Ficheiros

Carlos Tavares

2 de Dezembro de 2024

O que são ficheiros i

Um computador possui vários tipos de memória: **RAM** (transitória), **ROM** (permanente) e os **discos rígidos** (persistente).

Hoje em dia a informação persistente está disponível de muitas maneiras: discos rígidos, pen drives, ou até pela internet. Muitos destes dispositivos têm maneiras diferentes de armazenar a informação.

É necessária uma "camada"de abstração que permita ao utilizador aceder á informação sem ter de se preocupar com detalhes de como ela está armazenada e gerir a consistência do sistema.

O que são ficheiros ii

Resposta: O sistema de ficheiros. O sistema de ficheiros é gerido pelo **sistema operativo**. Geralmente disponíveis em ficheiros e pastas, entidades com o que os utilizadores estão habituados a lidar.

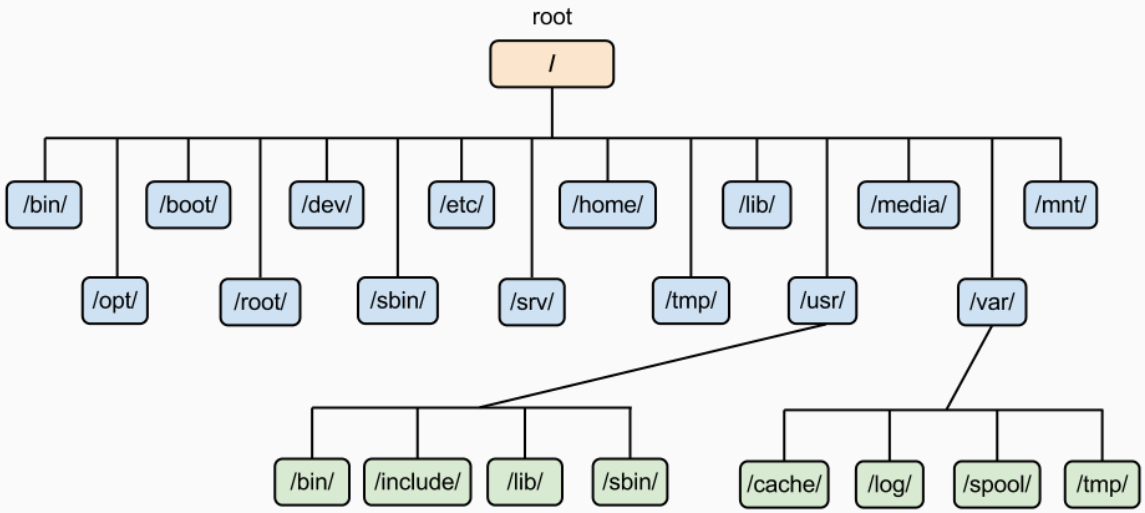

Figura 1: Linux file system

O que são ficheiros iii

Diretórios e ficheiros têm nomes: por exemplo, exemplo_ficheiros **(diretório)** e exemplo.txt **(ficheiro)**.

Um caminho é um diretório + nome_do_ficheiro. Exemplo:

exemplo_ficheiros/exemplo.txt

O que são ficheiros iv

Os caminhos podem ser relativos ou absolutos. Considere que estamos na seguinte pasta:

/algum_caminho/diretório_atual

Um ficheiro exemplo.txt que está no **diretório_atual**, terá um

caminho relativo: exemplo.txt

caminho absoluto: /algum_caminho/diretório_atual/exemplo.txt

Abrir um Ficheiro

- Utilize a função **open()** para abrir um ficheiro.
- Sintaxe:

  objeto_ficheiro = open(nome_ficheiro, [modo_de_acesso])
- Modos de acesso: w, r, a;
- Um objeto ficheiro permite-nos chamar outros métodos de suporte associados a ele.

Ler um Ficheiro

```
# Abrir um ficheiro para leitura
with open('exemplo.txt', 'r') as ficheiro:
    conteúdo = ficheiro.read()
    print(conteúdo)
```

- A instrução **with** trata automaticamente do fecho do ficheiro.
- O método **read()** lê todo o ficheiro de uma só vez.

Exemplo de Iteração num Ficheiro

```
with open('exemplo.txt', 'r') as ficheiro:
    for linha in ficheiro:
        print(linha, end='')
```

- Isto lê cada linha do ficheiro uma de cada vez.
- O parâmetro end='' na função print evita espaçamentos duplos causados pelo caractere de nova linha no final de cada linha.

Escrever num Ficheiro

```
# Abrir um ficheiro para escrita
with open('exemplo.txt', 'w') as ficheiro:
    ficheiro.write('Olá, Mundo!')
```

- Abrir um ficheiro no modo 'w' irá criá-lo se não existir.
- Tenha cuidado, pois irá sobrescrever o ficheiro se já existir.

Adicionar a um Ficheiro

```
# Abrir um ficheiro para adição
with open('exemplo.txt', 'a') as ficheiro:
    ficheiro.write('Adicione esta linha.')
```

- O modo 'a' irá adicionar ao final do ficheiro sem o sobrescrever.
- Também irá criar o ficheiro caso ele não exista.

Fechar um Ficheiro

- É boa prática fechar o ficheiro quando terminar de o utilizar.
- Utilize o método close() para fechar o ficheiro.
- Ao usar with, o ficheiro é fechado automaticamente no final do bloco.

Exercicio. Faça um programa para fazer uma tabela com as frequencias das palavras num ficheiro.

Exercicio. Faça um programa para ler um ficheiro no formato .json

O que são Ficheiros Binários?

Até agora lidamos com ficheiros que contêm texto. É tudo o que há? Não... há **ficheiros binários**!

- Ficheiros binários armazenam dados como bytes brutos.
- Não são legíveis por humanos, ao contrário dos ficheiros de texto.
- Exemplos de utilização comuns incluem:
  - Imagens (.jpg, .png)
  - Áudio (.mp3, .wav)
  - Vídeos (.mp4, .avi)
  - Executáveis compilados (.exe, .bin)
- Ficheiros binários não requerem codificação ou descodificação ao serem lidos ou escritos.

Ler Ficheiros Binários em Python

- Abra ficheiros binários com o modo 'rb'.
- Os dados são lidos como bytes, que podem ser processados ou armazenados.

Exemplo: Ler um Ficheiro Binário

```
with open('exemplo.jpg', 'rb') as ficheiro:
    conteudo = ficheiro.read()
    print(conteudo[:10]) # Imprime os primeiros 10 bytes
```

Pontos importantes:

- conteudo é um objeto de bytes.
- Não é aplicada qualquer codificação durante a leitura.

Escrever Ficheiros Binários em Python

- Abra ficheiros binários com o modo ’wb’.
- Escreva bytes brutos diretamente no ficheiro.

Exemplo: Escrever num Ficheiro Binário

with open(’exemplo.bin’, ’wb’) as ficheiro:
ficheiro.write(b’\x48\x65\x6C\x6C\x6F’) # Escreve "Hello"em bytes

Pontos importantes:

- Sequências de bytes (b”) são escritas no ficheiro.
- O ficheiro será criado se não existir ou substituído se já existir.

Ficheiros Binários vs. Ficheiros de Texto

Diferenças principais:

- **Formato de Dados:**
  - Texto: Caracteres legíveis por humanos.
  - Binário: Dados brutos em bytes.
- **Modos de Ficheiro:**
  - Texto: 'r', 'w', 'a'
  - Binário: 'rb', 'wb', 'ab'
- **Codificação:**
  - Texto: Requer codificação (e.g., UTF-8).
  - Binário: Não aplica codificação/descodificação.
- **Utilizações:**
  - Texto: Logs, documentos, configurações.
  - Binário: Imagens, vídeos, executáveis.

Converter entre Texto e Binário

Texto para Binário:

```
texto = "Olá, Mundo!"
dados_binarios = texto.encode('utf-8') # Converte para bytes
print(dados_binarios) # Output: b'Olá, Mundo!'
```

Binário para Texto:

```
dados_binarios = b'Olá, Mundo!'
texto = dados_binarios.decode('utf-8') # Converte bytes para string
print(texto) # Output: Olá, Mundo!
```

Exercicio. Fala um programa para ler um ficheiro em binário, e escrever o seu conteúdo em hexadecimal.

Perguntas?