PYTHON ACADEMY

MANIPULAÇÃO DE DATAS E HORAS NO PYTHON

Chega de sofrer para manipular Datas e Horas no Python. Bora aprender de uma vez por todas!

PYTHONACADEMY.COM.BR

Salve salve Pythonista 👋

Neste artigo, vamos explorar o poder da linguagem de programação Python para manipular datas e horários.

Como programadores, muitas vezes precisamos lidar com informações de tempo em nossas aplicações, seja para agendar tarefas, calcular intervalos ou exibir datas e horários para o usuário.

Portanto, é fundamental conhecer as ferramentas e bibliotecas disponíveis em Python para facilitar essa manipulação.

Trabalhando com a biblioteca datetime

A biblioteca `datetime` é uma das ferramentas mais úteis para trabalhar com datas e horários em Python.

Ela fornece uma série de classes e métodos para manipular e formatar datas e horários.

Vamos começar importando a biblioteca:

```
import datetime
```

Obtendo a data e hora atual

Para obter a data e hora atual, podemos utilizar a classe `datetime` e o método `now()`:

```
agora = datetime.datetime.now()
print(agora)
```

A saída será algo como:

```
2022-01-10 14:25:30.567894
```

Manipulando datas e horários

Podemos criar objetos `datetime` para representar uma data específica.

Para isso, utilizamos a função `datetime.datetime()` e informamos o ano, mês, dia, hora, minuto e segundo:

```
data = datetime.datetime(2022, 1, 1, 12, 0, 0)
print(data)
```

A saída será:

```
2022-01-01 12:00:00
```

Podemos ainda obter partes específicas de uma data ou horário. Por exemplo, para obter o ano de uma data:

```
ano = data.year
print(ano)
```

A saída será:

```
2022
```

Podemos também realizar operações matemáticas com datas, como somar ou subtrair dias, semanas, meses ou anos:

```
data = datetime.datetime(2022, 1, 1)
nova_data = data + datetime.timedelta(days=7)
print(nova_data)
```

A saída será:

```
2022-01-08 00:00:00
```

Formatando datas e horários

A biblioteca `datetime` também nos permite formatar datas e horários de acordo com nossas necessidades.

Podemos utilizar o método `strftime()` e especificar um formato desejado.

Por exemplo, para exibir uma data no formato dia/mês/ano:

```
data = datetime.datetime(2022, 1, 10)
data_formatada = data.strftime("%d/%m/%Y")
print(data_formatada)
```

A saída será:

```
10/01/2022
```

> 💡 *Estou construindo o **DevBook**, uma plataforma que usa IA para criar ebooks técnicos — com código formatado e exportação em PDF. Te convido a conhecer!*

Lidando com fusos horários

Quando trabalhamos com informações de tempo, muitas vezes precisamos lidar com diferentes fusos horários.

A biblioteca `datetime` não possui suporte nativo para isso, mas podemos utilizar a biblioteca `pytz` para lidar com fusos horários.

Para utilizar a biblioteca `pytz`, é necessário instalá-la primeiro. Você pode fazer isso através do gerenciador de pacotes `pip`:

```
pip install pytz
```

Após a instalação, podemos importar a biblioteca e utilizar a função `timezone()` para definir um fuso horário específico:

```
import pytz

fuso_horario = pytz.timezone('America/Sao_Paulo')
data = datetime.datetime.now(fuso_horario)
print(data)
```

A saída será a data e hora atual no fuso horário de São Paulo:

```
2022-01-10 14:25:30.567894-03:00
```

Trabalhando com intervalos de tempo

A biblioteca `datetime` também oferece suporte para trabalhar com intervalos de tempo.

Podemos utilizar a classe `timedelta` para representar uma duração de tempo específica.

Por exemplo, para representar um intervalo de 2 horas:

```
intervalo = datetime.timedelta(hours=2)
print(intervalo)
```

A saída será:

```
2:00:00
```

Podemos somar ou subtrair intervalos de tempo de objetos `datetime`. Por exemplo, para obter a data e hora daqui a 3 dias:

```
data_atual = datetime.datetime.now()
data_futura = data_atual + datetime.timedelta(days=3)
print(data_futura)
```

A saída será:

```
2022-01-13 14:25:30.567894
```

Conclusão

Neste artigo, exploramos diversas funcionalidades da biblioteca `datetime` em Python para manipular datas e horários.

Vimos como obter a data e hora atual, manipular datas e horários, formatar informações de tempo, lidar com fusos horários e trabalhar com intervalos de tempo.

Essas habilidades são essenciais para diferentes aplicações que envolvam informações de tempo.

Portanto, é fundamental dominar esses conceitos para realizar operações precisas, evitar erros e fornecer uma melhor experiência aos usuários.

Lembre-se de que a biblioteca `datetime` é apenas uma das ferramentas disponíveis em Python para trabalhar com datas e horários.

Existem outras bibliotecas, como a `dateutil`, que fornecem funcionalidades adicionais para tarefas mais avançadas.

Portanto, não hesite em explorar outras opções se necessário.

Espero que este artigo tenha sido útil para você entender como manipular datas e horários em Python.

Agora, você está preparado para utilizar esses recursos em suas próprias aplicações e tirar o máximo proveito da linguagem.

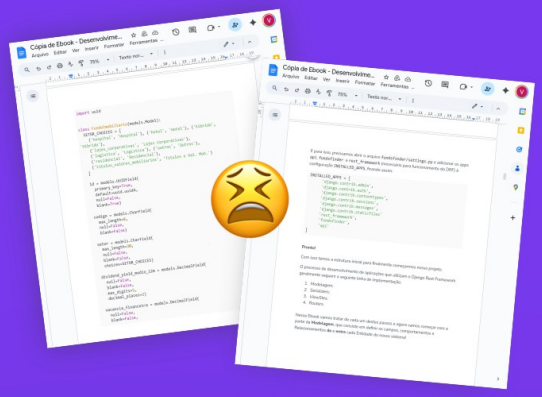
Nos vemos no próximo artigo, Pythonista! 👋

Não se esqueça de conferir!

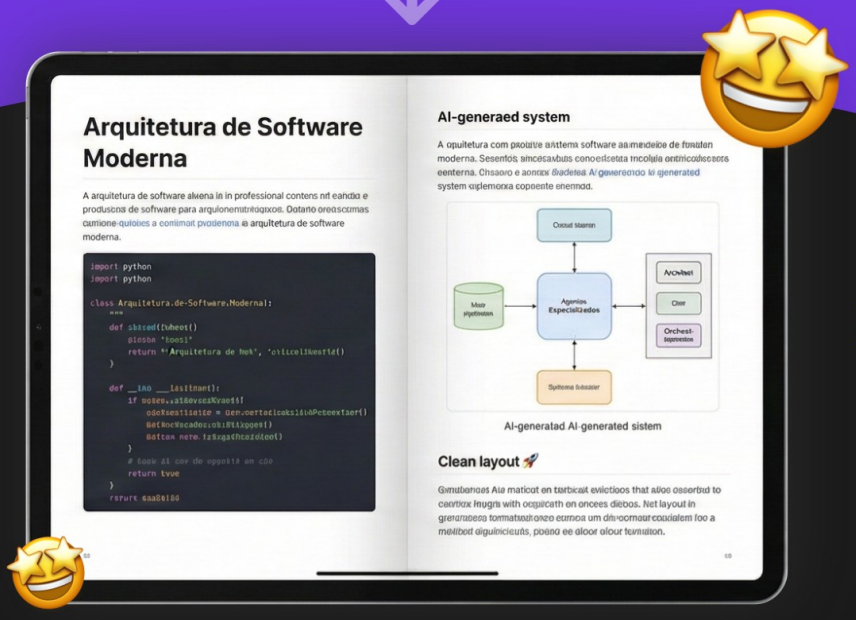
DevBook

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