

实验报告

23020007067 李子昊

2024 年 8 月 29 日

目录

1	课后练习	1
1.1	练习内容	1
1.2	练习结果	1
2	git 实例	3
2.1	git 实例展示	3
2.2	git 个人心得	8
3	latex 实例	9
3.1	latex 实例展示	9
3.2	latex 个人心得	11
4	github 仓库网址	11

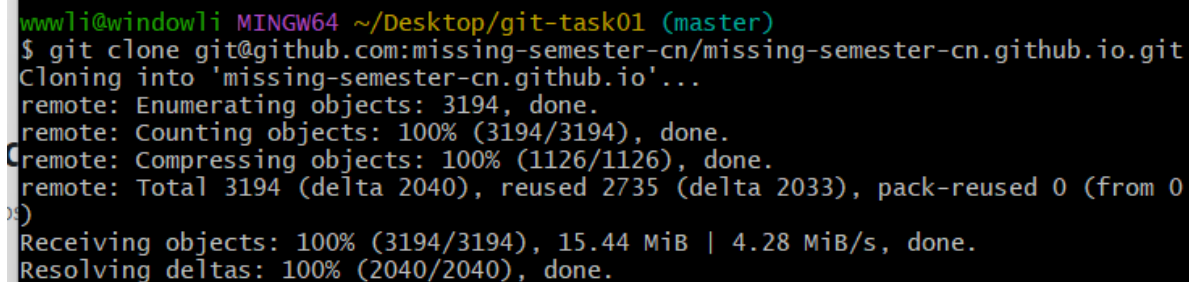
1 课后练习

1.1 练习内容

1. 如果您之前从来没有用过 Git，推荐您阅读 Pro Git 的前几章，或者完成像 Learn Git Branching 这样的教程。重点关注 Git 命令和数据模型相关内容；
2. 克隆本课程网站的仓库
 - 2.1 将版本历史可视化并进行探索
 - 2.2 是谁最后修改了 README.md 文件？（提示：使用 git log 命令并添加合适的参数）
 - 2.3 最后一次修改 _config.yml 文件中 collections: 行时的提交信息是什么？（提示：使用 git blame 和 git show）

1.2 练习结果

1. 习题一：克隆版本控制网址 (Git) 的仓库



```
wwwli@windowli MINGW64 ~/Desktop/git-task01 (master)
$ git clone git@github.com:missing-semester-cn/missing-semester-cn.github.io.git
Cloning into 'missing-semester-cn.github.io'...
remote: Enumerating objects: 3194, done.
remote: Counting objects: 100% (3194/3194), done.
remote: Compressing objects: 100% (1126/1126), done.
remote: Total 3194 (delta 2040), reused 2735 (delta 2033), pack-reused 0 (from 0)
Receiving objects: 100% (3194/3194), 15.44 MiB | 4.28 MiB/s, done.
Resolving deltas: 100% (2040/2040), done.
```

图 1: 1

- 2 习题二：

- 2.1 将'git log --pretty=oneline --all --graph --abbrev-commit' 取别名并且查看历史版本。

```

wwwli@windowli MINGW64 ~/Desktop/git-task01 (master)
$ alias git-log='git log --pretty=oneline --all --graph --abbrev-commit'

wwwli@windowli MINGW64 ~/Desktop/git-task01 (master)
$ git-log
* 764966c (HEAD -> master, origin/master) first

wwwli@windowli MINGW64 ~/Desktop/git-task01 (master)
$ cd missing-semester-cn.github.io

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git-log
* af054fa (HEAD -> master, origin/master, origin/HEAD) Merge pull request #172 from pspdad
|
| * 9baa48c remove irrelevant text
| * f5df7de fix wrong index
| * ef9a2f7 fix typo
|/
|
| * dd3f3dd Merge pull request #171 from HowieChih/for-better-understanding
|/
|
| * 8e26b4a 更新#课程概览与 shell##一个功能全面又强大的工具关于修改亮度文件报错的翻译
| * 4e2ff43 更新#课程概览与 shell##一个功能全面又强大的工具关于修改亮度文件报错的翻译
|/
|
| * d284d3e Merge pull request #170 from crosscap/typo-fix
|/
|
| * 6e27b4f fix some typo in version-control.md
|/
|
| * f3773ac Merge pull request #169 from xjzh123/patch-1
|/
|
| * 45897e1 修复multiple错译为“乘积”
| * ede827e Merge pull request #168 from DXShelley/master
|/
|
| * 6221bbc Merge branch 'master' of https://pi-gitserver.kooldns.cn/dxshelley/missing-sem

```

图 2: 2.1

2.2 查看 README.md 文件的最近一次修改

```

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git-log -1 README.md
* de98852 将readme文件中的url的绝对路径改为相对路径，不用重复访问github，利于分享传播

```

图 3: 2.2

2.3 查看某个文件的最近一次修改了什么信息

```

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git blame _config.yml | grep collections
a88b4eac (Anish Athalye 2020-01-17 15:26:30 -0500 18) collections:

```

图 4: 2.3

2 git 实例

2.1 git 实例展示

表 1: git 实例展示

1	git clone	克隆仓库
2	git remote add origin	将本地与远端连接
3	alias git-log='git log --pretty=oneline '	取别名
4	git-log -1 README.md	查看某个文件的最近一次修改
5	git blame config.yml grep collections	查看某个文件的最近一次修改了什么信息
6	git remote -v	查看远程版本库信息
7	git pull origin master	下载代码并且快速合并
8	git branch dev1	创建分支
9	git branch -d dev1	删除分支
10	git tag li	创建标签
11	git tag -d li	删除标签
12	git merge dev1	合并分支
13	git mv file1.txt file2.txt	文件改名
14	git rm --cached file2.txt	停止跟踪文件但不删除
15	git commit --amend	修改最后一次提交
16	git push origin master	上传代码并且快速合并
17	git pull origin master --allow-unrelated-histories	将远程分支合到当前分支
18	git fetch origin master	A 从远端仓库获取
19	git push -f --set-upstream origin master:master	强制上传至远程
20	git branch	查看分支

```
wwwli@windowli MINGW64 ~/Desktop/git-task01 (master)
$ git clone git@github.com:missing-semester-cn/missing-semester-cn.github.io.git
Cloning into 'missing-semester-cn.github.io'...
remote: Enumerating objects: 3194, done.
remote: Counting objects: 100% (3194/3194), done.
remote: Compressing objects: 100% (1126/1126), done.
remote: Total 3194 (delta 2040), reused 2735 (delta 2033), pack-reused 0 (from 0)
Receiving objects: 100% (3194/3194), 15.44 MiB | 4.28 MiB/s, done.
Resolving deltas: 100% (2040/2040), done.
```

图 5: 克隆版本控制网址 (Git) 的仓库

```

wwwli@windowli MINGW64 ~/Desktop/git-task01 (master)
$ git remote add origin git@github.com:newbeginnerlzh/git-task01.git

wwwli@windowli MINGW64 ~/Desktop/git-task01 (master)
$ git push -u origin master
Enumerating objects: 2, done.
Counting objects: 100% (2/2), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 216 bytes | 216.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To github.com:newbeginnerlzh/git-task01.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

```

图 6: 将本地仓库与远端仓库连接并将本地仓库内容传到远端。
<https://github.com/newbeginnerlzh/git-task01.git>

```

wwwli@windowli MINGW64 ~/Desktop/git-task01 (master)
$ alias git-log='git log --pretty=oneline --all --graph --abbrev-commit'

wwwli@windowli MINGW64 ~/Desktop/git-task01 (master)
$ git-log
* 764966c (HEAD -> master, origin/master) first

wwwli@windowli MINGW64 ~/Desktop/git-task01 (master)
$ cd missing-semester-cn.github.io

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git-log
* af054fa (HEAD -> master, origin/master, origin/HEAD) Merge pull request #172 from pspdad
|
| * 9baa48c remove irrelevant text
| * f5df7de fix wrong index
| * ef9a2f7 fix typo
|/
|
| * dd3f3dd Merge pull request #171 from HowieChih/for-better-understanding
|/
| * 8e26b4a 更新#课程概览与 shell##一个功能全面又强大的工具关于修改亮度文件报错的翻译
| * 4e2ff43 更新#课程概览与 shell##一个功能全面又强大的工具关于修改亮度文件报错的翻译
|/
|
| * d284d3e Merge pull request #170 from crosscap/typo-fix
|/
| * 6e27b4f fix some typo in version-control.md
|/
|
| * f3773ac Merge pull request #169 from xjzh123/patch-1
|/
|
| * 45897e1 修复multiple错译为“乘积”
| * ede827e Merge pull request #168 from DXShelley/master
|/
|
| * 6221bbc Merge branch 'master' of https://pi-gitserver.kooldns.cn/dxshelley/missing-sem

```

图 7: 将'git log --pretty=oneline --all --graph --abbrev-commit' 取别名并且查看历史版本。

```

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git-log -1 README.md
* de98852 将readme文件中的url的绝对路径改为相对路径，不用重复访问github，利于分享传播

```

图 8: 查看某个文件的最近一次修改

```
wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git blame _config.yml | grep collections
a88b4eac (Anish Athalye 2020-01-17 15:26:30 -0500 18) collections:
```

图 9: 查看某个文件的最近一次修改了什么信息

```
wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git remote -v
origin  git@github.com:missing-semester-cn/missing-semester-cn.github.io.git (fetch)
origin  git@github.com:missing-semester-cn/missing-semester-cn.github.io.git (push)
```

图 10: 查看远程版本库信息

```
wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git fetch origin
```

图 11: 从远程库中获取代码

```
wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git pull origin master
From github.com:missing-semester-cn/missing-semester-cn.github.io
* branch      master      -> FETCH_HEAD
Already up to date.
```

图 12: 下载代码并且快速合并

```
wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ touch file1.txt

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

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

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

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git add .

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git commit -m "second"
[master 20b5a21] second
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 file1.txt
```

图 13: 创建文件并将其保存到本地仓库

```
wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ vi file1.txt
```

图 14: 用 vi 编译文本文件

```

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git branch dev1

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git branch
dev1
* master

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git branch -d dev1
Deleted branch dev1 (was 20b5a21).

```

图 15: 创建分支、显示分支、删除分支

```

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git tag li

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git tag
li

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git tag -d li
Deleted tag 'li' (was 20b5a21)

```

图 16: 创建标签、显示标签、删除标签

```

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (dev1)
$ git checkout master
Switched to branch 'master'
M   file1.txt
Your branch is ahead of 'origin/master' by 1 commit.
(use "git push" to publish your local commits)

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git merge master
Already up to date.

```

图 17: 切换分支，合并分支

```

wwwli@windowli MINGW64 ~/Desktop/git-task01 (master)
$ git mv file1.txt file2.txt

```

图 18: 文件改名

```

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (dev1)
$ git checkout master
Switched to branch 'master'
M   file1.txt
Your branch is ahead of 'origin/master' by 1 commit.
(use "git push" to publish your local commits)

wwwli@windowli MINGW64 ~/Desktop/git-task01/missing-semester-cn.github.io (master)
$ git merge master
Already up to date.

```

图 19: 停止跟踪文件但不删除

```

wwwli@windowli MINGW64 ~/Desktop/git-task01 (master)
$ git commit --amend
[master 8040f32] first
Date: Wed Aug 28 16:42:21 2024 +0800
1 file changed, 1 insertion(+)
create mode 160000 missing-semester-cn.github.io

```

图 20: 修改最后一次提交

```

wwwli@windowli MINGW64 ~/Desktop/git-task01 (dev1)
$ touch gitignore

```

```

*.a

```

```

.gitignore[+] [unix] (07:59 01/01/1970)
-- INSERT --

```

图 21: 添加文件到忽略列表

```

2 wwwli@windowli MINGW64 ~/Desktop/git-task01 (master)
$ git pull origin master
From github.com:newbeginnerlzh/git-task01
* branch          master      -> FETCH_HEAD
Already up to date.

```

图 22: 下载代码并且快速合并


```
wwwli@windowli MINGW64 ~/Desktop/git-task01 (master)
$ git push origin master
Everything up-to-date
```

图 23: 上传代码并且快速合并

```
wwwli@windowli MINGW64 ~/Desktop/git-task01 (master|MERGING)
$ git push -f --set-upstream origin master:master
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 16 threads
Compressing objects: 100% (12/12), done.
Writing objects: 100% (14/14), 1.23 KiB | 1.23 MiB/s, done.
Total 14 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), done.
To github.com:newbeginnerlzh/git-task01.git
+ 764966c...800d1cf master -> master (forced update)
branch 'master' set up to track 'origin/master'.
```

图 24: 强制将本地仓库上传至远程

2.2 git 个人心得

通过学习了 git, 我掌握了版本控制的核心概念, 这为代码管理提供了强大的工具。Git 允许我们轻松地跟踪项目的所有变化, 回溯到任何一个历史版本, 并在多人协作时避免冲突。理解分支和合并的工作原理, 可以帮助我们更好地组织和协调项目开发, 避免开发过程中可能遇到的混乱。

在我的实际使用中, 认识到了频繁提交 (commit) 的重要性, 每次提交都是一个小而完整的变更。这不仅有助于更清晰地理解项目的历史, 还能够出现问题时快速定位问题的根源。通过编写有意义的提交信息, 可以更轻松地理解每次变更的意图和目的。

最后, 我认为 Git 的协作功能如 pull request 和代码审查使得团队开发更加高效。通过定期同步 (pull) 和推送 (push), 团队成员能够保持代码库的一致性。同时, 利用 Git 的分支策略, 可以为新功能、修复 Bug 或试验性开发创建独立的环境, 确保主代码库的稳定性。

3 latex 实例

表 2: latex 实例展示

- 1 创建目录
- 2 创建章节
- 3 引用图片
- 4 使用 usepackage
- 5 添加标题、作者、日期
- 6 创建表格
- 7 改变字体颜色
- 8 调整字体大小

3.1 latex 实例展示

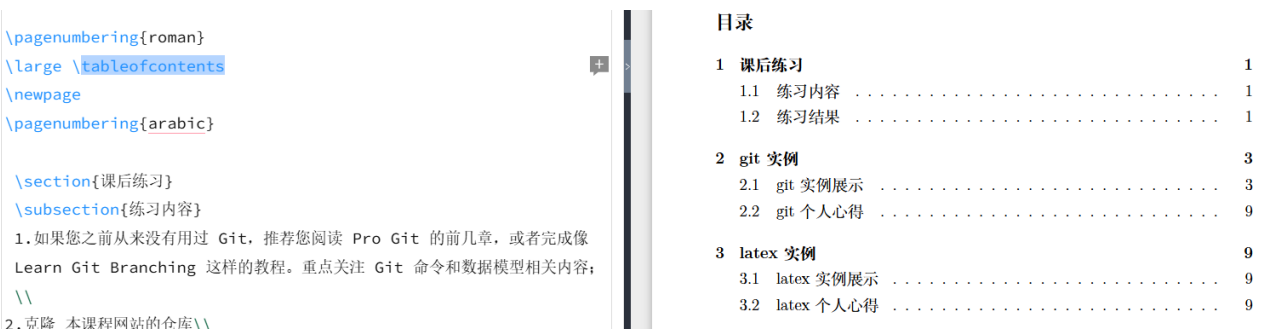


图 25: 创建目录



图 26: 创建章节



图 27: 引用图片

```
\usepackage{color}
\usepackage{float}
\usepackage{graphicx}
\usepackage[left=2cm,right=2cm,top=2cm,bottom=2cm,head=1cm,head
```

图 28: 使用 usepackage

```
\documentclass[a4paper, 12pt]{article}
\usepackage[UTF8]{ctex}
\title{实验报告}
\author{23020007067 李子昊}
\date{\today}
```

实验报告

23020007067 李子昊

2024 年 8 月 29 日

图 29: 添加标题作者日期

```
\begin{table}[H]
\centering
\caption{{\color{red}git实例展示}}
\begin{tabular}{ccl}
1&git clone &克隆仓库 \\
2&git remote add origin &将本地与远端连接 \\
3&alias git-log='git log --pretty=oneline ' &取别名 \\
4&git-log -1 README.md &查看某个文件的最近一次修改 \\
5&git blame _config.yml | grep collections &查看某个文件的最近一次修改了什么信息 \\
6&git remote -v &查看远程版本库信息 \\
7&git pull origin master &下载代码并且快速合并 \\
8&git branch dev1 &创建分支 \\
9&git branch -d dev1 &删除分支 \\
10&git tag li &创建标签 \\
11&git tag -d li &删除标签 \\
12&git merge dev1 &合并分支 \\
13&git mv file1.txt file2.txt &文件改名
\end{tabular}
\end{table}
```

4.1 git 实例展示

text

表 1: git 实例展示

1	git clone	克隆仓库
2	git remote add origin	将本地与远端连接
3	alias git-log='git log --pretty=oneline '	取别名
4	git-log -1 README.md	查看某个文件的最近一次修改
5	git blame _config.yml grep collections	查看某个文件的最近一次修改了什么信息
6	git remote -v	查看远程版本库信息
7	git pull origin master	下载代码并且快速合并
8	git branch dev1	创建分支
9	git branch -d dev1	删除分支
10	git tag li	创建标签
11	git tag -d li	删除标签
12	git merge dev1	合并分支
13	git mv file1.txt file2.txt	文件改名
14	git rm --cached file2.txt	停止跟踪文件但不删除
15	git commit --amend	修改最后一次提交
16	git push origin master	上传代码并且快速合并
17	git pull origin master --allow-unrelated-histories	将远程分支合并到当前分支
18	git fetch origin master	A 从远端仓库获取
19	git push -f --set-upstream origin master:master	强制上传至远程
20	git branch	查看分支

图 30: 创建表格

subsection{git个人心得}

{\color{blue}通过学习了git,我掌握了版本控制的核心概念,这为代码管理提供了强大的工具。}Git允许我们轻松地跟踪项目的所有变化,回溯到任何一个历史版本,并在多人协作时避免冲突。理解分支和合并的工作原理,可以帮助我们更好地组织和协调项目开发,避免开发过程中可能遇到的混乱。

在我的实际使用中,认识到了频繁提交(commit)的重要性,每次提交都是一个小小而完整的变更。这不仅有助于更清晰地理解项目的历史,还能够在出现问题时快速定位问题的根源。通过编写有意义的提交信息,可以更轻松地理解每次变更的意图和目的。

最后,我认为 Git 的协作功能如 pull request 和代码审查使得团队开发更加高效。通过定期同步(pull)和推送(push),团队成员能够保持代码库的一致性。同时,利用 Git 的分支策略,可以为新功能、修复 Bug 或试验性开发提供清晰的流程。

图 31: 改变字体颜色

```
\large \tableofcontents
```

1 课后练习

1

图 32: 调整字体大小

3.2 latex 个人心得

学习 LaTeX 的过程中，我深刻体会到它作为一种强大排版工具的优势。LaTeX 让你能够专注于内容本身，而不必为格式问题烦恼。它的高度可定制性和丰富的包支持，使得我可以创建专业水准的文档，无论是学术论文、演讲稿，还是复杂的数学公式，LaTeX 都能轻松胜任。

在实际使用中，掌握基本的命令和结构是关键。起初，LaTeX 的语法可能显得有些繁琐，挺令人崩溃。但当熟悉了如何使用环境 (environment)、命令和包 (package)，我会发现其强大的排版能力和一致性远超其他工具。特别是在处理长篇文档时，LaTeX 的自动目录生成、引用管理和跨章节的引用功能尤为实用。

此外，良好的代码管理习惯同样重要。将文档拆分成多个部分，合理组织文件结构，可以提高项目的可维护性。通过使用版本控制工具如 Git，你可以轻松跟踪文档的变化，确保每个版本的稳定性和可回溯性。整体来说，LaTeX 不仅是一个文档排版工具，更是一种高效、专业的工作方式。

4 github 仓库网址

<https://github.com/newbeginnerlzh/git-task01.git>