PYTHON ACADEMY

DICT COMPREHENSION (COMPREENSÃO DE DICT) NO PYTHON

Nesse ebook, vamos aprender uma ferramenta avançada muito útil do Python para tratamento e manipulação de dicionários!

PYTHONACADEMY.COM.BR

Fala grande **Dev**!

Dict Comprehensions é uma ferramenta muito útil e poderosa do Python para manipulação de dicionários (dicts)! Com essa técnica, podemos tratar dicionários de uma maneira rápida e concisa!

Essa técnica segue o mesmo conceito das mais conhecidas *List Comprehensions*!

Ainda não sabe manusear listas usando List Comprehensions? 😱

Então P-A-R-A tudo e vai lá no post sobre List Comprehensions. Fico te aguardando voltar!

… Tô esperando …

Pronto! Agora podemos continuar! 😉

O início de tudo: o `dict`

Antes de tudo, vamos falar rapidamente sobre o tipo de dado `dict`. (Se você já sabe, pode pular essa seção 😉)

Dicionário em Python é uma coleção de dados sem ordem onde cada elemento possui um par chave/valor.

E o que é um par chave/valor?

Basicamente é uma forma de se indexar um valor a partir de uma chave.

Isso dá acesso eficiente (pros puristas, temos **O(1)** aqui!) aos valores da estrutura de dados.

A sintaxe para **criar** dicionários é a seguinte:

```python
# Dicionário vazio
dicionario = {}

# Dicionário comum
dicionario = {'jedi': 10, 'sith': 7}

# Dicionário com chaves inteiras
dicionario = {1: 'Baby Yoda', 2: 'Yoda'}

# Dicionário misturado
dicionario = {'especie': 'Humano', 1: ['Obi Wan Kenobi', 'Qui-Gon
        Jinn']}

# Outra forma de criação, usando dict()
dicionario = dict({'jedi': 10, 'sith': 7})
```

Já para **acessar** elementos:

```python
dicionario = {'nome': 'Vinícius Ramos', 'idade': 29}

# Saída: Vinícius
print(dicionario['nome'])

# Saída: 29
print(dicionario.get('idade'))

# Caso não encontre, devolva o valor None
print(dicionario.get('altura', None))

# Será lançada uma exceção KeyError
print(dicionario['endereco'])
```

Para **atualizar** valores:

```
dicionario = {'nome': 'Vinícius Ramos', 'idade': 29, 'empresa': 'Pytho-
        nAcademy'}

# Atualiza dados
dicionario['idade'] = 30
dicionario['empresa'] = 'PythonAcademy Inc.'
```

Para **remover** elementos:

```
dicionario = {'nome': 'Vinícius Ramos', 'idade': 29, 'empresa': 'Pytho-
        nAcademy'}

# Remove a chave/valor 'idade': 29
dicionario.pop('idade')

# Remove um par aleatório
dicionario.popitem()

# Remove todos os itens
dicionario.clear()

# Deleta 'empresa'
del dicionario['empresa']
```

Pronto! Você agora é um expert em Dicionários!

Agora vamos pro prato principal: *Dict Comprehension*!

Antes que eu me esqueça! Se não é inscrito no canal, curta, dê um jóinh…..

Ops! Canal errado! 😆

Dict Comprehensions

Dict Comprehensions foram introduzidas na linguagem através da especificação PEP 274.

Sua sintaxe básica é:

```
{chave: valor for elemento in iteravel}
```

Agora respira que vamos entender cada ponto:

- `chave`: será a chave de cada elemento do dicionário resultante.
- `valor`: valor daquela chave.
- `elemento`: é a unidade de iteração do iterável `iterável` (se for uma lista, por exemplo, `elemento` irá receber o valor iteração à iteração)
- `iteravel`: conjunto de dados que estão sendo iterados (pode ser uma lista ou um `set`, por exemplo)

Pra esclarecer, vamos à um exemplo:

```
dicionario = {elemento: elemento*2 for elemento in range(6)}
```

Aqui, cada elemento da lista resultante de `range(6)` (0, 1, 2, 3, 4, 5) será convertido em:

- Uma chave com o mesmo valor do `elemento` da lista.
- `elemento*2` é o valor de cada chave (multiplicar por 2 cada elemento).

O resultado será:

```
{0: 0, 1: 2, 2: 4, 3: 6, 4: 8, 5: 10}
```

Outro exemplo, com chaves alfabéticas e manipulação de strings com *f-strings*:

```
lista = ['Ferrari', 'Lamborghini', 'Porsche']
dicionario = {
  f'{elemento.lower()}': f'Montadora: {elemento.upper()}' for elemento
        in lista
}
```

Resultando em:

```
{
  'ferrari': 'Montadora: FERRARI',
  'lamborghini': 'Montadora: LAMBORGHINI',
  'porsche': 'Montadora: PORSCHE'
}
```

Também é possível iterar sobre um outro dicionário através do método `items()`.

Ele retorna a chave e o valor de cada elemento do dicionário de entrada.

Veja um exemplo:

```python
import locale

# Configura o locale pra Português do Brasil (pt_BR)
locale.setlocale(locale.LC_MONETARY, 'pt_BR.utf8')

carros_esportivos = {
  'ferrari': 1299000,
  'lamborghini': 1100000,
  'porsche': 759000
}

dict_saida = {
  chave: f'{chave.upper()}: {locale.currency(valor)}' for chave, valor
        in carros_esportivos.items()
}
```

Essa seria a saída:

```python
{
  'ferrari': 'FERRARI: R$ 1299000,00',
  'lamborghini': 'LAMBORGHINI: R$ 1100000,00',
  'porsche': 'PORSCHE: R$ 759000,00'
}
```

Legal não é mesmo?!

Assim como acontece em *List Comprehensions*, também podemos adicionar lógica condicional (`if` / `else`).

> 💡 *Estou desenvolvendo o **DevBook**, uma plataforma que usa IA para gerar ebooks técnicos profissionais. Te convido a conhecer!*

Dict Comprehensions com if

Podemos adicionar lógica condicional à construção do dicionário resultante do nosso *Dict Comprehension*.

Podemos adicionar uma expressão condicional em três posições distintas:

Na construção da **chave**. Sintaxe:

```
{chave if condicao: valor for elemento in iteravel}
```

Na expressão que definirá o valor da chave:

```
{chave: valor if condicao for elemento in iteravel}
```

Ao final da expressão, filtrando os dados do iterável:

```
{chave: expressao for elemento in iteravel if condicao}
```

Podendo ser um **mix** da primeira versão, segunda ou terceira (***Aí o bagulho fica lôco!***).

Por exemplo, se quisermos filtrar o dicionário de carros esportivos acima, pegando apenas aqueles com valor superior à R$ 1.000.000,00?

```
import locale

# Configura o locale pra Português do Brasil (pt_BR)
locale.setlocale(locale.LC_MONETARY, 'pt_BR.utf8')

carros_esportivos = {
  'ferrari': 1299000,
  'lamborghini': 1100000,
  'porsche': 759000
}

dict_saida = {
  chave: f'{chave.upper()}: {locale.currency(valor)}'
  for chave, valor in carros_esportivos.items() if valor > 1000000
}
```

Resultaria em:

```
{
  'ferrari': 'FERRARI: R$ 1299000,00',
  'lamborghini': 'LAMBORGHINI: R$ 1100000,00'
}
```

Dict Comprehensions com vários *if*

E se quisermos alterar a chave e o valor na mesma expressão?

Podemos fazer da seguinte forma:

```
import locale

# Configura o locale pra Português do Brasil (pt_BR)
locale.setlocale(locale.LC_MONETARY, 'pt_BR.utf8')

carros_esportivos = {
  'ferrari': 1299000,
  'lamborghini': 1100000,
  'porsche': 759000
}

dict_saida = {
  chave if valor > 1000000 else f'{chave}-valor-abaixo':
    f'{chave.upper()}: {locale.currency(valor)}'
    if valor > 1000000 else f'{chave.upper()}: Valor abaixo de R$
        1.000.000,00'
  for chave, valor in carros_esportivos.items()
}
```

Nesse exemplo:

- Chave será `f'{chave}-valor-abaixo'` caso `valor` seja menor que 1000000.
- Valor será `f'{chave.upper()}: Valor abaixo de R$ 1.000.000,00'` caso `valor` seja menor que 1000000.

O resultado seria:

```
{
  'ferrari': 'FERRARI: R$ 1299000,00',
  'lamborghini': 'LAMBORGHINI: R$ 1100000,00',
  'porsche-valor-abaixo': 'Valor abaixo de R$ 1.000.000,00'
}
```

Contudo, aqui vai uma dica.

Percebeu o quão "embolado" ficou o código?

Não é por que o Python possibilite isso, que seja uma boa ideia utilizá-lo dessa forma.

Nunca se esquecam do primeiro Zen do Python: "Bonito é melhor que feio".

Ainda não conhece o explicativo do "Zen do Python" da Python Academy? 😱

Conheça o Zen of Python clicando aqui) e se possível, o tatue no braço.

Brincadeira, só tatuá-lo já serve! 😆

Conclusão

Nesse *post* vimos como podemos usar *dict comprehensions* para criar e manipular dicts de uma maneira poderosa e eficiente.

Vimos quão eficiente essa técnica pode ser e as diversas formas de utilizá-la.

Agora que você está craque em *Dict Comprehensions*, ***que tal começar a utilizá-lo?***

Então… **Mão na massa**! 💪 💪

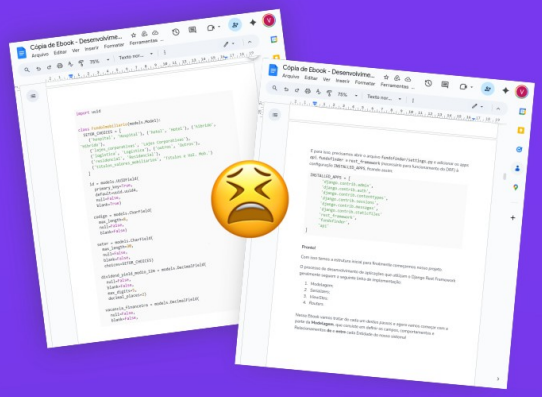
Até o próximo *post*!

Não se esqueça de conferir!

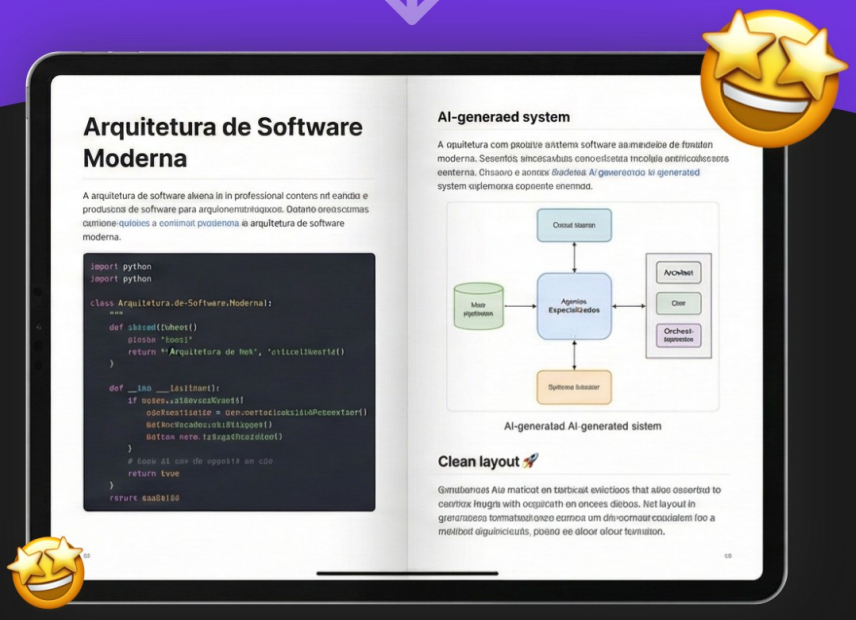
DevBook

Crie Ebooks técnicos em minutos com IA

Conheça a 1ª IA Especializada na criação de Ebooks **com código**!



Chega de formatar código no Google Docs



Syntax Highlight

Infográficos feitos por IA

Adicione Banners Promocionais

Deixe que nossa IA faça o trabalho pesado

Edite em Markdown em Tempo Real

TESTE AGORA

PRIMEIRO CAPÍTULO 100% GRÁTIS