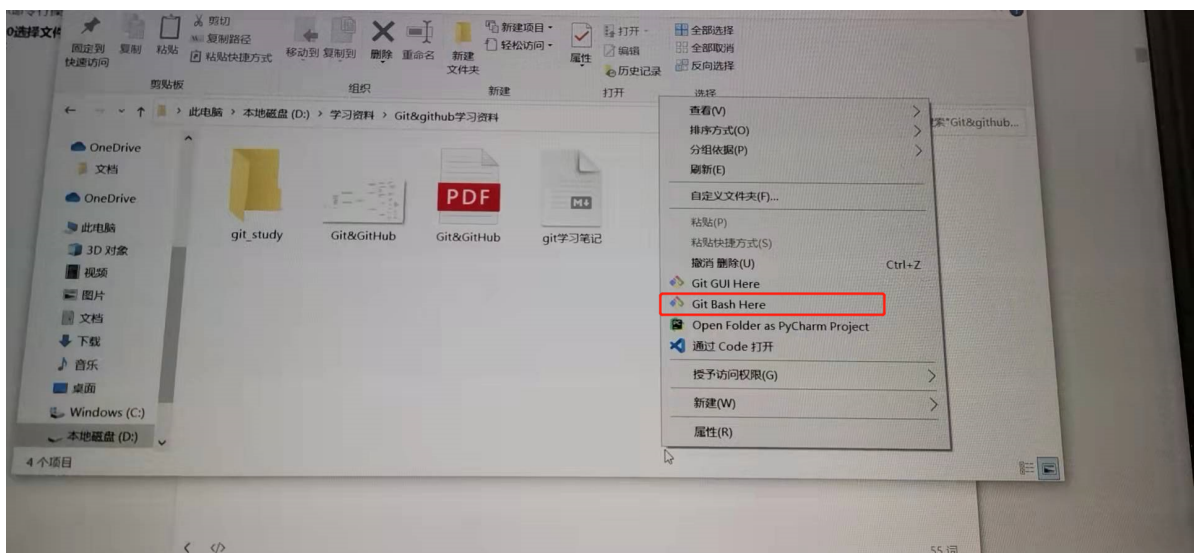


2.Git安装

3.Git命令行操作

3.0选择文件夹目录或新建一个文件夹

1.选择文件夹目录：前往指定的文件夹，点击右键，运行“Git Bash Here”指令。

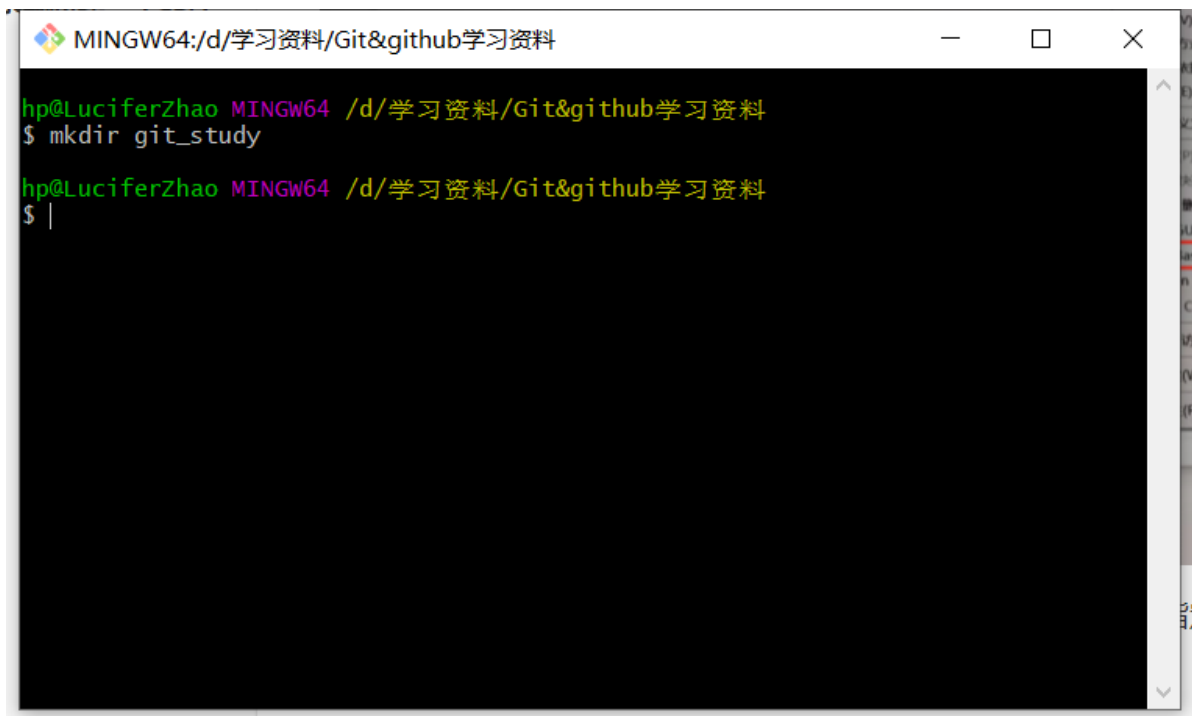


2.在任意处，点击右键，运行“Git Bash Here”指令。再前往指定的的目录。

3.新建文件夹

指令：mkdir 文件夹名字

效果：

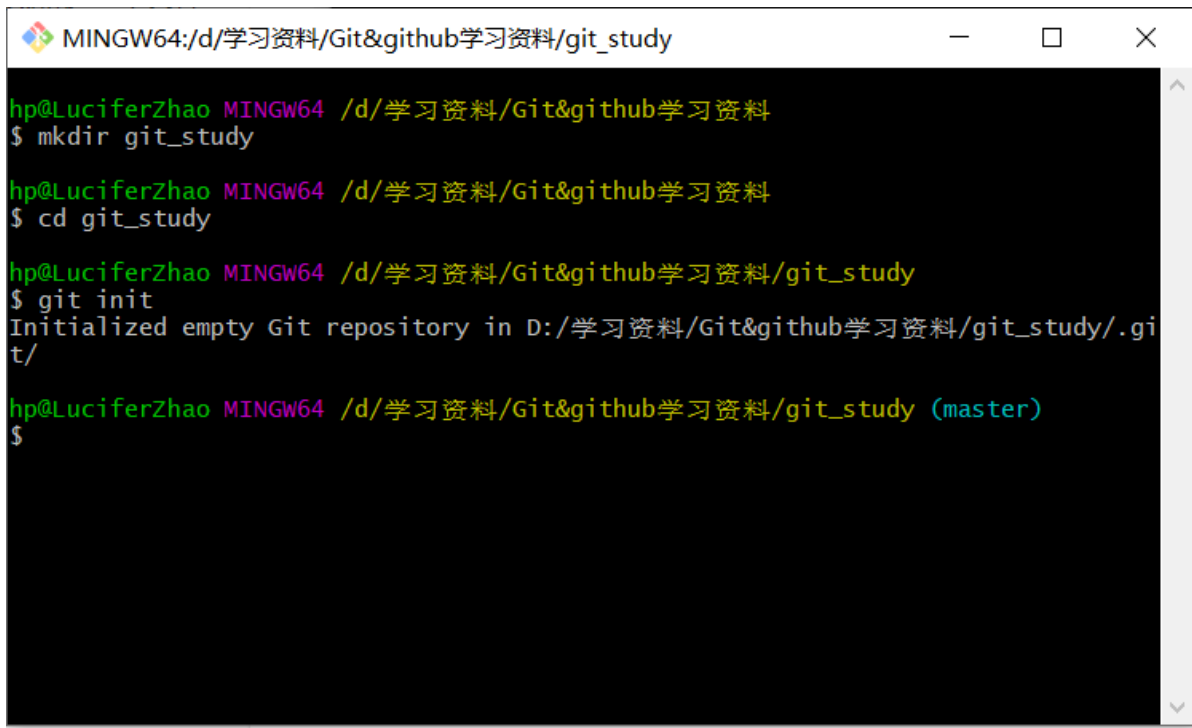


```
MINGW64:/d/学习资料/Git&github学习资料
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料
$ mkdir git_study
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料
$ |
```

3.1本地库初始化

指令：git init

效果：



```
MINGW64:/d/学习资料/Git&github学习资料/git_study
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料
$ mkdir git_study
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料
$ cd git_study
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study
$ git init
Initialized empty Git repository in D:/学习资料/Git&github学习资料/git_study/.git/
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$
```

注意：.git 目录中存放的是本地库相关的子目录和文件，不要删除，也不要胡乱修改。

3.2设置签名

作用：区分不同开发人员的身份。

辨析：这里设置的签名和登录远程库(代码托管中心)的账号、密码没有任何关系。

形式：

用户名：Lucifer

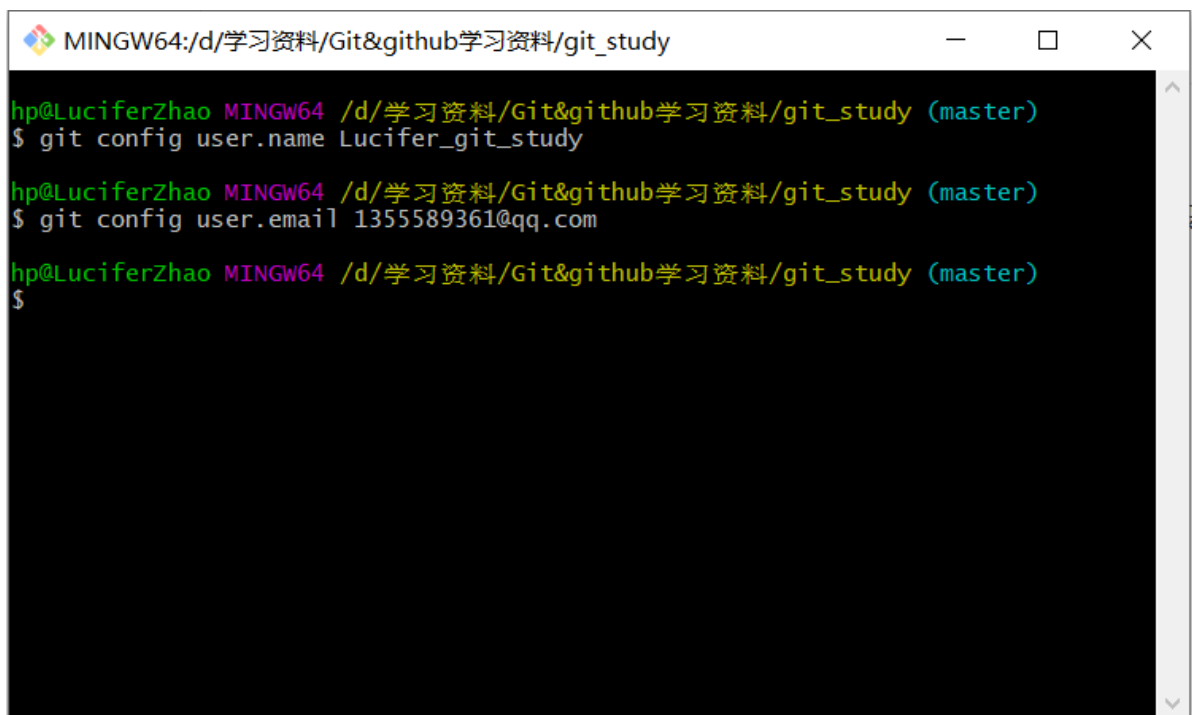
邮箱：1355589361@qq.com

命令：

项目级别/仓库级别：仅在当前本地库范围内有效

```
git config user.name Lucifer_git_study
```

```
git config user.email 1355589361@qq.com
```

A screenshot of a MINGW64 terminal window. The title bar shows the path 'MINGW64:/d/学习资料/Git&github学习资料/git_study'. The terminal content shows three lines of commands and their prompts: 1. 'hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)' followed by '\$ git config user.name Lucifer_git_study'. 2. 'hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)' followed by '\$ git config user.email 1355589361@qq.com'. 3. 'hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)' followed by '\$'.

```
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git config user.name Lucifer_git_study

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git config user.email 1355589361@qq.com

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$
```

信息保存位置：./.git/config 文件

```
MINGW64:/d/学习资料/Git&github学习资料/git_study
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git config user.name Lucifer_git_study
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git config user.email 1355589361@qq.com
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ cat .git/config
[core]
    repositoryformatversion = 0
    filemode = false
    bare = false
    logallrefupdates = true
    symlinks = false
    ignorecase = true
[user]
    name = Lucifer_git_study
    email = 1355589361@qq.com
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ |
```

系统用户级别：登录当前操作系统的用户范围

git config --global user.name Lucifer

git config --global user.email 1355589361@qq.com

```
MINGW64:/d/学习资料/Git&github学习资料/git_study
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git config --global user.name Lucifer
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git config --global user.email 1355589361@qq.com
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ |
```

信息保存位置：~/.gitconfig 文件

```
MINGW64:/c/Users/hp

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git config --global user.name Lucifer

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git config --global user.email 1355589361@qq.com

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ cd ~

hp@LuciferZhao MINGW64 ~
$ cat ~/.gitconfig
[user]
    name = Lucifer
    email = 1355589361@qq.com

hp@LuciferZhao MINGW64 ~
$
```

级别优先级：

就近原则：项目级别优先于系统用户级别，二者都有时采用项目级别的签名

如果只有系统用户级别的签名，就以系统用户级别的签名为准

二者都没有不允许

3.3基本操作

3.3.1状态查看

命令：git status

描述：可用于查看工作区、暂存区状态。

3.3.2添加至暂存区

命令：git add [file name]

描述：将工作区的“新建/修改”添加到暂存区。

3.3.3从暂存区撤销

命令：git rm --cached [file name]

描述：将文件从暂存区撤出，恢复至未添加至暂存区前的状态。

3.3.4提交至本地库

命令：git commit -m "[commit message]" [file name]

描述：将暂存区的内容提交到本地库。

3.3.5查看历史记录

命令: git log

```
MINGW64:/d/学习资料/Git&github学习资料/git_study
1 file changed, 1 insertion(+)

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git log
commit a8eb9c44edfacc7615d96c741bb21559711a7436 (HEAD -> master)
Author: Lucifer_git_study <1355589361@qq.com>
Date: Thu Dec 9 17:32:40 2021 +0800

    for test history

commit 51fd61e639c711b08ee1603d5d5b4b569e8e816d
Author: Lucifer_git_study <1355589361@qq.com>
Date: Thu Dec 9 17:13:16 2021 +0800

    My second commit,modify good.txt

commit 0ba8cb3d7bb75aa47fb8e51f13fd5ac6cb92374e
Author: Lucifer_git_study <1355589361@qq.com>
Date: Thu Dec 9 17:00:32 2021 +0800

    My first commit.new file good.txt
```

当历史记录特别特别的多时，屏幕无法一次之内显示全部的历史记录，此时控制屏幕翻页的操作为：

b: 向前翻页

空格: 向后翻页

q: 退出

或采用一下指令显示部分历史信息：

①只显示索引及提交信息: git log --pretty=oneline

```
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git log --pretty=oneline
a8eb9c44edfacc7615d96c741bb21559711a7436 (HEAD -> master) for test history
51fd61e639c711b08ee1603d5d5b4b569e8e816d My second commit,modify good.txt
0ba8cb3d7bb75aa47fb8e51f13fd5ac6cb92374e My first commit.new file good.txt
```

②只显示索引末尾几位及提交信息: git log --oneline

```
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git log --oneline
a8eb9c4 (HEAD -> master) for test history
51fd61e My second commit,modify good.txt
0ba8cb3 My first commit.new file good.txt
```

③只显示索引末尾几位、指针移动步数信息及提交信息: git reflog

```
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git reflog
a8eb9c4 (HEAD -> master) HEAD@{0}: commit: for test history
51fd61e HEAD@{1}: commit: My second commit,modify good.txt
0ba8cb3 HEAD@{2}: commit (initial): My first commit.new file good.txt
```

HEAD@{移动到当前版本需要多少步}

3.4版本前进后退

本质: 指针移动

HEAD →

```
fd83eb9 (HEAD -> master) HEAD@{0}: commit: insert qqqqqqqq edit
7bf0e31 HEAD@{1}: commit: insert pppppp edit
2679109 HEAD@{2}: commit: insert oooooo edit
9a9ebe0 HEAD@{3}: commit: insert nnnnnnnnn edit
49f1bd3 HEAD@{4}: commit: insert mmmmmmmmm edit
a6ace91 HEAD@{5}: commit: insert llllllll edit
3dd95d7 HEAD@{6}: commit: insert kkkkkkkkk edit
42e7e84 HEAD@{7}: commit: insert jjjjjjjj edit
7c265b1 HEAD@{8}: commit: insert iiiiii
c309b92 HEAD@{9}: commit: insert hhhhhh edit
7305cd8 HEAD@{10}: commit: insert ggggggggg edit
ede116d HEAD@{11}: commit: insert fffffff edit
6325c55 HEAD@{12}: commit: insert eeeeeee edit
a709ad9 HEAD@{13}: commit: for test history
bfb79b7 HEAD@{14}: commit: My second commit,modify good.txt
ac5c801 HEAD@{15}: commit (initial): My first commit.new file good.txt
```

①基于索引值操作（推荐）：git reset --hard [局部索引值]

```
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git reset --hard 51fd61e
HEAD is now at 51fd61e My second commit,modify good.txt

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git reflog
51fd61e (HEAD -> master) HEAD@{0}: reset: moving to 51fd61e
0ba8cb3 HEAD@{1}: reset: moving to 0ba8cb3
a8eb9c4 HEAD@{2}: commit: for test history
51fd61e (HEAD -> master) HEAD@{3}: commit: My second commit,modify good.txt
0ba8cb3 HEAD@{4}: commit (initial): My first commit.new file good.txt
```

②使用^符号（只能后退）：git reset --hard HEAD^

```
MINGW64:/d/学习资料/Git&github学习资料/git_study

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git log --oneline
51fd61e (HEAD -> master) My second commit,modify good.txt
0ba8cb3 My first commit.new file good.txt

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git reset --hard HEAD^
HEAD is now at 0ba8cb3 My first commit.new file good.txt

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git log --oneline
0ba8cb3 (HEAD -> master) My first commit.new file good.txt

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$
```

注：一个^表示后退一步，n个表示后退n步。

③使用~符号（只能后退）：git reset --hard HEAD~n

注：n表示后退n步。

3.3.6reset命令的三个参数对比

①--soft:

仅仅在本地库移动HEAD指针。



②--mixed:

在本地库移动HEAD指针。

重置缓存区。



③--hard

在本地库移动HEAD指针。

重置缓存区。

重置工作区。



3.3.7删除文件并找回

前提：删除前，文件存在时的状态提交到了本地库。

操作：git reset --hard [指针位置]

删除操作已经提交到本地库：指针位置指向历史记录。

```
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git reset --hard 0383942
HEAD is now at 0383942 new git_delete_study.txt
```

删除操作尚未提交到本地库：指针位置使用 HEAD。

```
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ rm git_delete_study.txt

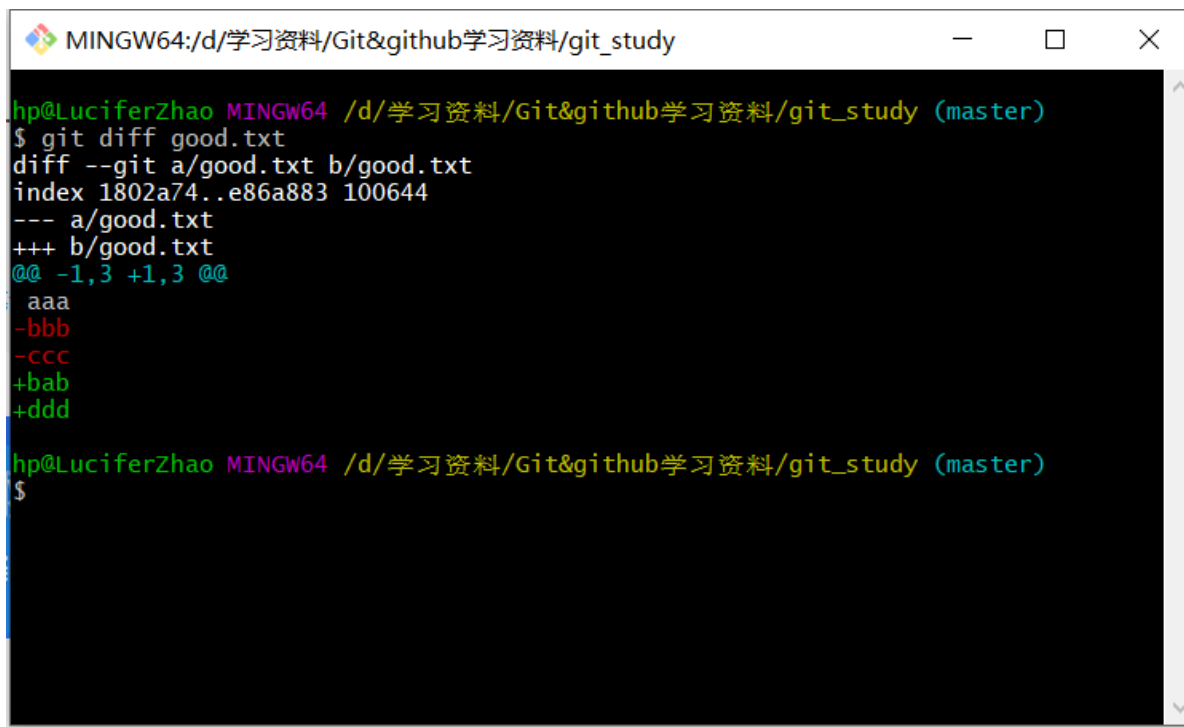
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git add git_delete_study.txt

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:    git_delete_study.txt

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git reset --hard HEAD
HEAD is now at 0383942 new git_delete_study.txt
```

3.3.8比较文件差异

将工作区中的文件和暂存区进行比较：git diff [文件名]



```
MINGW64:/d/学习资料/Git&github学习资料/git_study
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git diff good.txt
diff --git a/good.txt b/good.txt
index 1802a74..e86a883 100644
--- a/good.txt
+++ b/good.txt
@@ -1,3 +1,3 @@
   aaa
-  bbb
-  ccc
+  bab
+  ddd

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$
```

将工作区中的文件和本地库历史记录比较：git diff [本地库中历史版本] [文件名]

```

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git diff HEAD good.txt
diff --git a/good.txt b/good.txt
index e86a883..35fbd83 100644
--- a/good.txt
+++ b/good.txt
@@ -1,3 +1,4 @@
   aaa
-  bab
+  bbb
+  ccc
   ddd

```

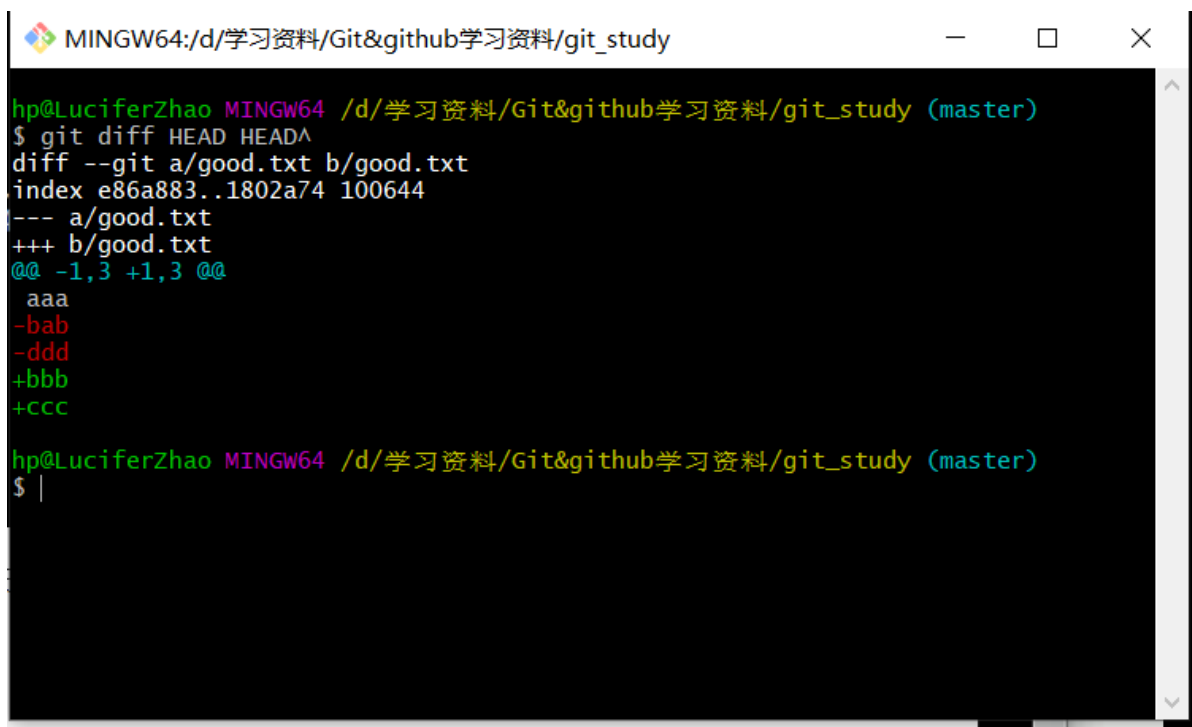
不带文件名比较多个文件: git diff

```

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git diff
diff --git a/git_delete_study.txt b/git_delete_study.txt
index 7cf6361..0b15cb8 100644
--- a/git_delete_study.txt
+++ b/git_delete_study.txt
@@ -3,4 +3,6 @@ am
   learning
   how
   to
-  delete
+  compare
+  file
+  difference
diff --git a/good.txt b/good.txt
index e86a883..35fbd83 100644
--- a/good.txt
+++ b/good.txt
@@ -1,3 +1,4 @@
   aaa
-  bab
+  bbb
+  ccc
   ddd

```

比较两个历史版本: git diff [本地库中历史版本] [本地库中历史版本]



```

MINGW64:/d/学习资料/Git&github学习资料/git_study
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git diff HEAD HEAD^
diff --git a/good.txt b/good.txt
index e86a883..1802a74 100644
--- a/good.txt
+++ b/good.txt
@@ -1,3 +1,3 @@
   aaa
-  bab
-  ddd
+  bbb
+  ccc

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ |

```

4.分支管理

4.1什么是分支

在版本控制过程中，使用多条线同时推进多个任务。

4.2分支的好处

同时并行推进多个功能开发，提高开发效率。

各个分支在开发过程中，如果某一个分支开发失败，不会对其他分支有任何影响。失败的分支删除重新开始即可。

4.3分支操作

创建分支

操作：git branch [分支名]

```
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git branch hot_fix
```

查看分支

操作：git branch -v

```
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git branch -v
hot_fix 0fef2df 基于多个文件比较的修改
* master 0fef2df 基于多个文件比较的修改
```

切换分支

操作：git checkout [分支名]

```
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git checkout hot_fix
Switched to branch 'hot_fix'
```

合并分支

第一步：切换到接受修改的分支（被合并，增加新内容）上。

操作：git checkout [被合并分支名]

第二步：执行 merge 命令

操作：git merge [有新内容分支名]

```
MINGW64:/d/学习资料/Git&github学习资料/git_study
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (hot_fix)
$ git branch -v
* hot_fix 5c6853c test branch hot_fix
  master 0fef2df 基于多个文件比较的修改

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (hot_fix)
$ git checkout master
Switched to branch 'master'

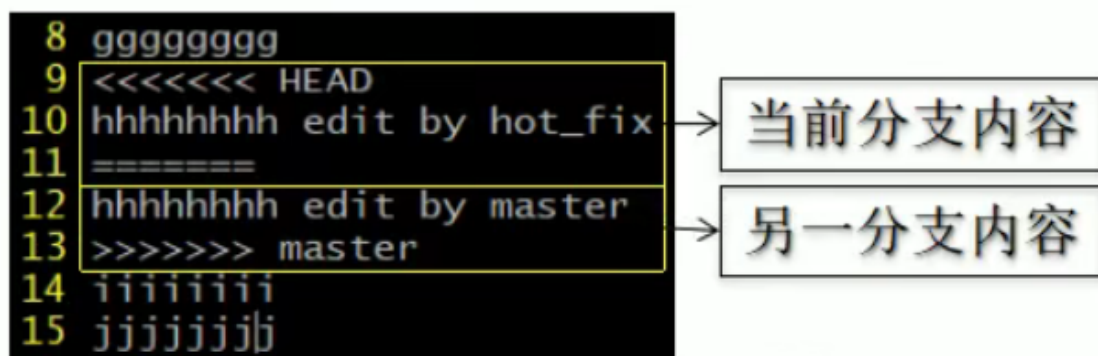
hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git merge hot_fix
Updating 0fef2df..5c6853c
Fast-forward
 good.txt | 1 +
 1 file changed, 1 insertion(+)

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$ git branch -v
  hot_fix 5c6853c test branch hot_fix
* master 5c6853c test branch hot_fix

hp@LuciferZhao MINGW64 /d/学习资料/Git&github学习资料/git_study (master)
$
```

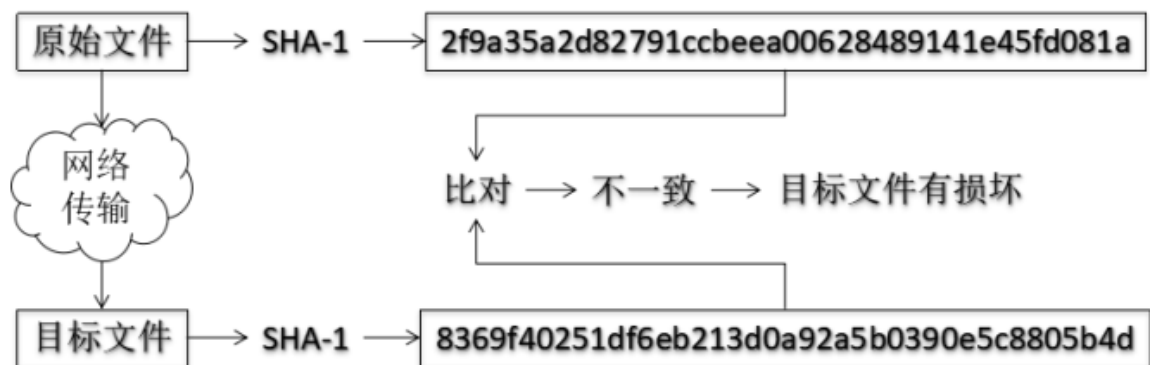
解决冲突

冲突的表现：



冲突的解决：

第一步：编辑文件，删除特殊符号。

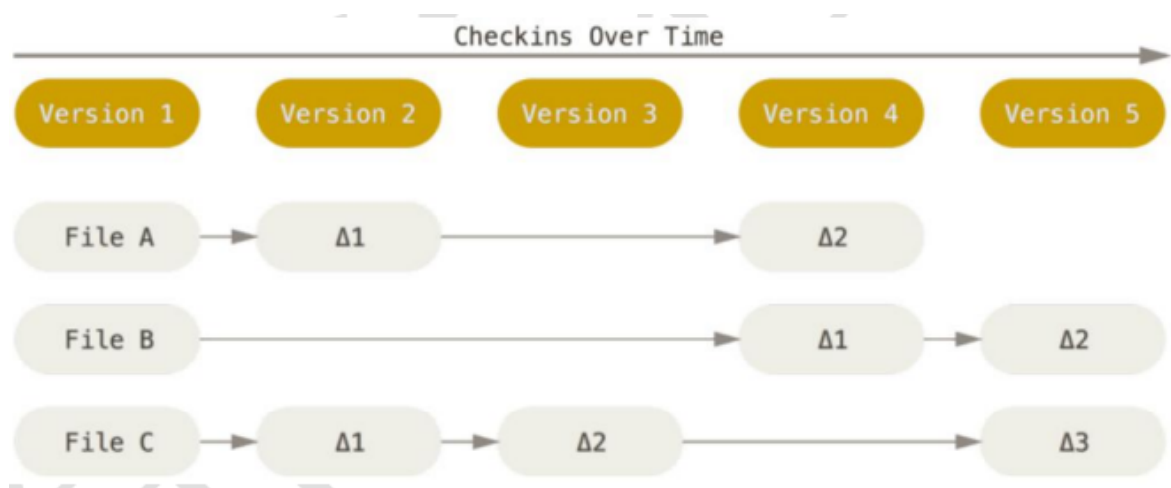


Git 就是靠这种机制来从根本上保证数据完整性的。

5.2Git版本管理机制

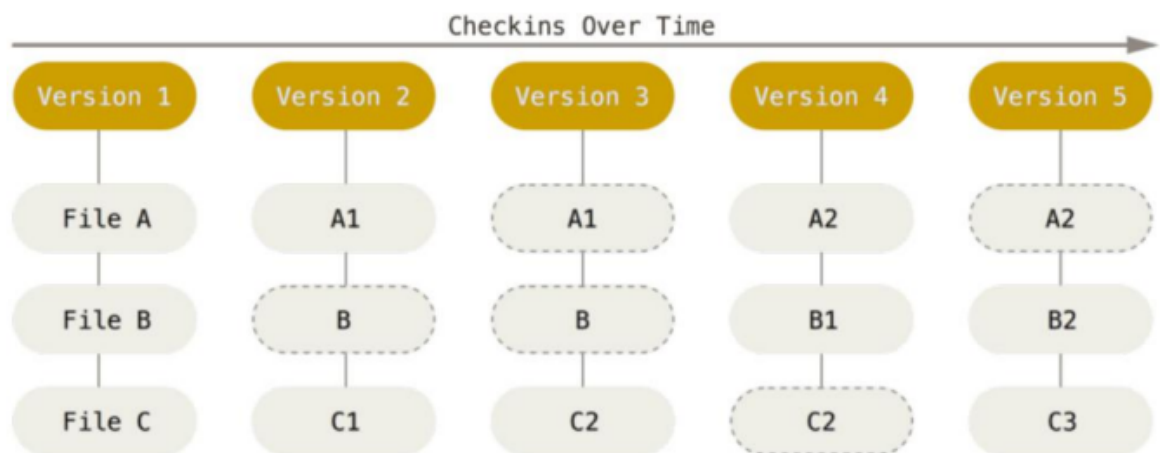
5.2.1集中式版本控制工具的文件管理机制

以文件变更列表的方式存储信息。这类系统将它们保存的信息看作是一组基本文件和每个文件随时间逐步累积的差异。



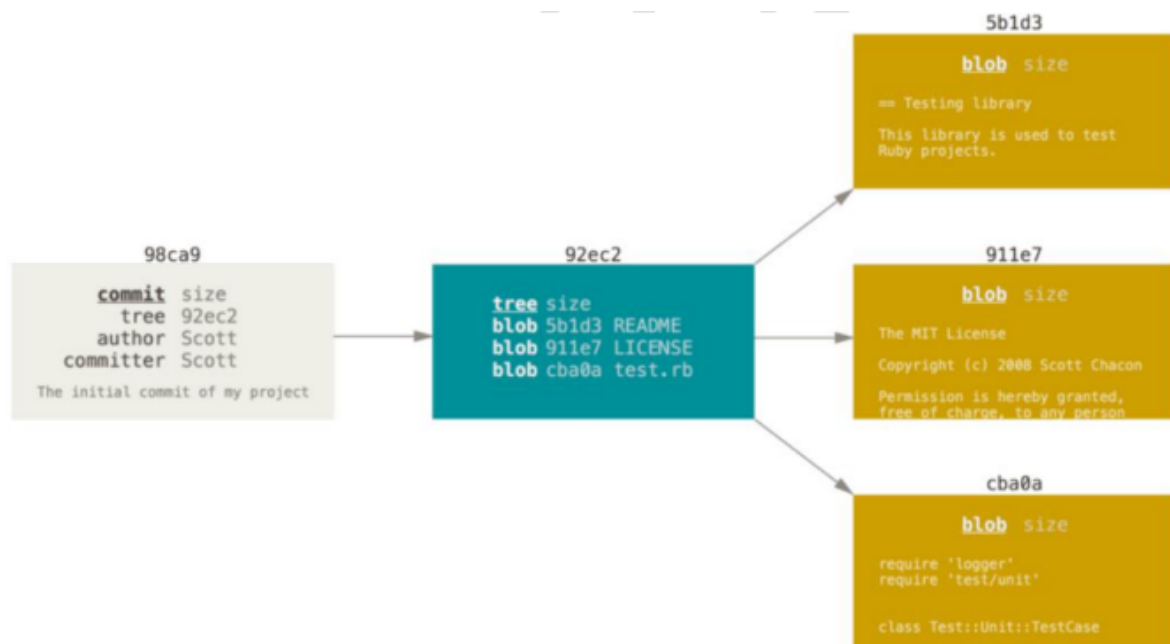
5.2.2Git 的文件管理机制

Git 把数据看作是小型文件系统的一组快照。每次提交更新时 Git 都会对当前的全部文件制作一个快照并保存这个快照的索引。为了高效，如果文件没有修改，Git 不再重新存储该文件，而是只保留一个链接指向之前存储的文件。所以 Git 的工作方式可以称之为快照流。

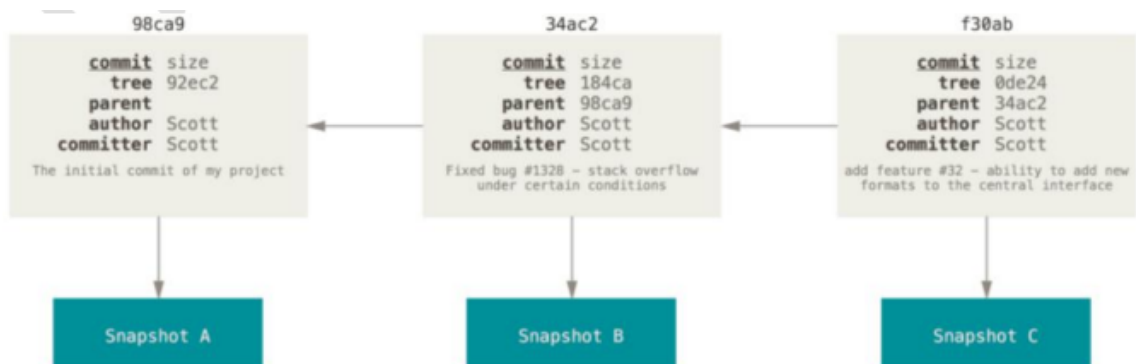


5.2.3Git 文件管理机制细节

Git 的“提交对象”：



提交对象及其父对象形成的链条：



5.3Git分支管理机制

5.3.1分支的创建

本质：新建一个指针，指向同一块地址。



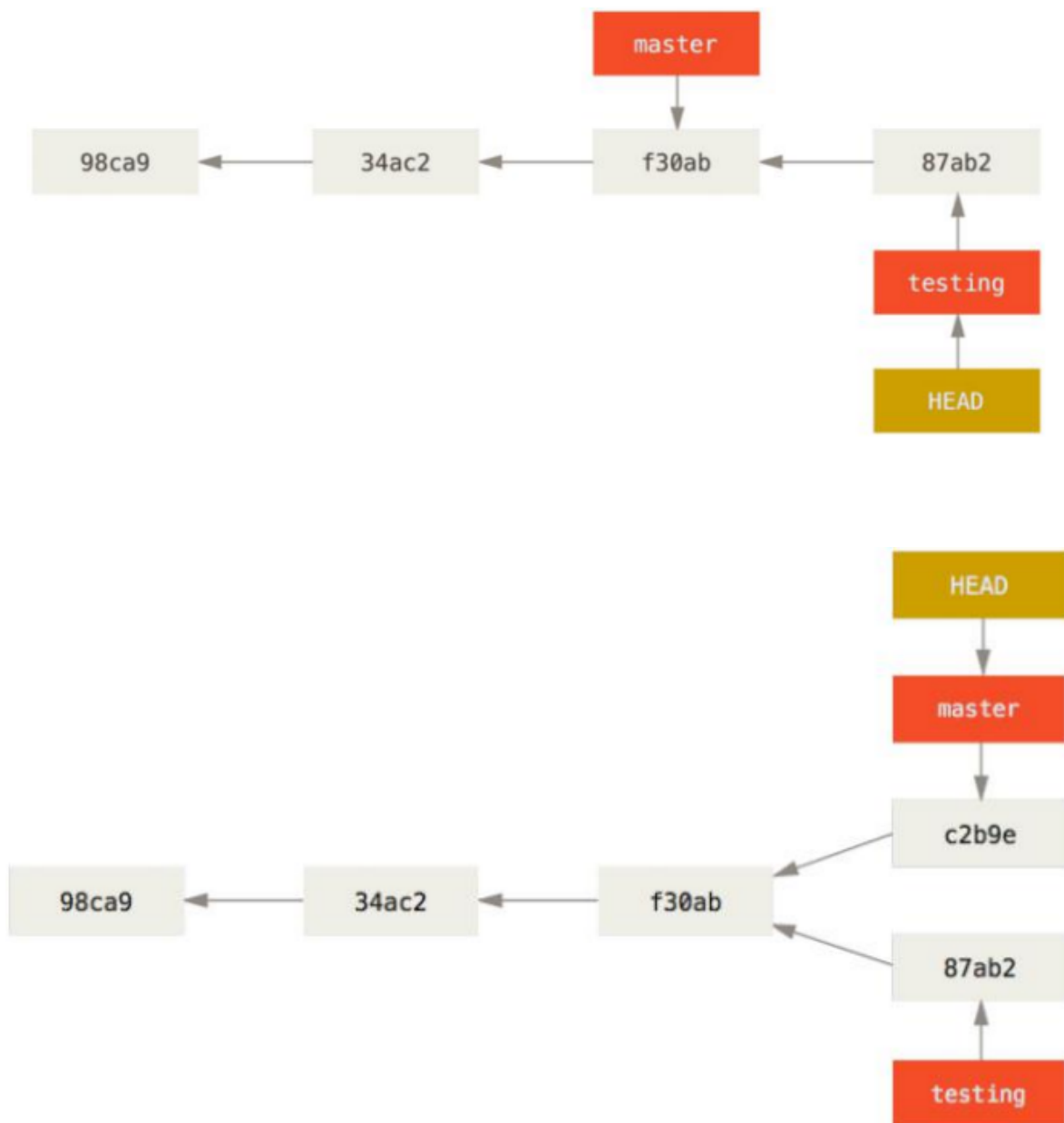
5.3.2分支的切换

本质：改变HEAD指针指向的地址。



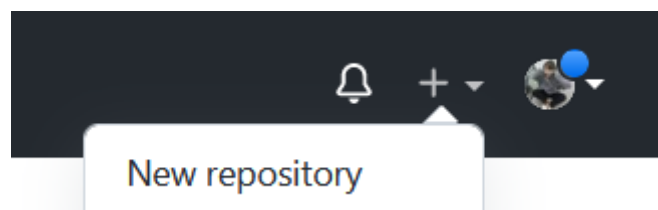
5.3.3分支的管理

本质：不同的分支指向不同的指针。




6. GitHub

6.1 创建远程库



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Owner *  LuciferZhao

Repository name * 远程库名字

Great repository names are short and memorable. Need inspiration? How about fuzzy-eureka?

Description (optional) 描述

☒ Public
Anyone on the internet can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository. 设置远程库公开或私有

Initialize this repository with:
Skip this step if you're importing an existing repository.

☐ Add a README file
This is where you can write a long description for your project. [Learn more.](#)

☐ Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

☐ Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository 创建远程库

6.2创建远程库地址别名

查看当前所有远程地址别名: `git remote -v`

```
MINGW64:/d/学习资料/尚硅谷/Git&github学习资料/GitHub_study
hp@LuciferZhao MINGW64 /d/学习资料/尚硅谷/Git&github学习资料/GitHub_study (master)
$ git remote -v

hp@LuciferZhao MINGW64 /d/学习资料/尚硅谷/Git&github学习资料/GitHub_study (master)
$ git remote add github_study https://github.com/LuciferZhao/github_study.git

hp@LuciferZhao MINGW64 /d/学习资料/尚硅谷/Git&github学习资料/GitHub_study (master)
$ git remote -v
github_study    https://github.com/LuciferZhao/github_study.git (fetch)
github_study    https://github.com/LuciferZhao/github_study.git (push)

hp@LuciferZhao MINGW64 /d/学习资料/尚硅谷/Git&github学习资料/GitHub_study (master)
$
```

设置某个远程库地址的别名: `git remote add [别名] [远程地址]`

```
hp@LuciferZhao MINGW64 /d/学习资料/尚硅谷/Git&github学习资料/GitHub_study (master)
$ git remote add github_study https://github.com/LuciferZhao/github_study.git
```

6.3推送

操作: `git push -u [别名/远程地址] [分支名]`

```

hp@LuciferZhao MINGW64 /d/学习资料/尚硅谷/Git&github学习资料/GitHub_study (master)
$ git push -u https://gitee.com/LuciferZhao/git_study.git master
warning: ----- SECURITY WARNING -----
warning: | TLS certificate verification has been disabled! |
warning: -----
warning: HTTPS connections may not be secure. See https://aka.ms/gcmcore-tlsverify for more information.
warning: ----- SECURITY WARNING -----
warning: | TLS certificate verification has been disabled! |
warning: -----
warning: HTTPS connections may not be secure. See https://aka.ms/gcmcore-tlsverify for more information.
warning: ----- SECURITY WARNING -----
warning: | TLS certificate verification has been disabled! |
warning: -----
warning: HTTPS connections may not be secure. See https://aka.ms/gcmcore-tlsverify for more information.
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 258 bytes | 258.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote: Powered by GITEE.COM [GNK-6.2]
To https://gitee.com/LuciferZhao/git_study.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'https://gitee.com/LuciferZhao/git_study.git'.

```

6.4克隆

操作：git clone [远程地址]

```

hp@LuciferZhao MINGW64 /d/学习资料/尚硅谷/Git&github学习资料/GitHub_clone
$ git clone https://gitee.com/LuciferZhao/git_study.git
Cloning into 'git_study'...
warning: ----- SECURITY WARNING -----
warning: | TLS certificate verification has been disabled! |
warning: -----
warning: HTTPS connections may not be secure. See https://aka.ms/gcmcore-tlsverify for more information.
warning: ----- SECURITY WARNING -----
warning: | TLS certificate verification has been disabled! |
warning: -----
warning: HTTPS connections may not be secure. See https://aka.ms/gcmcore-tlsverify for more information.
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.

```

效果：

- ①完整的把远程库下载到本地。
- ②创建远程地址别名。
- ③初始化本地库。

6.5拉取

操作：git pull [远程库地址别名] [远程分支名]

pull=fetch+merge

git fetch [远程库地址别名] [远程分支名]

git merge [远程库地址别名/远程分支名]

```
hp@LuciferZhao MINGW64 /d/学习资料/尚硅谷/Git&github学习资料/GitHub_study (master)
$ git fetch github_study master
From github.com:LuciferZhao/github_study
* branch          master       -> FETCH_HEAD

hp@LuciferZhao MINGW64 /d/学习资料/尚硅谷/Git&github学习资料/GitHub_study (master)
$ git merge github_study/master
Updating 81a3a49..eeb38d3
Fast-forward
 readme.txt | 1 +
 1 file changed, 1 insertion(+)
```

6.6解决团队协作冲突

如果不是基于 GitHub 远程库的最新版所做的修改，不能推送，必须先拉取。

拉取下来后如果进入冲突状态，则按照“分支冲突解决”操作解决即可。

6.7SSH免密登录

1.进入当前用户的家目录

命令：\$ cd ~

2.删除.ssh目录

命令：\$ rm -rvf.ssh

3.运行命令生成.ssh密钥目录

命令：\$ ssh-keygen -t rsa -C [登录邮箱]

(注意：这里-C 这个参数是大写的 C)

4.进入.ssh目录查看文件列表

命令：\$ cd .ssh

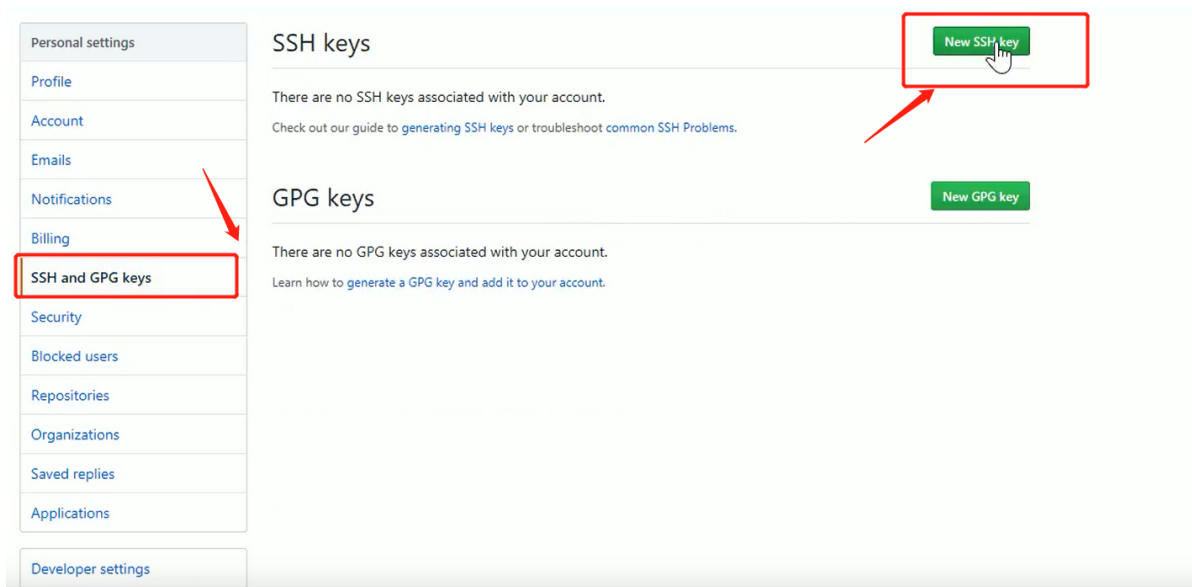
命令：\$ ls -lF

5.查看 id_rsa.pub 文件内容

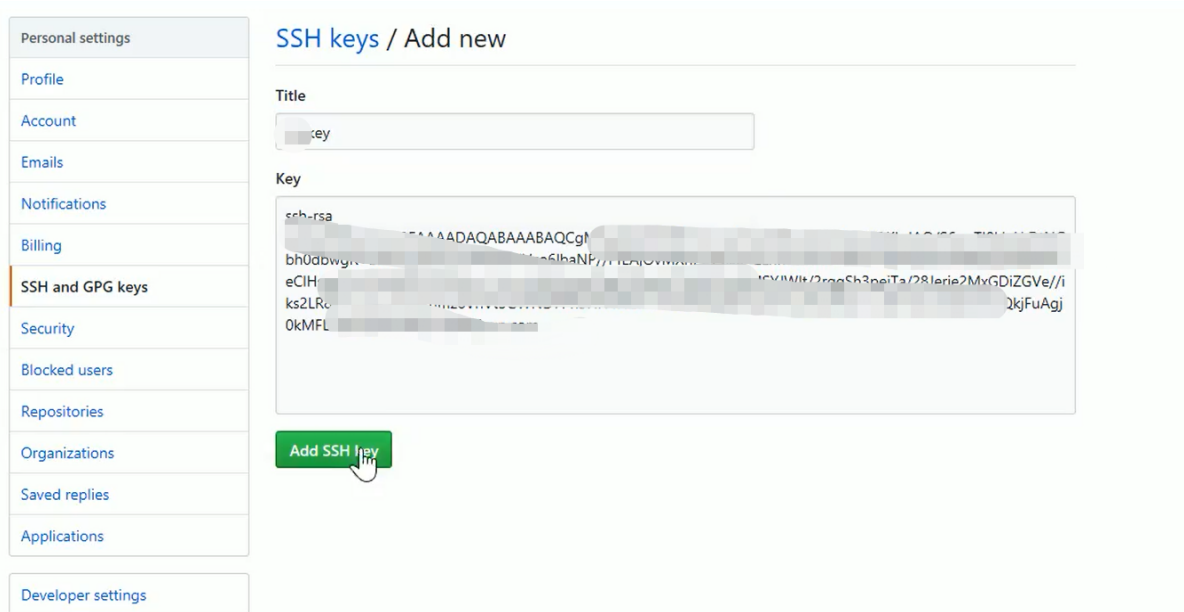
命令：\$ cat id_rsa.pub

6.复制 id_rsa.pub 文件内容，登录 GitHub，点击用户头像→Settings→SSH and GPG keys

7.点击New SSH Key



8. 输入复制的密钥信息



9. 回到 Git bash 创建远程地址别名

命令: `git remote add [别名] [SSH链接]`

10. 推送文件进行测试