Gestão de Arquivo Python

O manuseio de arquivos é uma parte importante de qualquer aplicativo da web.

Python possui diversas funções para criar, ler, atualizar e excluir arquivos.

## **Manipulação de arquivos**

A função principal para trabalhar com arquivos em Python é a open()função.

A open()função leva dois parâmetros; nome do arquivo e modo .

Existem quatro métodos (modos) diferentes para abrir um arquivo:

"r"- Ler - Valor padrão. Abre um arquivo para leitura, erro se o arquivo não existir

"a"- Anexar - Abre um arquivo para anexar, cria o arquivo se ele não existir

"w"- Write - Abre um arquivo para gravação, cria o arquivo caso ele não exista

"x"- Criar - Cria o arquivo especificado, retorna um erro se o arquivo existir

Além disso vocêpoderáespecificar se o arquivo deverá ser tratado como modo binário ou texto

"t"- Texto - Valor padrão. Modo texto

"b"- Binário - Modo binário (por exemplo, imagens)

## **Sintaxe**

Para abrir um arquivo para leitura basta especificar o nome do arquivo:

f = open("demofile.txt")

O código acima é igual a:

f = open("demofile.txt", "rt")

Como "r"for read e "t"for text são os valores padrão, não é necessário especificá-los.

**Nota:** Certifique-se de que o arquivo exista, caso contrário você receberá um erro.

## **Abra um arquivo no servidor**

Suponha que temos o seguinte arquivo, localizado na mesma pasta do Python:

arquivo demo.txt

Hello! Welcome to demofile.txt  
This file is for testing purposes.  
Good Luck!

Para abrir o arquivo, use a função integrada open().

A open()função retorna um objeto de arquivo, que possui um read()método para ler o conteúdo do arquivo:

### **Exemplo**

f = open("demofile.txt", "r")  
print(f.read())

Se o arquivo estiver localizado em um local diferente, você terá que especificar o caminho do arquivo, assim:

### **Exemplo**

Abra um arquivo em um local diferente:

f = open("D:\\myfiles\welcome.txt", "r")  
print(f.read())

## **Leia apenas partes do arquivo**

Por padrão o read()método retorna o texto inteiro, mas você também pode especificar quantos caracteres deseja retornar:

### **Exemplo**

Retorne os 5 primeiros caracteres do arquivo:

f = open("demofile.txt", "r")  
print(f.read(**5**))

## **Ler linhas**

Você pode retornar uma linha usando o readline()método:

### **Exemplo**

Leia uma linha do arquivo:

f = open("demofile.txt", "r")  
print(f.readline())

Ligando readline()duas vezes, você pode ler as duas primeiras linhas:

### **Exemplo**

Leia duas linhas do arquivo:

f = open("demofile.txt", "r")  
print(f.readline())  
print(f.readline())

Ao percorrer as linhas do arquivo, você pode ler o arquivo inteiro, linha por linha:

### **Exemplo**

Percorra o arquivo linha por linha:

f = open("demofile.txt", "r")  
for x in f:  
  print(x)

## **Fechar arquivos**

É uma boa prática sempre fechar o arquivo quando terminar de usá-lo.

### **Exemplo**

Feche o arquivo quando terminar:

f = open("demofile.txt", "r")  
print(f.readline())  
f.close()

**Nota:** Você deve sempre fechar seus arquivos; em alguns casos, devido ao buffer, as alterações feitas em um arquivo podem não aparecer até que você feche o arquivo.

## **Gravar em um arquivo existente**

Para gravar em um arquivo existente, você deve adicionar um parâmetro à open()função:

"a"- Anexar - será anexado ao final do arquivo

"w"- Escrever - substituirá qualquer conteúdo existente

### **Exemplo**

Abra o arquivo "demofile2.txt" e anexe o conteúdo ao arquivo:

f = open("demofile2.txt", "a")  
f.write("Now the file has more content!")  
f.close()  
  
#open and read the file after the appending:  
f = open("demofile2.txt", "r")  
print(f.read())

### **Exemplo**

Abra o arquivo "demofile3.txt" e substitua o conteúdo:

f = open("demofile3.txt", "w")  
f.write("Woops! I have deleted the content!")  
f.close()  
  
#open and read the file after the overwriting:  
f = open("demofile3.txt", "r")  
print(f.read())

**Nota:** o método "w" substituirá o arquivo inteiro.

## **Crie um novo arquivo**

Para criar um novo arquivo em Python, use o open()método, com um dos seguintes parâmetros:

"x"- Criar - criará um arquivo, retornará um erro se o arquivo existir

"a"- Anexar - criará um arquivo se o arquivo especificado não existir

"w"- Write - criará um arquivo se o arquivo especificado não existir

### **Exemplo**

Crie um arquivo chamado "meuarquivo.txt":

f = open("myfile.txt", "x")

Resultado: um novo arquivo vazio é criado!

### **Exemplo**

Crie um novo arquivo se ele não existir:

f = open("myfile.txt", "w")

## **Excluir um arquivo**

Para excluir um arquivo, você deve importar o módulo do SO e executar sua os.remove()função:

### **Exemplo**

Remova o arquivo "demofile.txt":

import os  
os.remove("demofile.txt")

## **Verifique se o arquivo existe:**

Para evitar erros, verifique se o arquivo existe antes de tentar excluí-lo:

### **Exemplo**

Verifique se o arquivo existe e exclua-o:

import os  
if os.path.exists("demofile.txt"):  
  os.remove("demofile.txt")  
else:  
  print("The file does not exist")

## **Excluir pasta**

Para excluir uma pasta inteira, use o os.rmdir()método:

### **Exemplo**

Remova a pasta "minhapasta":

import os  
os.rmdir("myfolder")

**Nota:** Você só pode remover pastas vazias .