

git 练习

蔺春名

2024 年 8 月 26 日

目录

1	四题	3
1.1	2	3
1.1.1	将版本历史可视化并进行探索	3
1.2	是谁最后修改了 README.md 文件?	3
1.2.1	最后一次修改 _config.yml 文件中 collections: 行时的提交信息是什么?	4
1.3	4	4
1.3.1	GitHub 上克隆某个仓库, 修改一些文件。当使用 git stash	5
1.3.2	当执行 git log -all --oneline 时	5
1.3.3	通过 git stash pop 命令来撤销 git stash 操作, 什么时候会用到这一技巧?	5
1.4	6	6
1.5	5	6
2	20 个练习	6
2.1	init missing-semester-cn	6
2.2	设置 remote	7
2.3	下载并使用 github cli(gh)	7
2.3.1	登录	7
2.4	fork 仓库	7
2.5	fork 之后设置为 fork 的仓库	7

目录	2
----	---

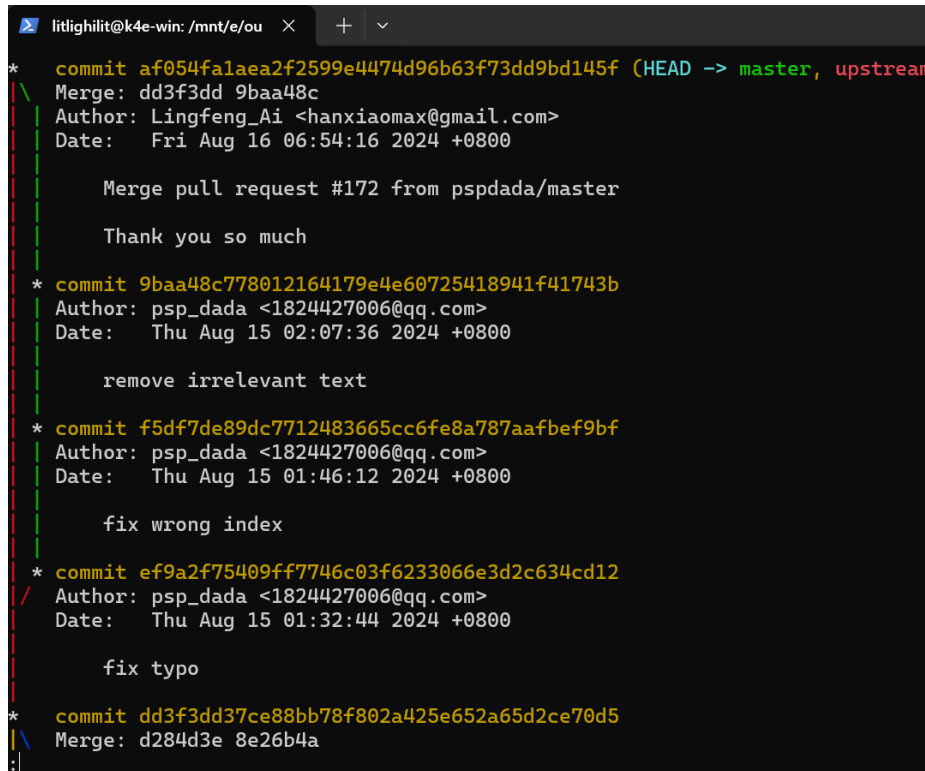
2.6 查看此时 remote	7
2.7 pull	8
2.8 增加 submodule	8
2.9 添加 solution 为 submodule	8
2.10 重置文件内容	8
2.10.1 修改文件	8
2.10.2 恢复	9
2.11 基于当前本地修改新建分支	9
2.12 提交 commit	9
2.13 pick 新分支的修改	9
2.14 全局设定默认编辑器	9
2.15 新建 github 仓库	9
2.16 上传	10
2.17 获得仓库信息	10
2.18 通过 branch+merge 来新加特性	10
2.19 添加 commit, 从文件中读取内容	10
2.20 使用 mergetool 处理冲突	10

1 四题

1.1 2

1.1.1 将版本历史可视化并进行探索

```
git log --graph
```



```
litlighthit@k4e-win: /mnt/e/ou × + v
* commit af054fa1aea2f2599e4474d96b63f73dd9bd145f (HEAD -> master, upstream
Merge: dd3f3dd 9baa48c
Author: Lingfeng_Ai <hanxiaomax@gmail.com>
Date: Fri Aug 16 06:54:16 2024 +0800

    Merge pull request #172 from pspdada/master

    Thank you so much
* commit 9baa48c778012164179e4e60725418941f41743b
Author: psp_dada <1824427006@qq.com>
Date: Thu Aug 15 02:07:36 2024 +0800

    remove irrelevant text
* commit f5df7de89dc7712483665cc6fe8a787aafbef9bf
Author: psp_dada <1824427006@qq.com>
Date: Thu Aug 15 01:46:12 2024 +0800

    fix wrong index
* commit ef9a2f75409ff7746c03f6233066e3d2c634cd12
Author: psp_dada <1824427006@qq.com>
Date: Thu Aug 15 01:32:44 2024 +0800

    fix typo
* commit dd3f3dd37ce88bb78f802a425e652a65d2ce70d5
Merge: d284d3e 8e26b4a
```

图 1: logGraph

1.2 是谁最后修改了 README.md 文件?

```
git log -1 --format='%an' -- README.md
或者使用 '--format="%ae"' 得到 email
```

ans: yuzq

ans:

yuzq@sunwayworld.com

1.2.1 最后一次修改 __config.yml 文件中 collections: 行时的提交信息是什么?

Step-Code 1: ../../step/get_msg_of_changed_line.sh

```
dir="missing-semester-cn.github.io"  ## cloned repo
path
basefile=__config.yml
file="$dir/$basefile"
# get line number
n='grep -n '^collections:$' "$file" | cut -d : -f 1'
echo $n

function getLastChangeMsg() {
    set -e
    loca dir="$1" basefile="$2" n="$3" commithash
    pushd "$dir"
    commithash='git blame -L $n "$basefile" | cut -d '
        ' -f 1'
    # f wanna se the whole commit info
    # git show "$commithash"
    git log -1 --format='%s' # only output commit
        message
    popd
}
getLastChangeMsg "$dir" "$basefile" "$n"
```

ans: "Redo
lectures as a
collection"

1.3 4.

从 GitHub 上克隆某个仓库，修改一些文件。当您使用 git stash 会发生什么？当您执行 git log -all -oneline 时会显示什么？通过 git stash pop 命令来撤销 git stash 操作，什么时候会用到这一技巧？

1.3.1 GitHub 上克隆某个仓库，修改一些文件。当使用 git stash

会将修改存储到 stash 栈中，使工作区回到最后一次提交的状态

```
bash: git: command not found
litlighthilit@k4e-win:/mnt/e/ouc/homework/cs_utils/git$ git status
On branch main
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   repr/repr.md

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

no changes added to commit (use "git add" and/or "git commit -a")
litlighthilit@k4e-win:/mnt/e/ouc/homework/cs_utils/git$ git stash
Saved working directory and index state WIP on main: 71cc77a submodule: missing-semester-cn.github.io
```

图 2: stash

1.3.2 当执行 git log -all -oneline 时

- git log 用于显示 commit 信息
- -all 选项会显示所有分支的提交历史，而不仅仅是当前分支。
- -oneline 选项会使每个提交只显示一行，通常包括提交的哈希值和提交信息。

```
litlighthilit@k4e-win:/mnt/e/ouc/homework/cs_utils/git$ git log --oneline --all
f359f9c (refs/stash) WIP on main: 71cc77a submodule: missing-semester-cn.github.io
9df4353 index on main: 71cc77a submodule: missing-semester-cn.github.io
71cc77a (HEAD -> main) submodule: missing-semester-cn.github.io
8fcd7a5 repr(git)
aa67b84 impr: fixup by wrap as a function 'getLastChangeMsg'
67e5cad step(git): get msg for last change for one line
4d36340 step(git): global .gitignore
25a409a step(git): alias for graph
```

图 3: logAll

1.3.3 通过 git stash pop 命令来撤销 git stash 操作，什么时候会用到这一技巧？

通过 git stash pop 命令来撤销 git stash 操作，什么时候会用到这一技巧？

1.4 6.

您可以通过执行

```
git config --global core.excludesfile ~/.gitignore_global
```

在 `~/.gitignore_global` 中创建全局忽略规则。配置您的全局 `gitignore` 文件来自动忽略系统或编辑器的临时文件，例如 `.DS_Store`

Step-Code 2: `../step/global_gitignore.sh`

```
git config --global core.excludesfile ~/.  
gitignore_global  
echo '.DS_Store' > ~/.gitignore_global
```

1.5 5.

请在 `/.gitconfig` 中创建一个别名，使您在运行 `git graph` 时，您可以得到 `git log -all -graph -decorate -oneline` 的输出结果；

Step-Code 3: `../step/alias.sh`

```
git config --global alias.graph 'log --all --graph --  
decorate --oneline'
```

<https://github.com/lit-notes/it-notes>

2 20 个练习

2.1 init missing-semester-cn

```
mkdir missing-semester-cn  
cd missing-semester-cn  
git init -b main
```

2.2 设置 remote

```
git remote add origin https://github.com/missing-semester-cn/missing-semester-cn.github.io.git
```

2.3 下载并使用 github cli(gh)

项目仓库: <https://github.com/cli/cli>

Windows 下解压其 bin/gh.exe 到 PATH 路径下即可

2.3.1 登录

gh auth **login** 按提示登录即可

按个人常用认证选项:

```
gh auth login --git-protocol=https --web
```

2.4 fork 仓库

```
gh repo fork https://github.com/missing-semester-cn/missing-semester-cn.github.io.git
```

2.5 fork 之后设置为 fork 的仓库

```
git remote add upstream https://github.com/missing-semester-cn/missing-semester-cn.github.io.git
git remote set-url git@github.com:lit-notes/missing-semester-cn.github.io.git
```

2.6 查看此时 remote

```
git remote -v
```

```
origin https://github.com/lit-notes/missing-semester-cn.github.io.git (fetch)
```

```
origin https://github.com/lit-notes/missing-semester-cn.github.io.git (push)
```

```
upstream https://github.com/missing-semester-cn/missing-semester-cn.github.io.git (fetch)
```

```
upstream https://github.com/missing-semester-cn/missing-semester-cn.github.io.git (push)
```

2.7 pull

```
git pull
```

```
remote: Enumerating objects: 133, done.
remote: Counting objects: 100% (124/124), done.
remote: Compressing objects: 100% (47/47), done.
remote: Total 89 (delta 60), reused 66 (delta 42), pack-reused 0 (from 0)
Unpacking objects: 100% (89/89), 30.93 KiB | 52.00 KiB/s, done.
```

2.8 增加 submodule

```
cd ..
git submodule add https://github.com/missing-
semester-cn/missing-semester-cn.github.io.git
./missing-semester-cn.github.io
```

2.9 添加 solution 为 submodule

```
git submodule add https://github.com/missing-semester-cn/missing-
-notes-and-solutions missing-notes-and-solutions

Cloning into 'xxx/homework/cs_utils/missing-notes-and-solutions'...
remote: Enumerating objects: 330, done.
remote: Counting objects: 100% (41/41), done.
remote: Compressing objects: 100% (25/25), done.
remote: Total 330 (delta 14), reused 30 (delta 10), pack-reused 289 (from 1)
Receiving objects: 100% (330/330), 14.94 MiB | 5.51 MiB/s, done.
Resolving deltas: 100% (122/122), done.
```

2.10 重置文件内容

2.10.1 修改文件

```
cd git/missing-semester-cn.github.io
echo something >> _config.yml
```


2.10.2 恢复

```
git checkout HEAD -- _config.yml
```

2.11 基于当前本地修改新建分支

```
git stash  
git checkout -b temp  
git stash apply
```

2.12 提交 commit

```
git add .  
git commit -m "feat: xxx"
```

2.13 pick 新分支的修改

```
nhash='git log -1 --format=%h'  
git checkout -  
git cherry-pick $nhash
```

2.14 全局设定默认编辑器

```
git config --global core.editor vim
```

否则 Linux 下默认有可能是 nano

2.15 新建 github 仓库

首先假定 upstream 已经先设置

```
gh repo create --source=. --remote=upstream
```

或者

```
gh repo create xxx
```

2.16 上传

```
git push -u upstream main
```

“-u, -set-upstream” 为设置默认的 fast-forward 源

2.17 获得仓库信息

```
gh repo view
```

输出描述和 README

2.18 通过 branch+merge 来新加特性

```
git branch new-feat
git checkout new-feat
# ... make some change here, then
git commit -m "impr: some feature"
git checkout -
git merge new-feat
```

2.19 添加 commit，从文件中读取内容

```
git commit -m "@file"
```

假设 file 中已经写好了 commit 信息

2.20 使用 mergetool 处理冲突

在出现合并冲突等问题时：

```
git config merge.tool vimdiff
git mergetool
```