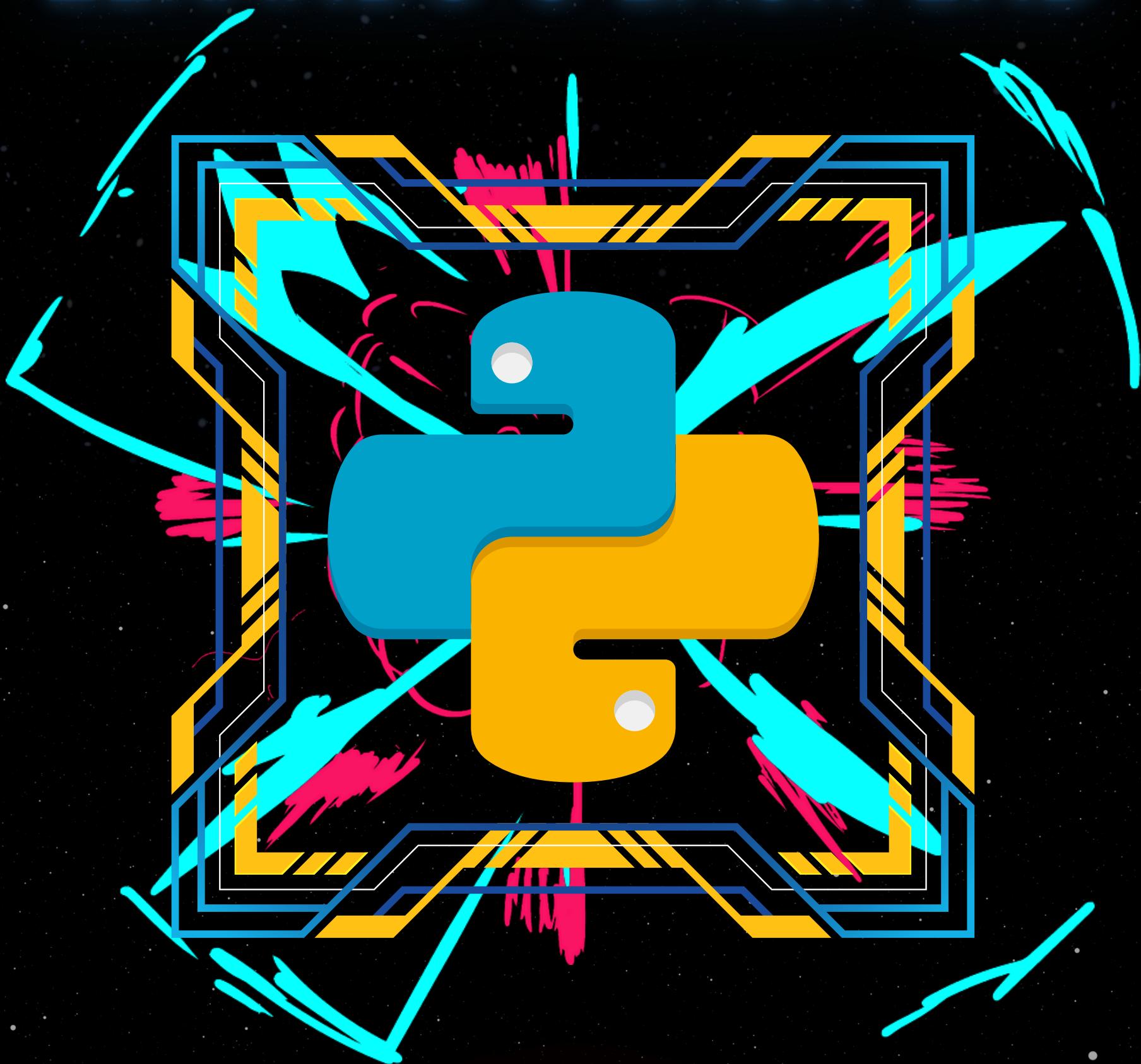


PYTHON

ZERANDO O BACK-END



Aprenda os fundamentos essenciais
de Python para back-end developer

JENNIFER SANTIAGO

FUNDAMENTOS ESSENCIAIS

Python para Desenvolvimento Back-end

Python é uma das linguagens de programação mais populares e versáteis do mundo, conhecida por sua simplicidade e legibilidade. Antes de mergulharmos no desenvolvimento back-end com frameworks como Django e Flask, é essencial dominar os fundamentos da linguagem. Neste ebook, abordaremos os principais conceitos do Python, acompanhados de exemplos de código para facilitar a compreensão.





VARIÁVEIS E TIPOS DE DADOS

Variáveis são usadas para armazenar informações. A linguagem suporta vários tipos de dados, incluindo inteiros, floats, strings e booleanos.

Variáveis

Variáveis são espaços na memória onde podemos armazenar dados. Em Python, você não precisa declarar o tipo da variável explicitamente.

```
Untitled-1

# Exemplo de uso de variáveis
nome = "João"
idade = 25
altura = 1.75

print(f"Nome: {nome}, Idade: {idade}, Altura: {altura}")
```



Tipos de Dados Comuns

Os principais tipos de dados em Python são: int (inteiro), float (decimal), str (string) e bool (booleano).

```
... Untitled-1

# Exemplos de tipos de dados
inteiro = 10
decimal = 3.14
texto = "Olá, Mundo!"
booleano = True

print(f"Int: {inteiro}, Float: {decimal}, Str: {texto}, Bool: {booleano}")
```





ESTRUTURAS DE CONTROLE DE FLUXO

As estruturas de controle de fluxo permitem que você controle a execução do código com base em condições.

Estruturas de Controle de Fluxo

Condicionais (if, elif, else)

As estruturas condicionais permitem que você tome decisões no seu código.

```
Untitled-1

# Exemplo de estrutura condicional
idade = 18

if idade ≥ 18:
    print("Você é maior de idade.")
else:
    print("Você é menor de idade.")
```



Estruturas de Controle de Fluxo

Laços de Repetição (for, while)

Laços são usados para executar um bloco de código várias vezes.

```
Untitled-1

# Exemplo de laço for
for i in range(5):
    print(f"Contagem: {i}")

# Exemplo de laço while
contagem = 0
while contagem < 5:
    print(f"Contagem: {contagem}")
    contagem += 1
```





ESTRUTURAS DE DADOS

Python oferece várias estruturas de dados embutidas, como listas, tuplas, dicionários e conjuntos.

Listas

Listas são coleções ordenadas e mutáveis de elementos.



The screenshot shows a code editor window titled "Untitled-1". In the top left corner, there are three small circular icons. The code itself is a Python script demonstrating list manipulation:

```
# Exemplo de uso de lista
frutas = ["maçã", "banana", "laranja"]
frutas.append("uva")
print(frutas)
```



Tuplas

Tuplas são coleções ordenadas e imutáveis de elementos.



Untitled-1

```
# Exemplo de uso de tupla
coordenadas = (10, 20)
print(coordenadas)
```



Dicionários

Dicionários são coleções desordenadas de pares chave-valor.

```
Untitled-1

# Exemplo de uso de dicionário
aluno = {
    "nome": "Maria",
    "idade": 22,
    "curso": "Engenharia"
}

print(aluno["nome"])
```





FUNÇÕES

Funções são blocos de código que realizam uma tarefa específica e podem ser reutilizadas.

Funções

Funções são blocos de código reutilizáveis que ajudam a modularizar e organizar o código.

```
Untitled-1

def saudacao(nome):
    return f"Olá, {nome}!"

print(saudacao("Maria"))

# Funções com parâmetros padrão
def somar(a, b=5):
    return a + b

print(somar(3))
print(somar(3, 7))
```





TRABALHANDO COM ARQUIVOS

Manipular arquivos é uma tarefa comum no desenvolvimento back-end, e Python facilita esse processo.

Manipulação de Arquivos

Ler e escrever arquivos é uma tarefa comum em programação.

```
Untitled-1

# Exemplo de escrita em arquivo
with open("exemplo.txt", "w") as arquivo:
    arquivo.write("Olá, Mundo!")

# Exemplo de leitura de arquivo
with open("exemplo.txt", "r") as arquivo:
    conteúdo = arquivo.read()
    print(contento)
```





MÓDULOS E PACOTES

Para organizar melhor o código e reutilizar funcionalidades, você pode dividir seu código em módulos e pacotes.

Importando Módulos

Módulos são arquivos que contêm código Python. Você pode importar módulos para reutilizar funções e variáveis.

```
Untitled-1

# Exemplo de importação de módulo
import math

print(math.sqrt(16))
```



Criando e Usando Pacotes

Pacotes são coleções de módulos. Eles ajudam a organizar o código em projetos maiores.

```
Untitled-1

# Estrutura de um pacote
# meu_pacote/
#     __init__.py
#     modulo1.py

# Conteúdo de modulo1.py
def funcao_modulo1():
    print("Função do módulo 1")

# Importando e usando o módulo
from meu_pacote.modulo1 import funcao_modulo1

funcao_modulo1()
```





MANIPULAÇÃO DE ERROS

Tratar erros de forma adequada é crucial para o desenvolvimento de aplicativos robustos.

Try, Except

Tratar erros de forma adequada é crucial para o desenvolvimento de aplicativos robustos.

```
Untitled-1

# Exemplo de tratamento de exceção:

try:
    resultado = 10 / 0
except ZeroDivisionError:
    print("Erro: divisão por zero!")
finally:
    print("Operação concluída.")
```



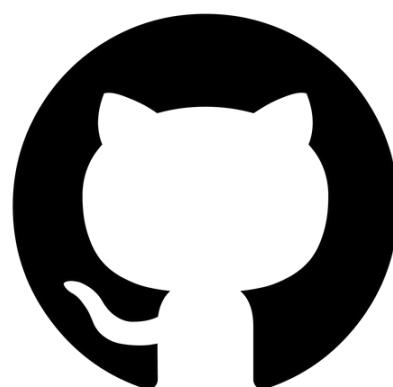
AGRADECIMENTOS

OBRIGADA POR LER ATÉ AQUI!

Este ebook foi gerado por IA, diagramado e editado por humano.

O conteúdo foi criado para fins didáticos, todas as informações são verídicas.

Você pode encontrá-lo no meu GitHub:



<https://github.com/Jennsaantiago/Ebook-Python-zerando-o-back-end>

Autora



Jennifer Santiago

[GitHub](#)

[LinkedIn](#)

[Instagram](#)

