

# Programação Orientada a Objetos

Introdução

Parte 2

# Programação Orientada a Objetos

## Classe:

No Python, novos objetos são criados a partir das classes por meio de atribuição. O objeto é uma instância da classe que possui características próprias. As classes são derivadas da classe base denominada *object*.

Veja como criamos um objeto:

```
Objeto = Classe()
```

No nosso exemplo seria:

```
Bulldog = Canino()
```

É como se o objeto fosse uma variável e o tipo fosse a classe.

Toda instância de classe ou variável tem seu próprio endereço de memória ou sua identidade.

Os objetos, que são instâncias de classes interagem entre si para servir ao propósito de uma aplicação em desenvolvimento.

# Programação Orientada a Objetos

## Classe:

Para criar uma classe em Python usamos a palavra reservada *class*.

```
class nome_classe:
```

```
    var = valor
```

```
    ...
```

```
    var = valor
```

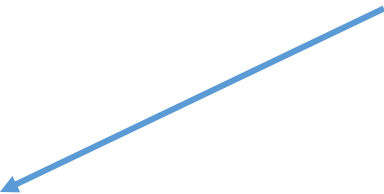
```
    def metodo(self, ...arg):
```

```
    ...
```

```
    def metodo(self, ...arg):
```

```
    ...
```

Representa o próprio objeto.



O primeiro argumento de um método é o *self*. Esta variável representa o próprio objeto. O nome *self* é uma convenção, podendo ser trocado por outro nome qualquer, porém é considerado como boa prática manter este nome.

# Programação Orientada a Objetos

heroi.py

```
class Heroi:
    """
    Classe de heróis
    """
    voa = False
    possui_arma = False
    lanca_teia = False
    frase_comum = ""

    def falar(self):
        print(self.frase_comum)

    def detalhar(self):
        if self.voa:
            print("O herói voa.")
        if self.possui_arma:
            print("O herói possui arma.")
        if self.lanca_teia:
            print("O herói lança teia.")
```

sistema.py

```
from heroi import Heroi

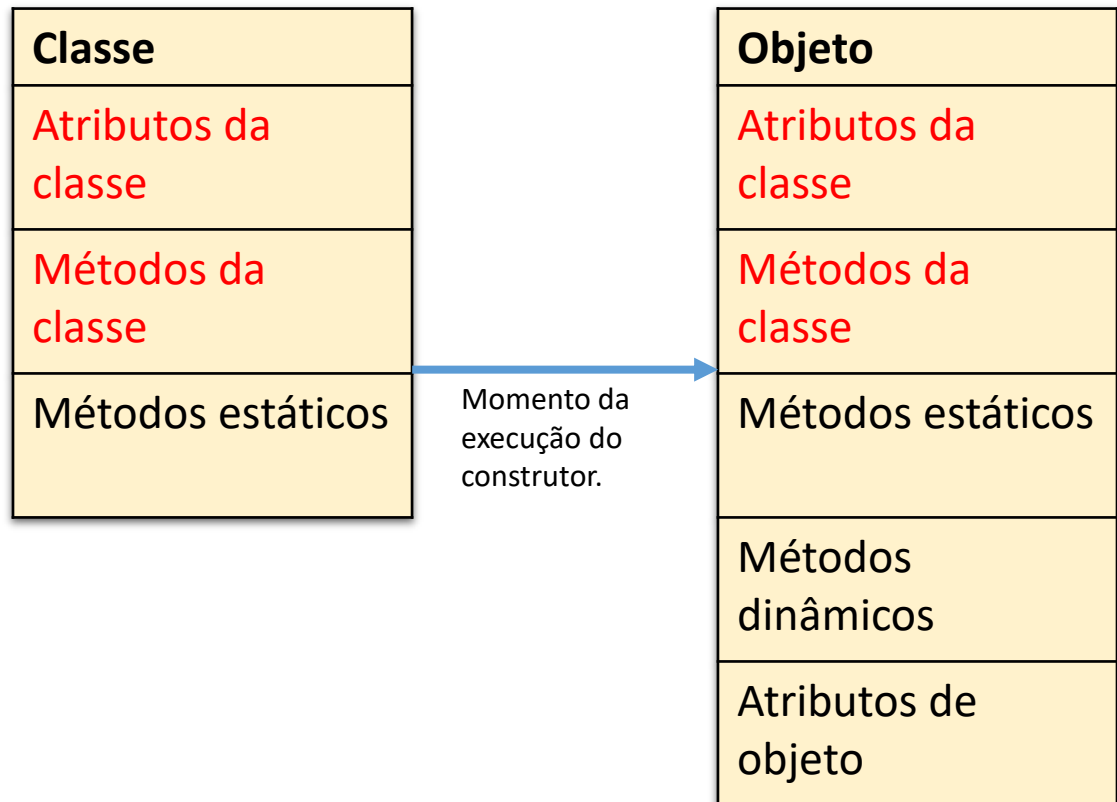
homem_aranha = Heroi()
homem_aranha.lanca_teia = True
print(homem_aranha.voa)
print(homem_aranha.lanca_teia)

he_man = Heroi()
he_man.possui_arma = True
he_man.lanca_teia = False
he_man.voa = False
he_man.frase_comum = "Eu tenho a força"
he_man.falar()

homem_aranha.detalhar()
he_man.detalhar()
```

# Programação Orientada a Objetos

## Classe:



Objeto = Classe()

No momento da criação do objeto, é executado o que chamamos de *construtor da classe*. O construtor é um método especial, chamado `__new__()`.

Após a chamada ao construtor, o método `__init__()` é chamado para inicializar a nova instância. O método `__init__()` pode ser usado para passar argumentos, assim podemos passar valores para os atributos do novo objeto.

Os métodos especiais em Python são identificados por nomes no padrão `__nome__()`. São utilizados dois underscores no início e no fim do nome.

# Programação Orientada a Objetos

heroi2.py

```
class Heroi:
    """
    Classe de heróis
    """
    def __init__(self, voa, possui_arma,
                  lanca_teia, frase_comum):
        print("Executando init...")
        self.voa = voa
        self.possui_arma = possui_arma
        self.lanca_teia = lanca_teia
        self.frase_comum = frase_comum

    def falar(self):
        print(self.frase_comum)

    def detalhar(self):
        if self.voa:
            print("O herói voa.")
        if self.possui_arma:
            print("O herói possui arma.")
        if self.lanca_teia:
            print("O herói lança teia.")
```

sistema2.py

```
from heroi2 import Heroi

# Heroi(voa, possui_arma, lanca_teia, frase_comum)
homem_aranha = Heroi(False, False, True, "")
print(homem_aranha.voa)
print(homem_aranha.lanca_teia)

he_man = Heroi(False, True, False, "Eu tenho a força!")
he_man.frase_comum = "Eu tenho a força"
he_man.falar()

homem_aranha.detalhar()
he_man.detalhar()
```

# CONTINUA...