

Manipulação de Arquivos

Nesta aula, vamos explorar como manipular arquivos em Python. A manipulação de arquivos é uma habilidade importante, pois permite que você leia, escreva e manipule dados armazenados em arquivos no sistema de arquivos do computador. Vamos começar!

1. Abertura de Arquivos

Em Python, utilizamos a função `open()` para abrir arquivos. Esta função retorna um objeto arquivo, que nos permite interagir com o arquivo. Veja os modos mais comuns:

- `"r"`: Abre para leitura (modo padrão).
- `"w"`: Abre para escrita (cria um arquivo novo ou sobrescreve o existente).
- `"a"`: Abre para adicionar (anexa ao final do arquivo).
- `"b"`: Modo binário (por exemplo, para imagens ou outros arquivos não-textuais).

```
# Abrindo um arquivo para leitura  
arquivo = open('exemplo.txt', 'r')
```

Context Manager (com `with`)

A forma mais segura e eficiente de abrir arquivos em Python é usando o `with`, que garante que o arquivo seja fechado adequadamente após o uso.

```
# Forma recomendada de abrir um arquivo
with open('exemplo.txt', 'r') as arquivo:
    conteudo = arquivo.read()
    print(conteudo) # Exibindo o conteúdo do arquivo
```

Usar `with` elimina a necessidade de fechar o arquivo manualmente com `arquivo.close()`.

2. Leitura de Arquivos

Existem várias maneiras de ler o conteúdo de um arquivo em Python:

read() – Lê o arquivo inteiro 

```
with open('exemplo.txt', 'r') as arquivo:  
    conteudo = arquivo.read() # Lê todo o conteúdo  
    print(conteudo)
```

readline() – Lê uma linha por vez 

```
with open('exemplo.txt', 'r') as arquivo:  
    linha = arquivo.readline() # Lê a primeira linha  
    print(linha)
```

readlines() – Lê todas as linhas e as retorna como uma lista 

```
with open('exemplo.txt', 'r') as arquivo:
    linhas = arquivo.readlines() # Retorna uma lista de linhas
    for linha in linhas:
        print(linha)
```

3. Escrita em Arquivos

Para gravar em arquivos, usamos os modos `"w"` ou `"a"`. Lembre-se de que o modo `"w"` sobrescreve o arquivo, enquanto o `"a"` apenas adiciona ao final.

Escrevendo um texto novo

```
with open('exemplo.txt', 'w') as arquivo:  
    arquivo.write('Este é um novo conteúdo.  
' )
```

Adicionando texto sem sobrescrever

```
with open('exemplo.txt', 'a') as arquivo:  
    arquivo.write('Este texto foi adicionado ao final.  
' )
```

4. Fechando Arquivos ✖

Se não usar `with`, é importante fechar o arquivo manualmente com o método `close()` para liberar recursos.

```
arquivo = open('exemplo.txt', 'r')  
conteudo = arquivo.read()  
arquivo.close() # Fechando o arquivo
```


Conclusão 🏁

Hoje, aprendemos os conceitos fundamentais sobre como **abrir**, **ler** e **escrever** em arquivos utilizando Python. Manipular arquivos é uma habilidade crucial para interagir com sistemas de arquivos e armazenar dados de forma persistente.

- **Abertura de arquivos:** Utilizamos a função `open()` e os modos como `"r"`, `"w"`, `"a"` e `"b"`.
- **Leitura de arquivos:** Vimos as funções `read()`, `readline()` e `readlines()` para diferentes necessidades.
- **Escrita em arquivos:** Aprendemos a escrever e anexar conteúdo com `write()`.
- **Fechamento de arquivos:** Entendemos a importância de fechar arquivos corretamente com `close()` ou utilizando o `with` para automação do processo.