Latex 与 Git

23020007011 崔涛

2024 年 9 月 11 日

1 实验目的

本次课程主要讲授了版本控制（Git）以及 Latex 文档编辑，通过对两者的学习来加强对两大便捷工具的使用

2 介绍

2.1 两大工具的优点

Git 是一个分布式版本控制系统，它允许使用者跟踪文件和目录的变化历史 Git 使得多人协作变得更加容易，多个开发者可以在同一个项目上工作，并轻松地合并各自的更改，方便了多人协作。

LaTeX 是一个高质量的排版系统，适合生成科学和数学文档。Latex 能够处理复杂的公式和表格，并自动处理文档的格式和布局，方便了文章排版。

3 练习内容

3.1 Git 学习样例 10 个

1. git init 可以在当前目录下创建 git 仓库, 第一次使用需要进行如下配置 git config [–global] user.name ”Your Name” git config [–global] user.email ”email@example.com”

2. Git 中有关文件提交的基础命令

- `git status`: 显示当前的仓库状态
- `git add <filename>`: 添加文件到暂存区
- `git commit`: 创建一个新的提交
- `git log`: 显示历史日志
- `git log --all --graph --decorate`: 可视化历史记录
- `git diff <filename>`: 显示与暂存区文件的差异
- `git diff <revision> <filename>`: 显示某个文件两个版本之间的差异
- `git checkout <revision>`: 更新 HEAD 和目前的分支

3.git reset 进行版本回退，**可以指定退回某一次提交的版本**。其格式为：git reset [-soft|-mixed|-hard] [HEAD]

- –soft 参数对于工作区和暂存区的内容都不变，只是将版本库回退到某个指定版本。
- –mixed 为默认选项，使用时可以不用带该参数。该参数将暂存区的内容退回为指定提交版本内容，工作区文件保持不变。
- –hard 参数将暂存区与工作区都退回到指定版本。切记工作区有未提交的代码时不要用这个命令，因为工作区会回滚。

4. 可以通过`git checkout -- [file]`让工作区的文件回到最近一次 `add` 或 `commit`时的状态。

5. 克隆 本课程网站的仓库并将版本历史可视化并进行探索找出是谁最后修改了 `README.md` 文件？查看最后一次修改 `_config.yml` 文件中 `collections:` 行时的提交信息是什么？

- 使用 git clone 加上仓库链接即可克隆仓库。
- git log –all –graph –decorate 将版本可视化历史化并进行探索
- 使用 git log -1 README.md 即可找出最后修改的人
- git blame _config.yml | grep collections 即可找出最后一次修改 `_config.yml` 文件中 `collections:` 行时的提交信息

6. 用 Git 时的一个常见错误是提交本不应该由 Git 管理的大文件，或者将含有敏感信息的文件提交给 Git。尝试向仓库中添加一个文件并添加提交信息，然后将其从历史中删除

- 输入（echo ”password123”>my_password）（git add .）（git commit -m ”add password123 to file”）（git log HEAD）这四条指令来提交一些敏感信息
- 使用`git filter-branch`删除记录，输入 git filter-branch –force –index-filter ’git rm –cached –ignore-unmatch ./my_password’ –prune-empty –tag-name-filter cat – –all 即可

7. 在 ~/.gitconfig 中创建一个别名，使在运行 git graph 时，您可以获得 git log –all –graph –decorate –oneline 的输出结果。

解答: 向文件中写入如下内容

[alias]

graph = log –all –graph –decorate –oneline

8. 配置全局 gitignore 文件来自动忽略系统或编辑器的临时文件

解答：通过执行 git config –global core.excludesfile ~/.gitignore_global 在 ~/.gitignore_global 中创建全局忽略规则，输入如下指令即可。git config –global core.excludesfile /.gitignore .Name

9. git 的分支操作

- git branch branch_name 创建分支
- git branch 查看分支

- git branch branch_name 切换到指定分支上的最新版本
- git merge branch_name 合并分支

10. 打开任意版本：git show hash(哈希值)

3.2 Latex 学习例子 10 个

1. 标题、日期与作者，这里可以用\title{}、\author、\date{}来设置，日期中选择\today会自动输出编译当天的日期。但是它们只能放在\documentclass与\begin{document}之间的区域），然后通过\maketitle设来展现出来。

2. 目录的实现：需要加入\tableofcontents

3. 插入图片需要使用graphicx宏包, 需要使用\includegraphics[width=8cm]{名字}来实现

4. 无序列表的插入使用\begin{itemize}和 \item和\end{itemize}即可 |

5. 有序列表使用\begin{enumerate} 和 \end{enumerate}即可 |

6. 插入代码：可以调用listings宏包，并且在正文中使用 lstlisting 指令框住所想插入的代码 |

```
\begin{listing}[language=python]
代码
\end{listing}
```

7. 参考文献使用 natbib 包

```
\usepackage{natbib}
参考
\begin{document}
```

8.Latex 中空格的添加

```
\enspace
quad
```

9. 使用 geometry 宏包可以设置页面参数，如下所示

$$\usepackage[top = 30mm, bottom = 30mm, inner = 20mm, outer = 45mm]\{geometry\}$$

10. 控制首行缩进。首先引用宏包

$$\usepackage\{indentfirst\}$$

之后使用

$$\setlength\{\parindent\}\{2em\}$$

控制下文首行缩进两字符。

4 解题感悟收获

通过学习 LaTeX，我学会了如何进行高效排版，学会了常见的文章结构布局的实现方式，可以随心所欲控制文章格式，还学会了使用模板来写作，大大提高了我的论文写作效率，Latex 还提高了我的文档排版的正确度，学会了利用 Latex 自动为题注进行编号并在文档中进行正确的引用，减

少了手动编号可能导致的错误。

通过学习 Git，我学会了如何进行版本控制，掌握了 Git 仓库的一些基本操作，这大大提高了我的团队协作效率，版本控制也不再是问题，我还学会了如何在大型项目中利用 Git 来变更追踪和管理，为以后的工作打下基础。github 路径您可以在此查看项目的源代码:

`https://github.com/cuitao223/homework1`