

# ECM251 - Linguagens I

Pacotes com Python  
Herança e Polimorfismo  
Prof. Murilo Zanini de Carvalho

# Objetivos da aula de hoje

- Compreender o que são pacotes.
- Construir pacotes com Python.
- Compreender o conceito de herança com Python.
- Aplicar a estrutura de herança com Python.
- Classes Abstratas
- Interfaces
- Compreender o conceito de polimorfismo com Python.
- Aplicar a estrutura de polimorfismo com Python.

# Pacotes com Python

# Pacotes com Python

- Pacotes são uma forma de organizar o nosso projeto e agrupar funcionalidades dos módulos de Python.
- Os arquivos dentro de um pacote, são normalmente chamados de módulos.
- Todo pacote é um diretório com um arquivo `__init__.py` dentro dele (pode ser um arquivo vazio, mas ele precisa existir).

# Pacotes com Python

- Módulos podem ser executados como programas principais, em geral, é realizado adicionando uma verificação do atributo `__name__` do script.
- Mais informações sobre módulos:  
<https://docs.python.org/3/tutorial/modules.html>

# Pacotes com Python



```
def ola():  
    print("ola mundo!")  
  
if __name__ == "__main__":  
    ola()
```

# Observação:

## Python Magic Methods

# Python Magic Methods

- Existem alguns métodos especiais no Python que não precisam ser invocados explicitamente. Eles são chamados de “*Magic Methods*”.
- Mais informações sobre *Magic Methods*:  
<https://python-course.eu/oop/magic-methods.php>



Retirado de  
([https://python-course.eu/images/oop/marvin\\_the\\_magician\\_300w.webp](https://python-course.eu/images/oop/marvin_the_magician_300w.webp)), em  
16/08/2022

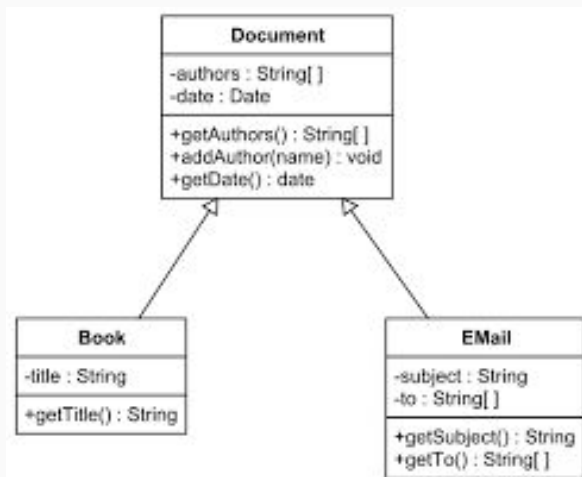


# Herança com Python

# Herança com Python

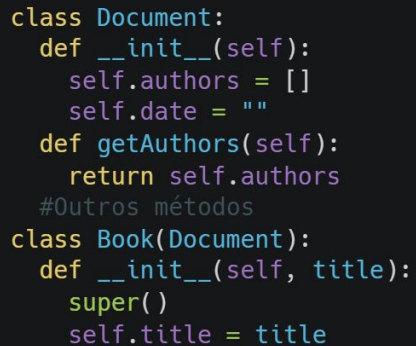
- É possível utilizar o conceito de herança em Python, possibilitando trabalhar com classes genéricas e classes específicas.
- Mais informações sobre herança:
  - <https://docs.python.org/3/tutorial/classes.html?highlight=class#inheritance>
  - <https://www.alura.com.br/apostila-python-orientacao-a-objetos/heranca-e-classes-abstratas>
  - <https://www.treinaweb.com.br/blog/utilizando-heranca-no-python>

# Herança com Python



Retirado de  
([https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSAZijejhQ8\\_VqmNqs-n9w06KGeKU72gQZMHg&usqp=CAU](https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSAZijejhQ8_VqmNqs-n9w06KGeKU72gQZMHg&usqp=CAU)), em  
16/08/2022

# Herança com Python



```
class Document:
    def __init__(self):
        self.authors = []
        self.date = ""
    def getAuthors(self):
        return self.authors
    #Outros métodos
class Book(Document):
    def __init__(self, title):
        super()
        self.title = title
```

# Classes Abstratas com Python

# Classes Abstratas com Python

- Com as classes abstratas, é possível criar generalizações que não podem ser instanciadas no projeto.
- Mais informações sobre classes abstratas:
  - <https://docs.python.org/3/library/abc.html?highlight=abstract>

# Classes Abstratas com Python

```
#Abstract Base Classes
from abc import ABC

class Document(ABC):
    def __init__(self):
        self.authors = []
        self.date = ""
    @abstractmethod
    def lala(self):
        pass
    def getAuthors(self):
        return self.authors
#Outros métodos
class Book(Document):
    def __init__(self, title):
        super()
        self.title = title
    def lala(self):
        print("Sobrecarga!")
```

# Classes Mix-In com Python



# Classes Mix-In com Python

- Em algumas situações, classes sem construtores são criadas apenas para adicionar uma funcionalidade em outras. Assim, é possível explorar a composição sem sobrecarregar a funcionalidade da classe.
- Mais informações sobre Mix-In:
  - <https://www.alura.com.br/apostila-python-orientacao-a-objetos/heranca-multipla-e-interfases>

# Interfaces com Python

# Interfaces com Python

- Não existe a palavra reservada Interface no Python. Essa implementação é realizada com classes abstratas e o método register, que vai criar um herança virtual. Essa herança virtual não traz as implementações do método pai, mas traz um contrato que deve ser cumprido entre a interface e quem está se registrando nela.
- Mais informações sobre herança:
  - <https://docs.python.org/3/tutorial/classes.html?highlight=class#inheritance>
  - <https://www.alura.com.br/apostila-python-orientacao-a-objetos/heranca-e-classes-abstratas>

# Polimorfismo com Python

# Polimorfismo com Python

- Determinação dinâmica de qual método deve ser invocado.
- <https://peps.python.org/pep-3119/>

# Perguntas?



Retirado de  
(<https://cdn-icons-png.flaticon.com/512/1268/1268705.png>), em 02/03/2022