Git 操作

唐浩晋 中国科学院大学

2021/7/18

汇报提纲

- Git 简介与安装
- 基本操作
- 开发流程和规范
- 进阶操作

第一部分

GIT 简介与安装

什么是 Git

- 分布式版本控制系统
- Linus 当年写 Linux 的时候找不到一个合适的版本控制系统, 于是自己整了一个
- 版本控制
- 分布式备份
- 协作

如何安装 Git

```
sudo apt install git
```

第二部分

基本操作

基本操作汇总

```
git init
git add
git status
git ls-files
git commit [--amend] [-m <message>]
git log
git reflog
git restore –staged <file>
git branch [(new-branch|-d <old-branch>)]
git checkout [-b] <branch>
git merge <branch>
git remote [(add <name> <url>|rename <old> <new>|remove <name>)]
git push [-u <remote>] [branch]
git clone <url> [path]
git diff [(commit ID [commit ID 2]|--cached [commit ID])]
```

第三部分

开发流程和规范

分支规范

- master

  发布分支, 其应当能够正常工作, 且经过充分的评估和测试. 一般**不会在该分支上开发**.

- develop

  开发分支, 其应当能够工作. 所有的开发工作都应该基于它, 但一般不会**直接**在该分支上开发.

- 临时分支

  开发时应该基于 develop 分支新建临时分支, 其命名应当遵循一定的**约定**且**有意义**, 如 feat-pipeline, fix-23 等.

分支规范

- 开发过程中应该经常同步本地的 develop 分支, 并将其合并入开发分支.
- 开发结束后, 将临时分支并入 develop 分支, 并将临时分支删除.
- 在经过**充分评估和测试**后, 才可以将 develop 分支并入 master 分支.

提交信息规范

- 基于约定式提交

  https://www.conventionalcommits.org/zh-hans/
- 提交信息的组成

```
> <header>
>
> [body]
>
> [footer]
```

提交信息规范

University of Chinese Academy of Sciences

- header

  只有一行, 包括 `<type>`, `[scope]`, `<subject>` 三个字段

  `<type>[scope]: <description>`

  - `type`: 用于说明 commit 类型， 一般分为以下几种
    - `build`: 与构建流程等有关的改动
    - `feat`: 新增 feature
    - `fix`: 修复八阿哥
    - `refactor`: 不改变行为的, 对代码结构的改动
    - `style`: 对代码风格的改动 (仅限缩进, 空行一类的简单改动)
    - `docs`: 对文档和注释的改动
    - `...`

提交信息规范

- header

  只有一行, 包括 `<type>`, `[scope]`, `<subject>` 三个字段

  `<type>[scope]: <description>`

  - `scope`: 用于说明此次 commit 影响的范围
  - `description`: 对代码变更的简短总结

提交信息规范

- body

  可选. 用于说明此次修改的动机和修改前后程序的行为差异.

- footer

  如果包含不兼容的修改, 则需要在此提及.

```
> BREAKING CHANGE: <description and migration help>
```

  如果更改涉及类似 GitHub 中的 issues 时, 也可以在此提及.

```
> Fixed #<issue number>
```

提交粒度规范

提交的粒度需要以功能点为单位，每次实现新功能后进行提交，并遵循：

- 将离散的任务划分到多次 commit 操作中，比如修复了两个不同的 bug 需要进行两次提交
- 在提交之前对提交结果进行充分测试，不要提交未完成的工作

通常可以不严谨地认为，如果你在编写提交信息时遇到了困难，那就说明提交的粒度太大.

文件跟踪规范

- 忽略自动生成的文件, 比如缩略图等.
- 忽略编译生成的中间文件、可执行文件等.
  - 如果一个文件是**通过另一个文件自动生成**的, 那就没必要放进版本库, 如 `Chisel` 编译中间文件 `*.fir`
- 忽略你自己的带有**敏感信息**的配置文件
  - 存放口令的配置文件.

第四部分

进阶操作

进阶操作

- https://git-man.tanghaojin.site/进阶操作.html
- https://git-scm.com/book/zh/v2

谢谢，欢迎批评指正！

中国科学院大学
University of Chinese Academy of Sciences