Môi trường phát triển

Lệnh Bash đơn giản

- **cd (thay đổi thư mục):** Lệnh này cho phép bạn điều hướng qua các thư mục khác nhau trong hệ thống tệp Linux. Bạn có thể sử dụng lệnh này để di chuyển đến một thư mục cụ thể, quay lại thư mục trước đó hoặc nhảy đến thư mục gốc của bạn.
  - Ví dụ: `cd development/java/Assignment1` (di chuyển đến thư mục Assignment1)
  - Ví dụ: `cd ..` (quay lại thư mục cha)
  - Ví dụ: `cd ~` (đi đến thư mục gốc của bạn)
- **mkdir (tạo thư mục):** Lệnh này được sử dụng để tạo các thư mục (thư mục) mới tại vị trí hiện tại của bạn.
  - Ví dụ: `mkdir new_folder` (tạo thư mục mới có tên "new_folder")
- **rm (xóa):** Lệnh này được sử dụng để xóa các tệp và thư mục. Hãy cẩn thận khi sử dụng lệnh này, vì các tệp đã xóa thường không thể khôi phục được từ dòng lệnh. Bạn có thể sử dụng cờ `-r` để xóa đệ quy các thư mục và nội dung của chúng.
  - Ví dụ: `rm important_file.txt` (xóa tệp có tên "important_file.txt")
  - Ví dụ: `rm -r old_folder` (xóa thư mục có tên "old_folder" và mọi thứ bên trong thư mục đó)
- **ll (liệt kê các tệp và thư mục):** Lệnh này cung cấp danh sách chi tiết các tệp và thư mục ở vị trí hiện tại của bạn. Lệnh này hiển thị thông tin như quyền, quyền sở hữu, kích thước và ngày sửa đổi.
  - Ví dụ: `ll` (liệt kê các tệp và thư mục ở vị trí hiện tại)
- **. (thư mục hiện tại):** Đây là cách viết tắt đặc biệt để chỉ thư mục hiện tại mà bạn đang ở.
  - Ví dụ: `code .` (mở VS Code trong thư mục hiện tại)
- **.. (thư mục cha):** Đây là cách viết tắt đặc biệt để chỉ thư mục cao hơn một cấp so với vị trí hiện tại của bạn (thư mục cha).
  - Ví dụ: `cd ../..` (lên hai cấp trong cấu trúc thư mục)
- **~ (thư mục gốc):** Đây là cách viết tắt để chỉ thư mục gốc cá nhân của bạn trong hệ thống tệp Linux.
  - Ví dụ: `cd ~` (điều hướng bạn đến thư mục home)
  - Ví dụ: `ll ~/Documents` (liệt kê nội dung thư mục Documents trong thư mục home)
- *** (ký tự đại diện):** Ký tự đặc biệt này được dùng để biểu diễn một hoặc nhiều ký tự khác. Ký tự này thường được dùng để chọn nhiều tệp hoặc thư mục cùng lúc.
  - Ví dụ: `ll *.java` (liệt kê tất cả các tệp có đuôi ".java" trong thư mục hiện tại)
  - Ví dụ: `rm data*` (xóa tất cả các tệp hoặc thư mục bắt đầu bằng "data")

Lệnh Git đơn giản

- **git status:** Lệnh này hiển thị trạng thái hiện tại của kho lưu trữ Git, bao gồm mọi thay đổi đã thực hiện nhưng chưa được cam kết.
  - Ví dụ: `git status`
- **git add:** Lệnh này phân giai đoạn các thay đổi, chuẩn bị để đưa chúng vào lần cam kết tiếp theo.
  - Ví dụ: `git add *` (phân giai đoạn tất cả các thay đổi trong thư mục hiện tại của bạn)
  - Ví dụ: `git add HelloWorld.java` (phân giai đoạn một tệp cụ thể)
- **git commit -m "{message}":** Lệnh này lưu các thay đổi đã phân giai đoạn của bạn cùng với một thông báo mô tả giải thích những gì bạn đã thực hiện. Thay thế `{message}` bằng thông báo cam kết thực tế của bạn.

    - Ví dụ: `git commit -m "Added initial code for HelloWorld"`
- **git push:** Lệnh này tải các cam kết cục bộ của bạn lên kho lưu trữ từ xa (như trên GitHub), cho phép những người khác xem các thay đổi của bạn.
    - Ví dụ: `git push` (đẩy lên kho lưu trữ)

Hệ thống thư mục Linux (Giao diện đơn giản hóa của /home)

Sau đây là giao diện đơn giản hóa về cách sắp xếp các tệp và thư mục cá nhân của bạn trong thư mục `/home` để phát triển:

```
/home/
├── development/
│ ├── java/
│ │ ├── Assignment1/
│ │ │ └── HelloWorld.java
│ │ ├── Assignment2/
│ │ └── Assignment3/
│ └── python/
├── data/
└── notes/
```

Trong cấu trúc này:

- `/home/` là thư mục chính.
    - `development/` là thư mục con trong `/home/`.
        - `java/` là thư mục con trong `development/`.
            - `Assignment1/` là thư mục con trong `java/` và chứa tệp `HelloWorld.java`.
            - `Assignment2/` là thư mục con trong `java/`.
            - `Assignment3/` là thư mục con trong `java/`.
        - `python/` là thư mục con khác trong `development/`.
    - `data/` là thư mục con trong `/home/`.
    - `notes/` cũng là thư mục con trong `/home/`.

Cấu trúc này giúp sắp xếp các loại tệp khác nhau của bạn.

Bắt đầu với Môi trường Phát triển của Bạn

Sau đây là cách bắt đầu làm việc trong môi trường phát triển của bạn:

1. **Mở terminal.** Nhấn nút terminal (có thể là biểu tượng trên thanh tác vụ hoặc bạn có thể cần tìm kiếm "Terminal" hoặc "Ubuntu" trong menu bắt đầu). Terminal này đang chạy một môi trường đặc biệt có tên là WSL (Hệ thống con Windows dành cho Linux), cho phép bạn chạy hệ điều hành Linux (trong trường hợp này là Ubuntu) trực tiếp trên máy tính Windows của bạn. Điều này giúp tách biệt môi trường mã hóa của bạn khỏi các hoạt động máy tính thông thường của bạn.

2. **Điều hướng đến thư mục bài tập của bạn.** Sử dụng lệnh `cd` để di chuyển đến đúng thư mục. Ví dụ, để đến bài tập Java đầu tiên của bạn, bạn sẽ nhập: `cd development/java/Assignment1`.
    - **Mẹo để Tự động hoàn thành:** Khi nhập tên thư mục hoặc tệp trong thiết bị đầu cuối, bạn thường có thể nhấn phím `Tab`. Nếu tên bạn đã nhập đủ duy nhất, thiết bị đầu cuối sẽ tự động hoàn thành phần còn lại của tên cho bạn. Điều này có thể giúp bạn tiết kiệm rất nhiều thời gian nhập!
3. **Mở Visual Studio Code.** Khi bạn đã vào đúng thư mục bài tập, hãy nhập `code .` và nhấn Enter. Lệnh `code` sẽ mở trình soạn thảo Visual Studio Code. `.` (dấu chấm) ở cuối lệnh sẽ yêu cầu VS Code mở trình soạn thảo với thư mục hiện tại làm không gian làm việc.

Biên dịch và chạy Java

Sau đây là cách bạn có thể biên dịch và chạy mã Java của mình:

1. **Điều hướng đến thư mục bài tập của bạn trong thiết bị đầu cuối.** Ví dụ: `cd /home/development/java/Assignment1`.
2. **Biên dịch mã Java của bạn.** Sử dụng lệnh `javac` theo sau là tên tệp Java của bạn (bao gồm phần mở rộng `.java`). Ví dụ, nếu tệp của bạn có tên là `HelloWorld.java`, bạn sẽ nhập: `javac HelloWorld.java`.
3. **Chạy mã Java đã biên dịch của bạn.** Sau khi mã được biên dịch thành công, bạn có thể chạy nó bằng lệnh `java` theo sau là tên tệp Java của bạn (không có phần mở rộng `.java`). Ví dụ: `java HelloWorld`.