命令行与文件系统--自动化(图形界面之下)

一、基本概念

1.目录树:像树一样的结构,文件夹(目录)可以包含子文件夹和文件,形成层级关系。

2.根目录: 目录树的最顶层, 所有文件和文件夹的起点: /。

3.路径: 文件或文件夹在目录树中的位置。

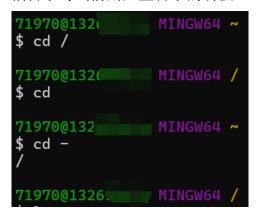
4.绝对路径: 从根目录开始的完整路径,在任何位置都能正确访问文件,例如 `/home/user/file.txt` (Linux) 或 `C:\Users\User\file.txt` (Windows)。

5.相对路径: 依赖当前工作目录的路径,例如 `documents/report.doc`(当前在 `C:\Users\User\`时)。

6.Unix 路径: / **Windows 路径:** \ (严格来说没有根目录)

| 符号 | 解释 |
|----|---------------|
| 1 | 根目录 (所有路径的起点) |
| | 当前目录 |
| | 上一级目录 |
| ~ | 当前用户的主目录 |
| - | 上次访问的目录 |

PS: 根目录与当前用户主目录的切换:

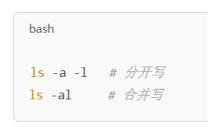


二、Shell 基本语法结构

1.空格:分隔符,不同部分用空格隔开,否则会被视为一个整体(如果路径或参数有空格,需要用引号""或"括起来)

2.短选项:由-引导,后面紧跟一个字母,可以合并使用

示例:



3.长选项:由 -- 引导,通常是完整单词

示例:

```
bash
ls --all --long
```

PS: 长短选项的一些区别

ls -a 和 ls --all 的功能是相同的,都是 **列出所有文件,包括隐藏文件(以 . 开头的文件)**。

区别:

- 1. ls -a
 - 使用 **短选项** (-a)。
 - 短选项可以合并, 比如 ls -al (同时 -a 和 -1)。
 - 适合快速输入。
- 2. ls --all
 - 使用 **长选项** (--all)。
 - 可读性更强,适合写脚本时提高可理解性。
- 4.参 数:一般是命令的目标,如文件、目录、字符串等

示例:

ls →命令
 -1 →选项
 /home → 参数

PS: 常用的 ls 命令 (可阅读在线手册或终端输入命令 ls --help)

```
# 列出当前目录的文件和文件夹(不包含隐藏文件)
1s
           # 显示所有文件,包括隐藏文件(以 `.` 开头的文件)
ls -a
           # 详细列表模式(显示权限、所有者、大小、时间等)
ls -1
          # 详细模式 + 显示隐藏文件 (等同于 `ls -a -l`)
ls -al
          # 以人类可读格式显示文件大小(如 KB、MB)(通常与 `-l` 搭配)
ls -h
ls -t
          # 按修改时间排序(最近修改的文件排在最前)
          # 按文件大小排序(最大文件排在最前)
ls -S
ls /home/user # 列出 `/home/user` 目录的文件
ls -1 ~/Desktop # 查看 `Desktop` 目录的详细信息
```

三、常用命令汇总

```
显示当前所在的目录路径
pwd
     显示当前目录下的文件和文件夹
ls
     读取并显示文件内容 (纯文本)
cat
     切换目录
cd
ср
     复制文件(cp -r 递归复制目录/文件夹)
       71970@13269. MINGW64 ~/Desktop
      $ cp before.doc ../Downloads/after.doc
      71970@1326. MINGW64 ~/Desktop
       cp before.doc ../Downloads
mv
     移动/重命名文件或目录(无 -r)
mkdir
     创建新目录
     删除文件 (rm -r 删除文件夹, -rf 强制删除), 不可找回
rm
df
     查看磁盘剩余空间
      71970@1326′ MINGW64 ~/Desktop
      $ df -h
      Filesystem
                         Size Used Avail Use% Mounted on
                              193G 184G 52% /
      C:/Program Files/Git
                         376G
                         100G
                               80G
                                    21G
                                        80% /e
     查看文件/文件夹占用磁盘空间, du-d 指定深度
du
      71970@13269 MINGW64 ~/Desktop
      $ du -h -d 1 .
      40M
             ./
      14M
      280K
      108M
      8.8G
      297M
             ./金融计算机语言
      1.4M
      9.3G
```

PS: 大模型解释 Bash 命令: du -s * | sort -nr > ~/report.txt 这个命令的作用是**计算当前目录下各文件/文件夹的大小,按大小降序排序,并将结果保存到** ~/report.txt **文件中**。

命令解析

将可读模式加入该命令,运行并查看结果:

四、创建私密代码仓库

1.GitCode 新建项目(私密代码仓)

2.Clone 到本地:

```
71970@132699 MINGW64 ~
$ cd repo/
                MINGW64 ~/repo
$ git clone git@gitcode.com:kaki5358979/kakiwork.git
Cloning into 'kakiwork'...
warning: You appear to have cloned an empty repository.
71970@132699 MINGW64 ~/repo
$ git clone git@gitcode.com:kaki5358979/kakiwork.git
fatal: destination path 'kakiwork' already exists and is not an
pty directory.
71970@13269 MINGW64 ~/repo
$ ls -l
total 4
drwxr-xr-x 1 71970 197609 0 3月 16 16:44 kakiwork/drwxr-xr-x 1 71970 197609 0 3月 11 23:00 week01/
               MINGW64 ~/repo
71970@132
$ cd kakiwork/
71970@132 MINGW64 ~/repo/kakiwork (main)
$ git log
fatal: your current branch 'main' does not have any commits yet
```

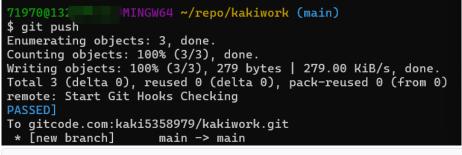
3.向仓库添加文件:

```
MINGW64 ~/repo/kakiwork (main)
$ cp ~/Desktop/金融计算机语言/测试文档.txt ./
71970@13269 MINGW64 ~/repo/kakiwork (main)
$ ls -al
total 5
drwxr-xr-x 1 71970 197609 0 3月 16 16:58 ./
drwxr-xr-x 1 71970 197609 0 3月 16 16:44 ../
drwxr-xr-x 1 71970 197609 0 3月 16 16:44 .git/
-rw-r--r-- 1 71970 197609 15 3月 16 16:58 测试文档.txt
71970@132. MINGW64 ~/repo/kakiwork (main)
$ git status
On branch main
No commits yet
Untracked files:
  (use "git add <file>..." to include in what will be committed
nothing added to commit but untracked files present (use "git a
to track)
              MINGW64 ~/repo/kakiwork (main)
$ git config --global core.quotepath false
71970@1326
                 MINGW64 ~/repo/kakiwork (main)
$ git status
On branch main
No commits yet
Untracked files:
  (use "git add <file>..." to include in what will be committed
nothing added to commit but untracked files present (use "git a
```

3.向仓库添加文件(续):

```
7<u>1970@13269</u> MINGW64 ~/repo/kakiwork (main)
$ git add .
71970@13269
               MINGW64 ~/repo/kakiwork (main)
$ git status
On branch main
No commits yet
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file: 测试文档.txt
71970@13269
               MINGW64 ~/repo/kakiwork (main)
$<mark>git commit -m</mark> "第一次向个人仓库添加文件"
[main (root-commit) d221956] 第一次向个人仓库添加文件
 1 file changed, 1 insertion(+)
 create mode 100644 测试文档.txt
71970@1326 MINGW64 ~/repo/kakiwork (main)
$ git log
commit d221956941390de77898d25ac62e741f61b8194b (HEAD -> main)
Author: kaki5358979 <kaki5358979@noreply.gitcode.com>
Date:
        Sun Mar 16 17:06:45 2025 +0800
    第一次向个人仓库添加文件
```

4.Push 到平台托管:



| kaki5358979 第一次向个人仓库添加文件 | | | |
|--------------------------|--------------|--------|--|
| d2219569 创建于 31 分钟前 🕒 | | | |
| 文件 | 最后提交记录 | 最后更新时间 | |
| □ 测试文档.txt | 第一次向个人仓库添加文件 | 31 分钟前 | |

一些总结

Git 命令总结

- 1. git status:查看当前仓库状态 (未追踪文件、已修改文件、暂存区状态等)。
- 2. git log: 查看提交历史(默认显示最近的提交记录)。
- 3. git diff: 查看未提交的代码改动 (与上次提交的区别)。
- 4. git add <file>: 将文件添加到暂存区,准备提交。
- 5. git commit -m "消息": 提交暂存区的更改, 并附加提交说明。
- 6. git push origin <分支>: 将本地提交推送到远程仓库的指定分支。