Git Instructions Summary

阅读说明

> 1.正文内容中圆弧括号 '()' 代表可选，即可省略
> 2.正文内容中方括号 '[]' 代表以指定类型格式字符串内容替换
> 3.此文针对 Git命令提示符(Git Bash) 操作环境，类似 DOS 操作方式
> 4.一般命令格式为：$ git [cmd] (- auxiliary character) ([argv1] ……)
> 5.'$' 一般已经给出，'git' 基本是所有命令的前缀，此两项可能省略（有特殊情况会额外说明）
> 6.部分指令操作需要注意操作位置和文件路径问题，即 cd 指令的结合使用
> 7.默认主分支由 master 更改为 main (master == main)，可能与政治原因有关，在指令输入时注意

The Installation and Preparation Of Local Environment (for Github)

Windows-10

> 1. 前往官网安装 Git
> 2. 本地安装配置 Git （勾选配置的时候以默认为主，可以根据实际情况稍加改动）
> 3. 打开 Git Bash 开始配置
> 4. 配置 本地用户名 / 邮箱 ，执行如下命令:
>
> ```
> git config --global user.name "your_name" git config --global
> user.email "your_email"
> ```
>
> 5. 生成密钥，执行如下命令:
>
> ```
> ssh-keygen -t rsa -C "your_email"
> ```
>
> 6. 将公钥（ .ssh 文件中的 rsa.pub ）添加到 GitHub 账户中(注意密钥名称格式应遵循相应的规范)
> 7. 测试，执行如下命令:
>
> ```
> ssh -T git@github.com
> ```

Linux-Ubuntu22.04

> 1. 打开终端（在终端中进行操作，包括安装和后面的使用）
> 2. 更新系统软件包依赖（可选），执行如下命令:
>
> ```
> sudo apt-get update
> ```
>
> 3. 安装 Git ，执行如下命令:
>
> ```
> sudo apt-get install git
> ```
>
> 4. 安装测试，执行如下命令:
>
> ```
> git --version
> ```
>
> 5. 配置 本地用户名 / 邮箱 ，执行如下命令:
>
> ```
> git config --global user.name "your_name" git config --global
> user.email "your_email"
> ```
>
> 6. 生成密钥，执行如下命令:
>
> ```
> ssh-keygen -t rsa -C "your_email"
> ```
>
> 7. 将公钥（ .ssh 文件中的 rsa.pub ）添加到 GitHub 账户中(注意密钥名称格式应遵循相应的规范)
> 8. 测试，执行如下命令:
>
> ```
> ssh -T git@github.com
> ```

Auxiliary Characters

- -h (--)help

> 指令使用查询/帮助文档
>
> - $ git help [cmd] (argvs)

User

- config

> - 设置查询更改（版本库，系统，全局……）指令
> -
> - $ git config --list
> -
> - $ git config (--global) user.name
> - $ git config (--global) user.email
> -
> - $ git config (--global) origin_name "new_name"
> - $ git config (--global) origin_email "new_email"

WorkArea

- clone

> 下载（拷贝）项目
>
> - $ git clone [url]
>
> url : Uniform Resource Locator 统一资源定位符（网址等）

- init (initialize)

> 初始化指令
>
> - $ git init
> - $ git init [project-name]
> - $ git init --bare <directory-name>

- clean

Repository(self/unite)

- fetch
- pull
- push

> 推送更新指令
>
> - $ git push (origin) (main) 无参数则默认为 origin 远端的 master 分支
> - $ git push [remote respository name] <branch_name>

> - $ git push [remote respository name] --force (???)
> - $ git push [remote respository name] --all (???)

- remote

> 远程同步指令
>
> - $ git remote -v
> - $ git remote show [remote repository name]
> - $ git remote add [repository_name] [URL]
> - $ git remote add set-url origin [URL_SSH] (???)
> - $ git remote rm <repository_name>

History Change (self)

- branch
- commit

> 提交指令 将暂存区的更改保存为一个提交到本地仓库区以准备推送
>
> - $ git -m [message]
> - $ git [file1] [file2] ... -m [message]
> - $ git commit -a (???)提交工作区自上次commit之后的变化，直接到仓库区
> - $ git commit -v (???)提交时显示所有diff信息

- merge
- rebase
- reset
- switch
- tag

File Change

- add

> 文件添加指令 将文件添加至工作区中，使 Git 可以追踪记录文件的修改并推送
>
> - $ git add .
> - $ git add [(Drive:)/sub-directory/.../]
> - $ git add specify_file.js
> - $ git add (Drive:)/sub-directory/.../specify_file.js
> - $ git add [file1] [file2] ...
> - $ git add -p 添加每个变化前，都会要求确认

- rm (remove)

> 文件删除指令 将文件从工作区中删除，不会再影响后续 Git 所有提交推送
>
> - $ git rm specify_file.js
> - $ git rm (Drive:)/sub-directory/.../specify_file.js
> - $ git rm [file1] [file2] ...
> - $ git rm --cached [file] 停止追踪指定文件，但该文件会保留在工作区

- mv (move)

> 文件修改指令（重命名，移动）
>
> - $ git mv [original_name] [new_name]
> -
> - $ git mv [filename.js] Drive:/new-path/.../filename.js
> -
> - $ git mv -f [filename_original] [filename_existing] 强制重命名或移动（覆盖）

- restore

Large Files Storeage

```
git lfs install

git lfs track "*.ext"

git add .gitattributes

git add .

git commit -m "Add large file using Git LFS"

git push
```

Examine The history && state

- bisect
- diff
- grep
- log
- show
- status

> 分支状态查询指令
>
> - $ git status

Reference Website

- Git 命令参考手册整理
- Git-Github 基础操作
- Git 命令使用文档