Usando variavéis e condicionais

**Docstring e metadados dunder**

Em todo script Python é uma boa prática incluir um comentário de multi linhas logo nas primeiras linhas do script explicando o objetivo do script e provendo documentação para o usuário.

#!/usr/bin/env python3

"""Hello World Multi Linguas.

Dependendo da lingua configurada no ambiente o programa exibe a mensagem

correspondente.

Como usar:

Tenha a variável LANG devidamente configurada ex:

export LANG=pt\_BR

Execução:

python3 hello.py

ou

./hello.py

"""

\_\_version\_\_ = "0.0.1"

\_\_author\_\_ = "Bruno Rocha"

\_\_license\_\_ = "Unlicense"

E além do comentário de documentação, chamado **DocString** é também comum a inclusão de variavéis de metadados que inician e terminam com 2 underlines \_\_ a palavra que usamos para designar essas variavéis é Dunder portanto, **Dunder version** se refere a \_\_version\_\_.

**Variavéis de Ambiente**

**Ambiente** é o termo que usamos para referir ao local onde o programa é executado, o ambiente em termos gerais é formado por um **shell** que em termos bem simplificados pode ser entendido como um local isolado onde o seu programa executa.

Neste **ambiente** existem variavéis que servem para configurar o comportamento do próprio ambiente, do sistema e dos programas que rodam.

No linux ao digitar env no terminal você verá uma lista de todas as variáveis do ambiente, ali por exemplo está o nome do usuário em USER e a linguagem em que o sistema operacional está configurado LANG.

Podemos dentro do Python acessar essas variáveis de ambiente atráves do módulo os.

import os

os.environ # um dicionário Python contendo as variavéis e seus valores

os.getenv("LANG") # função usada para buscar o valor de uma variável especifica.

**Obtendo uma parte de um texto (substring)**

Em nosso primeiro programa irá ler a variável de ambiente LANG e apartir dela decidir qual mensagem de Hello World imprimir na tela, portanto se for pt\_BR imprimimos Olá, Mundo! se for it\_IT será Ciao, Mondo! e assim por diante.

O problema é que o comando os.getenv("LANG") vai retornar algo como en\_US.utf8 e nós queremos apenas a primeira parte antes do ., nesse caso queremos en\_US ignorando a parte .utf8 e existem algumas formas de obter isso em Python.

# Variável completa

>>> os.getenv("LANG")

'en\_US.utf8'

# Apenas os 5 primeiros caractered

>>> os.getenv("LANG")[:5]

'en\_US'

# Com um valor default em caso da variável não existir

>>> os.getenv("LANG", "en\_US")[:5]

'en\_US'

# Através do aplit do texto

>>> os.getenv("LANG", "en\_US").split(".")[0]

'en\_US'

Em nosso programa vamos por enquanto escolher a versão

>>> os.getenv("LANG", "en\_US")[:5]

'en\_US'

Iremos explorar o significado de [:5] logo mais quando falarmos de iteração de objetos :)

Portanto nosso script agora contém:

#!/usr/bin/env python3

"""Hello World Multi Linguas.

Dependendo da lingua configurada no ambiente o programa exibe a mensagem

correspondente.

Como usar:

Tenha a variável LANG devidamente configurada ex:

export LANG=pt\_BR

Execução:

python3 hello.py

ou

./hello.py

"""

\_\_version\_\_ = "0.0.1"

\_\_author\_\_ = "Bruno Rocha"

\_\_license\_\_ = "Unlicense"

import os

current\_language = os.getenv("LANG", "en\_US")[:5]

msg = "Hello, World!"

print(msg)

E agora em posse do valor current\_language podemos usar as condicionais para decidir qual mensagem iremos exibir na tela :)

**Condicionais**

Em nosso programa precisamos usar um valor de uma variável current\_language para tomar uma decisão de qual mensagem exibir na tela, decisões que alteram o fluxo de execução do código são chamadas de "Condicionais".

Em Python o statement if é usado para definir um teste e sempre usamos if junto com uma expressão de comparação como por exemplo 1 > 2 e neste caso teriamos a condicional: se 1 for maior que 2 (execute determinada instrução) traduzindo isso para código seria:

if 1 > 2:

# faça algo aqui

Portanto agora podemos alterar nosso script e adicionar a condicional que irá alterar o valor de msg dependendo da linguagem definida no ambiente.

if current\_language == "pt\_BR":

msg = "Olá, Mundo!"

elif current\_language == "it\_IT":

msg = "Ciao, Mondo!"

elif current\_language == "fr\_FR":

msg = "Bonjour, Monde!"

Repare que usamos == para determinar uma expressão de comparação e usamos elif para adicionar mais condições ao bloco de código.

E encaixando isso no script.

#!/usr/bin/env python3

"""Hello World Multi Linguas.

Dependendo da lingua configurada no ambiente o programa exibe a mensagem

correspondente.

Como usar:

Tenha a variável LANG devidamente configurada ex:

export LANG=pt\_BR

Execução:

python3 hello.py

ou

./hello.py

"""

\_\_version\_\_ = "0.0.1"

\_\_author\_\_ = "Bruno Rocha"

\_\_license\_\_ = "Unlicense"

import os

current\_language = os.getenv("LANG", "en\_US")[:5]

msg = "Hello, World!"

# Aqui aplicamos a condicional

if current\_language == "pt\_BR":

msg = "Olá, Mundo!"

elif current\_language == "it\_IT":

msg = "Ciao, Mondo!"

elif current\_language == "fr\_FR":

msg = "Bonjour, Monde!"

print(msg)

Execute o script no terminal usando

LANG=it\_IT python3 hello.py

E experimente substituir o valor de LANG para testar a funcionalidade do script.

Pode também persistir o valor exportando a variável para o ambiente antes de executar:

export LANG=fr\_FR

python3 hello.py

**Commit**

Agora que o código está funcionando é importante adicionar ao histórico do git.

# veja a lista de alterações a serem aplicadas

git status

# adicione o script

git add hello.py

# faça o commit

git commit -m "Implementei o script multi linguas"

# mande para o repositório do Github

git push

:)