

O método find

Transcrição

[00:00] Lembram que o desafio que eu propus na aula passada foi conseguir capturar de forma dinâmica alguma subString dentro da nossa URL? Quando eu digo dinâmica, quero dizer que quero pegar um pedaço específico, independente se tem algo antes ou não, e do que tem antes ou não.

[00:18] Eu falei que o Python possui um método bem bacana para isso. Vou voltar para o argumento = 'moedaorigem=real'. Supondo que eu queira somente a palavra real, somente esse valor do meu argumento, nós sabemos fazer o fatiamento, mas o problema é que se eu colocar qualquer coisa antes, teríamos que ir ao código e mudar os índices do meu fatiamento.

[01:05] Quero apresentar para vocês um método de string chamado find. Como o nome já diz, ele encontra alguma coisa. Posso encontrar algum caractere que eu queira dentro da minha string argumento: `index = argumento.find ()`

[01:25] Para achar o real, é bem simples. Ele vem logo depois do símbolo de igual. Então, peço para ele encontrar o real: `index = argumento.find ("=")`. O nome index não foi coincidência. O método find retorna o índice do caractere buscado dentro de uma string. Caso ele não encontre, ele retorna -1, para nos indicar que não existe.

[02:03] Agora, vamos usar esse índice dentro do nosso fatiamento: `subString = argumento [index:]`

[02:30] Printando isso, ele me mostra =real. Lembram que o primeiro item do fatiamento é inclusivo e o segundo é exclusivo? Preciso colocar um + 1: `subString = argumento [index + 1:]`

[02:50] Agora consigo pegar simplesmente o real. O M está na posição 11. Quando coloco o +1, começo no índice 12, pegando do R em diante.

[03:12] Assim, eu eliminei aquele problema de ter um prefixo. Se eu colocar o ByteBank de novo, o endereço de câmbio, e rodar, ele somente vai nos dar o real. Meu código agora procura o igual. Retorna o índice para mim e eu passo esse índice para o fatiamento.

[03:35] Mas tem outra forma de resolver esse problema com outra função bem legal do Python, que é o split. Ela transforma uma string em um vetor, recebe de argumento um separador. Em alguns casos, esse método pode ser bem eficaz.

[03:56] Eu não vou mais receber o índice. Vou receber uma lista: `listaargumentos = argumento.split("=")`. O que separa o nome do meu argumento do valor do meu argumento, é o sinal de igual. Printando isso, ele cria um vetor e colocou uma string com um nome e uma string com o valor.

[04:40] Essa função é bem bacana. Para casos em que o separador é bem definido e que a string não é tão complexa, ela é muito boa. Mas quando a string fica mais complexa, com mais argumentos, com mais separações não tão específicas, acaba não sendo tão útil.

[05:05] Por enquanto é só. Espero que vocês façam os exercícios, e qualquer dúvida, não hesite em perguntar no fórum. Todo mundo vai estar disposto a responder e te ajudar. Espero vocês na próxima aula.