

ASIMOV

Arquivos

O Python usa objetos de arquivo para interagir com arquivos externos em seu computador. Esses objetos de arquivo podem ser qualquer tipo de arquivo que você tenha em seu computador, seja um arquivo de áudio, um arquivo de texto, e-mails, documentos do Excel, etc. Nota: provavelmente você precisará instalar certas bibliotecas ou módulos para interagir com esses vários tipos de arquivo, mas eles estão facilmente disponíveis. (Vamos abordar o download de módulos mais tarde no curso).

O Python possui uma função aberta aberta que nos permite abrir e utilizar métodos básicos com arquivos. Primeiro, precisamos de um arquivo. Usaremos uma mágica do iPython para criar um arquivo de texto!

Escrevendo um arquivo com iPython

```
In [1]: %%writefile test.txt
        Hello, this is a quick test file
```

Writing test.txt

Abrindo um arquivo com Python

Podemos abrir um arquivo com a função `open()`. A função `open` também possui argumentos (também chamados de parâmetros). Vamos ver como isso é usado:

```
In [2]: # Abre um arquivo txt já existente
        my_file = open('test.txt')
```

```
In [3]: # Agora podemos ler o arquivo
        my_file.read()
```

Out[3]: 'Hello, this is a quick test file'

```
In [4]: # Mas o que acontece se tentarmos lê-lo novamente?
        my_file.read()
```

Out[4]: ''

Isso acontece porque você pode imaginar que o "cursor" de leitura esteja no final do arquivo depois de ter lido. Portanto, não há nada a ler. Podemos redefinir o "cursor" assim:

```
In [8]: # Procure o início do arquivo (índice 0)
```

```
my_file.seek(0)
```

Out[8]: 0

```
In [6]: # Lê novamente  
my_file.read()
```

Out[6]: 'Hello, this is a quick test file'

Para não ter que reiniciar todas as vezes, também podemos usar o método `readlines`. Tenha cuidado com arquivos grandes, já que tudo será mantido na memória. Nós aprenderemos como iterar sobre arquivos grandes mais tarde no curso.

```
In [9]: # Readlines retorna uma lista das linhas no arquivo.  
my_file.readlines()
```

Out[9]: ['Hello, this is a quick test file']

Escrevendo um arquivo

Por padrão, usando a função `open()` só nos permitirá ler o arquivo, precisamos passar o argumento `'w'` para escrever sobre o arquivo. Por exemplo:

```
In [39]: # Adiciona um segundo argumento à função, 'w', que significa escrita  
my_file = open('test.txt', 'w')
```

```
In [40]: # Escreve no arquivo  
my_file.write('This is a new line')
```

```
In [43]: # Lê o arquivo  
my_file.read()
```

Out[43]: 'This is a new line'

Iterando através de um arquivo

Vamos dar uma pequena olhada sobre como iterar através do arquivo. Primeiro vamos criar um novo arquivo de texto:

```
In [44]: %%writefile test.txt  
First Line  
Second Line
```

Overwriting test.txt

Agora podemos usar um pouco de fluxo para dizer o programa para através de cada linha do arquivo e fazer algo:

```
In [45]: for line in open('test.txt'):  
    print(line)
```

First Line

Second Line

Não se preocupe em entender completamente isso ainda, pois os laços serão abordados em breve. Mas vamos quebrar o que fizemos acima. Nós dissemos que para cada linha neste arquivo de texto, vá em frente e imprima essa linha. É importante notar algumas coisas aqui:

- 1.) Poderíamos ter chamado o objeto 'linha' qualquer coisa (veja o exemplo abaixo).
- 2.) Ao não chamar `.read()` no arquivo, o arquivo de texto inteiro não foi armazenado na memória.
- 3.) Observe o recuo na segunda linha para impressão. Este espaço em branco é necessário em Python.

Aprenderemos muito mais sobre isso mais tarde.

In [46]:

```
for asdf in open('test.txt'):
    print(asdf)
```

First Line

Second Line