01

Métodos especiais

Transcrição

[00:00] Sejam bem-vindos a mais essa aula do nosso curso de Python. Nessa aula, quero mostrar o que são métodos especiais. São coisas muito interessantes e que podem dar funcionalidades muito bacanas para as nossas classes.

[00:16] Antes disso, quero fazer uma afirmação. Quero dizer que tudo em Python é uma instância de alguma classe maior. Por exemplo, no momento em que você cria string = “bytebank”, você está criando uma variável do tipo str, que acessa essa classe str. Como consequência, pode usar seus métodos, que foi o que fizemos ao longo do curso. Usamos o método find, lower, replace. Todos dessa classe str.

[00:50] Se você está usando o PyCharm, existe uma forma bem simples de entrar nessa classe. Eu digito str, pressiono ctrl e clico no str. Perceba que já fui jogado para um arquivo totalmente diferente. É um arquivo de built-in.py.

[01:05] Nós já comentamos que tudo que é built-in vem junto com a instalação do Python, como o método re, que usamos na última aula. Aqui tem uma descrição rápida do que é essa classe. E depois temos os métodos.

[01:25] Vamos dar uma olhada no capitalize. Ele diz que faz o primeiro caracteres ser maiúsculo e todo o resto ser minúsculo. É como se fosse um título. Vamos procurar agora o find. Ele diz que o find retorna o valor mais baixo onde uma subString é encontrada. Foi exatamente o que fizemos. Podemos ver até aquele argumento que usamos para conseguir usar o find para chamar a moeda destino, e outro argumento sobre até onde a busca pode ir.

[02:25] Encontramos também o lower aqui. Outro método que nós usamos. Vamos procurar pelos métodos especiais da classe str. Vou dizer que o método especial começa com dois underlines e termina com dois underlines.

[02:39] Esses métodos especiais dão funções para as nossas classes. Vou destacar o método len: print (len(string)). Ele me diz que bytebank possui oito caracteres. Agora, quero criar uma variável do tipo inteiro: x = 200. Vou dar um print nisso: print(len(X)). Será que ele vai dizer que é igual a três, por ter três caracteres? Não funcionou. Ele me diz que objetos do tipo inteiro não possuem a função len.

[03:43] Vamos dar uma olhada nessa classe int e quais métodos ela possui. Quero procurar aqui dentro o método len. Ele não tem. Repare que na classe str existe o método especial len, eu usei função len e o string e funcionou, retornou o tamanho. Já a classe int não tem o método especial len. Eu usei e deu erro.

[04:35] A boa notícia é que posso criar um método especial len dentro da nossa classe, da classe que estamos criando, para dar alguma funcionalidade no momento em que tentamos extrair o comprimento da nossa classe.

[04:52] Vamos voltar para o main. Eu tenho já o argumentosUrl, que é uma instância da nossa classe. Vou tentar printar o tamanho dela: print (len(argumentosUrl)). Rodando isso, ele me fala que objeto do tipo ExtratorArgumentosUrl não possuem len. O mesmo erro que tivemos no inteiro.

[05:18] Vou abrir nossa classe e vou ter que retornar alguma coisa para o Python saber o que eu considero como tamanho da minha classe. def__len__(self): return 10

[05:41] Se eu rodar agora, ele printa 10. Alguma coisa deu erro, porque eu não tinha o tipo len na minha classe. Agora não dá mais erro. Ele retorna alguma coisa. Mas faz sentido o tamanho da minha classe ser 10? O que isso significa? Não tem 10 aqui. Minha url não tem 10 caracteres somente. Tem muito mais.

[06:04] Eu quero que o tamanho da minha classe seja o tamanho do atributo url. Se tivesse algum outro atributo eu poderia decidir entre qual eu quero usar para representar o tamanho, mas como só tem, uso simplesmente: def__len__(self): return len(self.url)

[06:25] Agora ele retornou 74. Dentro da url tem 74 caracteres. Se eu apagar, ele me mostra 73. Está funcionando perfeitamente. Estamos utilizando uma função do Python para fazer alguma coisa com a nossa classe, e tudo isso graças ao método especial.

[06:49] Atentem-se a um detalhe. Olhando para esse init e para esse erro que nós retornamos, eu vejo que o LookupError talvez não seja o mais adequado para identificar o que está acontecendo no momento em que nossa url é inválida.

[07:02] LookupError é um erro de pesquisa. Estamos tendo um erro de passagem de argumento.

[07:09] Quero mostrar uma coisa para vocês na documentação do Python. Vamos em built-in Exceptions. Todas as exceções que o Python possui. Vamos procurar a que estamos usando, LookupError. Esse erro serve para quando estou passando um índice inválido. Não é o que está acontecendo com a gente. Estamos tendo um erro de passagem de argumento.

[07:56] Para essa aula não ficar muito comprida, vou deixar o Lookup, mas quero que vocês saibam que existem muitos erros que estão dentro da biblioteca do Python, da documentação do Python, e que você pode encontrar e aplicar o que for melhor para sua classe, para o seu tipo de erro.

[08:12] Na próxima aula vamos ver uma forma de representar melhor nossa classe. Quando tento dar um print (argumentosUrl), ele me mostra algo bem esquisito. Na próxima aula vamos resolver isso.