

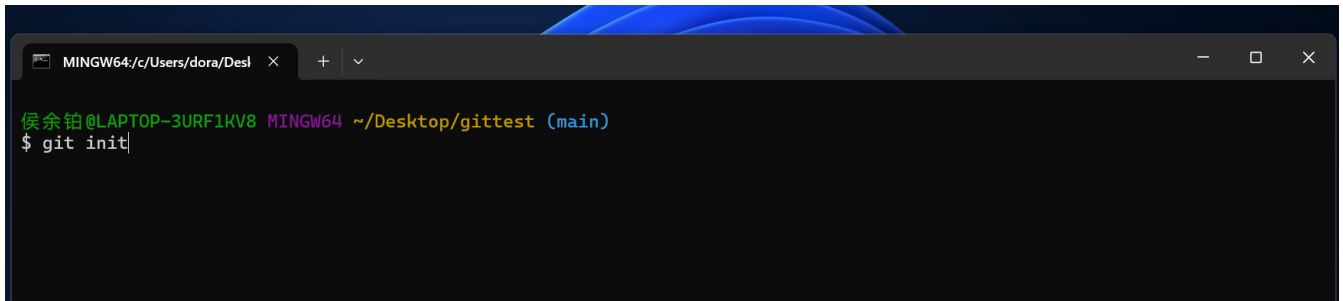
系统开发工具第一次实验报告

24020007044 侯余铂

2025 年 9 月 5 日

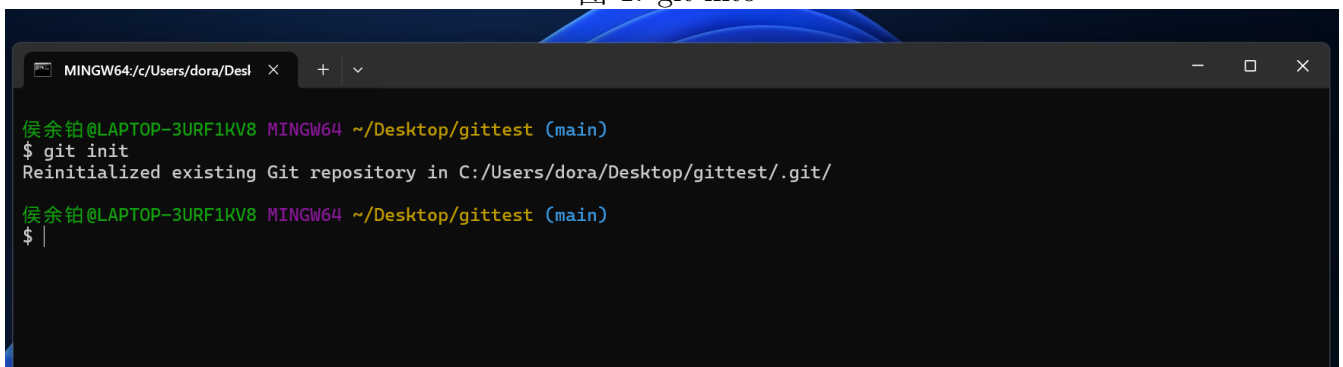
以下是 20 个关于 git 的命令的总结:

1. git init: 初始化一个新的 Git 仓库, 在本地上创建一个仓库



```
MINGW64/c/Users/dora/Desl x + v
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git init
```

图 1: git into



```
MINGW64/c/Users/dora/Desl x + v
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git init
Reinitialized existing Git repository in C:/Users/dora/Desktop/gittest/.git/
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ |
```

图 2: git init -1

2. git config --global user.name: 创建用户的姓名

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git config --global user.name|
```

图 3: user.name

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git config --global user.name
dorasweater
```

图 4: user.name -1

3. git config --global user.email: 用来创建用户的邮箱

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git config --global user.email|
```

图 5: user.email

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git config --global user.email
2759685528@qq.com
```

图 6: user.email -1

4. touch file1.tet: 用来创建一个新的文件 file.tet

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ touch file1.tet
```

图 7: touch file

名称	修改日期	类型
📁 .git	2025/9/3 22:41	文件夹
📄 file1.tet	2025/9/3 22:51	TET 文件

图 8: touch file1 -1

5. git status: 用于查看缓存是否有未添加到暂存区的文件

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git status
```

图 9: git status

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file1.tet

nothing added to commit but untracked files present (use "git add" to track)
```

图 10: git status -1

6. git add .: 将其所有文件添加到暂存区

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git add .|
```

图 11: git add .

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git add .

侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file1.tet
```

图 12: git add . -1

7. git commit -m "add file1": 将暂存区的文件提交到仓库中，并加以备注，例如现在的 add file1

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git commit -m "add file1"
```

图 13: git commit

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git commit -m "add file1"
[main (root-commit) b074f63] add file1
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 file1.tet
```

图 14: git commit -1

8. git log: 查看日志的，用于看之前提交的记录

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git log
```

图 15: git log

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git log
commit b074f638ef1fdf39fc9d26a74a88dca2113e9736 (HEAD -> main)
Author: dorasweater <2759685528@qq.com>
Date:   Wed Sep 3 22:57:18 2025 +0800

    add file1
```

图 16: git log -1

9. vi file1.tet: 打开文件 file1.tet 来编写，然后再保存

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ vi file1.txt
```

图 17: vi file1.tet

```
#include"FinancialRecord.h"
class FinancialManger
{
private:
    std::vector<FinancialRecord*> records;
    double balance;
    double targetBalance;
public:
    FinancialManger():balance(0),targetBalance(200000){}
    ~FinancialManger()
    {
        for (auto record : records)
        {
            delete record;
        }
    }
};
```

图 18: vi file1.tet -1

10. `git log --pretty=oneline --all --graph --abbrev-commit`: 用于查看简化的日志, 让排版更清晰更方便我们检查

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git log --pretty=oneline --all --graph --abbrev-commit
```

图 19: git-log

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git log --pretty=oneline --all --graph --abbrev-commit
* b074f63 (HEAD -> main) add file1
```

图 20: git-log -1

11. `git reset --hard b074f63`: 回退到之前这个版本去

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git reset --hard b074f63|
```

图 21: git reset

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git reset --hard b074f63
HEAD is now at b074f63 add file1
```

图 22: git reset -1

12. git reflog: 用于查看回退和 tiaozuan

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git log|
```

图 23: git log

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git log
commit b074f638ef1fdf39fc9d26a74a88dca2113e9736 (HEAD -> main)
Author: dorasweater <2759685528@qq.com>
Date:   Wed Sep 3 22:57:18 2025 +0800

    add file1
```

图 24: git log -1

13. vi file1.tet: 打开文件 file1.tet 来编写，然后再保存

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ vi file1.txt
```

图 25: vi file1.tet

```
#include"FinancialRecord.h"
class FinancialManger
{
private:
    std::vector<FinancialRecord*> records;
    double balance;
    double targetBalance;
public:
    FinancialManger():balance(0),targetBalance(200000){}
    ~FinancialManger()
    {
        for (auto record : records)
        {
            delete record;
        }
    }
};
```

图 26: vi file1.tet -1

14. `git log --pretty=oneline --all --graph --abbrev-commit`: 用于查看简化的日志, 让排版更清晰更方便我们检查

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git log --pretty=oneline --all --graph --abbrev-commit
```

图 27: git-log

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git log --pretty=oneline --all --graph --abbrev-commit
* b074f63 (HEAD -> main) add file1
```

图 28: git-log -1

15. `git reset --hard b074f63`: 回退到之前这个版本去

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git reset --hard b074f63|
```

图 29: git reset

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git reset --hard b074f63
HEAD is now at b074f63 add file1
```

图 30: git reset -1

16. git reflog: 用于查看回退和删除既用于查看本地仓库引用变更历史的命令，它记录了几几乎所有对仓库的操作

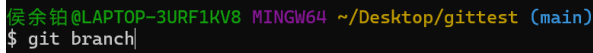
```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git reflog|
```

图 31: git reflog

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git reflog
b074f63 (HEAD -> main) HEAD@{0}: reset: moving to b074f63
b074f63 (HEAD -> main) HEAD@{1}: reset: moving to b074f63
b074f63 (HEAD -> main) HEAD@{2}: commit (initial): add file1
```

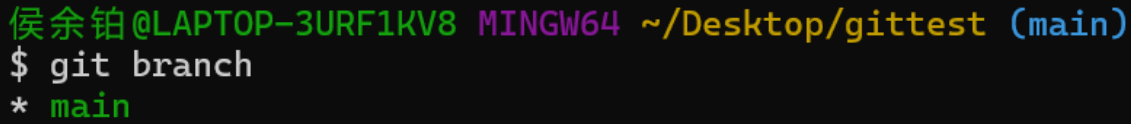
图 32: git reflog -1

17. git branch: 用于查看分支的指令



```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git branch
```

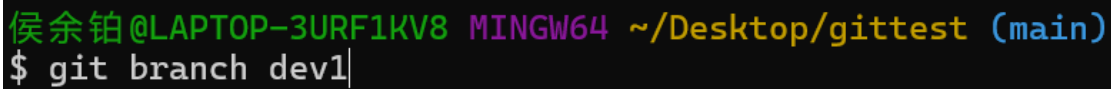
图 33: git branch



```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git branch
* main
```

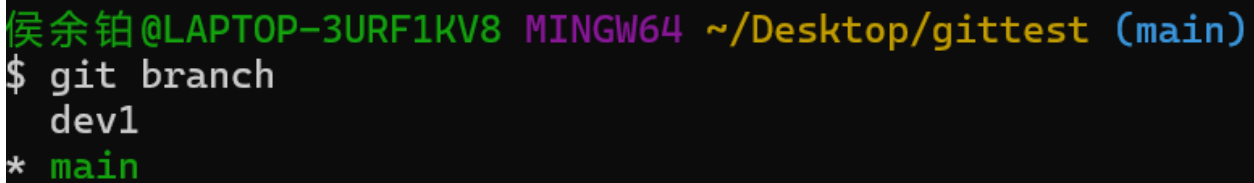
图 34: git branch -l

18. git branch dev1: 用于重新建一个分支 dev1



```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git branch dev1
```

图 35: git branch



```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git branch
dev1
* main
```

图 36: git branch -l

19. git checkout dev1: 用于跳转到 dev1 分支上去的一个指令

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git checkout dev1|
```

图 37: git checkout dev1

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git checkout dev1
Switched to branch 'dev1'

侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (dev1)
$ |
```

图 38: git checkout dev1 -1

20. git checkout -b dev02: 用于原来没有 dev02 时，在跳转到 dev02 分支时，同时新建一个 dev02 分支

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git checkout -b dev02|
```

图 39: git checkout -b

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git checkout -b dev02
Switched to a new branch 'dev02'

侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (dev02)
$ |
```

图 40: git checkout -b -1

21. git merge dev1: 用于合并分支，将 dev1 合并到 main 中

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git merge dev1
```

图 41: git merge

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git merge dev1
Already up to date.

侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ |
```

图 42: git merge -1

22. git branch -d dev02: 将 dev02 分支删除掉

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git branch -d dev02|
```

图 43: branch -d

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git branch -d dev02
Deleted branch dev02 (was b074f63).
```

图 44: branch -d -1

23. git remote add origin: 添加一个远程仓库 origin

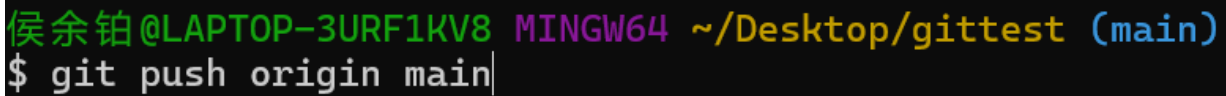
```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git remote add origin git@github.com:Dora-pt/git_study.git
```

图 45: git remote add origin

```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git remote
origin
|
```

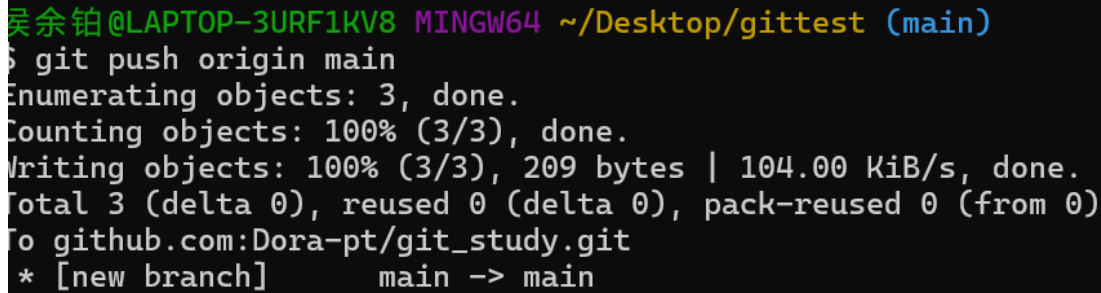
图 46: git remote add origin -1

24. git push origin main: 将本地分支 main 推送到远端仓库 origin 中去



```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git push origin main|
```

图 47: git push



```
侯余铂@LAPTOP-3URF1KV8 MINGW64 ~/Desktop/gittest (main)
$ git push origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 209 bytes | 104.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:Dora-pt/git_study.git
 * [new branch]      main -> main
```

图 48: git push -1

通过本次的学习让我收获了很多的东西，让我明白了 git 的基本使用，一些基本的语句，同时还学会了本地仓库如何与远端仓库进行文件的传送，如何与 github 建立联系，同时还让我学习了关于 latex 的相关语法，如何引用一些宏包，如何使用它的语法等等

GitHub:git@github.com:Dora-pt/git_study.git