BLOG

Postado em Python, em 27 de dezembro de 2021

TRY E EXCEPT NO PYTHON – TRATAMENTO DE ERROS NO PYTHON

Essa é uma aula muito interessante, pois eu quero te mostrar como fazer o tratamento de erros utilizando Try e Except no Python.

Caso prefira esse conteúdo no formato de vídeo-aula, assista ao vídeo abaixo ou acesse o nosso canal do YouTube!

Para receber por e-mail o(s) arquivo(s) utilizados na aula, preencha:

Seu melhor e-mail

Enviar

Try e Except no Python

Você provavelmente já deve ter visto alguns erros no Python enquanto você estava executando o seu código, não é mesmo?

Então eu preparei essa aula para te mostrar como tratar erros no Python para evitar com que esses erros apareçam para o seu usuário final.

Tratamento de Erros no Python

Os erros no Python ou em qualquer outra linguagem de programação existem e vão na verdade te dar uma informação do motivo daquele erro.

Dessa forma você pode fazer o seu tratamento ou modificar o seu código para que esse erro não aconteça novamente.

```
print(texto)

---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_27372/3642110558.py in <module>
----> 1 with open('aruivo.txt', 'r') as file_object:
      2     texto = file_object.read()
      3     print(texto)

FileNotFoundError: [Errno 2] No such file or directory: 'aruivo.txt'
```

Tentativa de abrir um arquivo com a estrutura with

Nesse caso nós estamos tentando ler arquivo no Python, só que aqui nós temos o erro **FileNotFoundError**, ou seja, erro de arquivo não encontrado.

Isso quer dizer que por algum motivo não foi possível encontrar esse arquivo. Neste caso é porque colocamos o nome diferente, então de fato não existe.

A ideia é tratar esse erro para que o usuário não tenha a visualização desse erro, pois algumas vezes a mensagem de erro não é muito intuitiva.

Por esse motivo é que nós vamos utilizar o **Try** e **Except** para "tratar" esse erro e fazer com que o usuário não visualize esse erro.

Lendo o arquivo:

```
In [6]: try:
            with open('aruivo.txt', 'r') as file_object:
                texto = file_object.read()
                print(texto)

        except:
            print('Deu ruim')

        Deu ruim
```

Tratando o erro com Try e Except

Nesse caso o **try** é para que o Python tente executar aquele código e caso não consiga executar por conta de um erro ele vai retornar o que temos dentro do **except**.

Dessa maneira você pode personalizar o que será mostrado ao usuário ao invés de uma mensagem de erro grande que pode ser difícil de entender do que se trata.

OBS: Caso você queira verificar todos os erros pode clicar aqui para ser direcionado a página do Python onde vai conseguir visualizar esses possíveis erros.

Vamos agora para os tipos de erros no Python e daremos um exemplo que como tratá-los. Claro que cada erro você pode fazer um tratamento diferente para que se adeque ao seu trabalho.

FileNotFoundError:

```
try:
    with open('aruivo.txt', 'r') as file_object:
        texto = file_object.read()
        print(texto)


except FileNotFoundError as error:
    print(error)

except:
    print('Deu ruim')

[Errno 2] No such file or directory: 'aruivo.txt'
```

Tratamento de um erro específico

Nesse caso nós vamos ter um tratamento específico para o erro de arquivo não encontrado e caso aconteça outro tipo de erro nós vamos retornar a mensagem "Deu ruim".

ZeroDivisionError:

```
a = 10
b = 0

try:
    print(a/b)

except ZeroDivisionError as error:
    print(error)
```

```
division by zero
```

Erro de divisão por zero

O próximo erro que vamos abordar é o **ZeroDivisionError**, que é o erro de divisão por zero.

Novamente você pode fazer um tratamento especial para o erro, pode retornar uma mensagem para o usuário informando que o valor não pode ser zero, ou qualquer outra informação que o ajude.

O último exemplo que vamos tratar não é um erro novo, mas uma maneira de tratar esses erros.

Else e Finally:

```
In [32]: a = 10
         b = 0

         try:
             print(a/b)

         except ZeroDivisionError as error:
             print(error)

         else:
             print( ' Sem erros')

         finally:
             print('Aqui sempre vai printar')
```

```
division by zero
Aqui sempre vai printar
```

Else e Finally para o tratamento de erros

Aqui nós estamos tentando fazer uma divisão. Caso seja encontrado o erro de divisão por zero vamos rodar o código **except**, caso contrário ele vai passar direto e rodar o código dentro do **else**.

Só que independente de termos ou não um erro o código dentro do **finally** será executado, então se estiver abrindo uma página e deu erro, você vai querer fechar essa página.

Ou mesmo se conseguiu executar o que precisa, você pode querer fechar essa página para que não fique com várias páginas abertas por exemplo.

Conclusão do Try e Except no Python

Nessa aula eu te mostrei de forma rápida como você pode fazer o tratamento de erros no Python para evitar com que esse erro apareça para o seu usuário final.

A ideia é evitar com que o usuário veja aquela mensagem grande de erro ou até mesmo uma mensagem que ele não entenda.

Então o objetivo é tratar esse erro de forma com que o usuário repita um procedimento, altere alguma informação, recebe um texto ou um aviso detalhado do problema para que ele saiba o que aconteceu de fato!

Hashtag Treinamentos

Para acessar outras publicações de Python, clique aqui!

Quer aprender mais sobre Python com um minicurso gratuito?

Preencha com o seu e-mail*

Seu melhor e-mail

QUERO ME CADASTRAR

Heitor Catunda

Expert em conteúdos da Hashtag Treinamentos. Auxilia na criação de conteúdos de variados temas voltados para aqueles que acompanham nossos canais.