

## Interpolação - Python 2 vs Python 3

A interpolação de strings também mudou entre o Python 2 e o Python 3.

Como você já viu, no Python 3 usa-se a função `format` junto com a sintaxe `{}` dentro da string, por exemplo:

```
"{} {}".format(1, 2)
```

O Python 2 usava uma sintaxe especial, ao invés do `format` era preciso usar o caractere `%`. Veja o exemplo:

```
"%d %d" % (1, 2)
```

Repare também que o `%` também era utilizado dentro da string.

Mais exemplos, sempre comparando o Python 2 com Python 3, existem no link: <https://pyformat.info/>  
(<https://pyformat.info/>).

Vale a pena ver!

### No Python 3.6+

A partir da [versão 3.6 do Python](https://docs.python.org/3/whatsnew/3.6.html#whatsnew36-pep498) (<https://docs.python.org/3/whatsnew/3.6.html#whatsnew36-pep498>), foi adicionado um novo recurso para realizar a interpolação de strings. Esse recurso é chamado de `f-strings` ou `formatted string literals`.

Esse recurso funciona da seguinte forma. Vamos imaginar que temos uma variável `nome`:

```
>>> nome = 'Matheus'
>>> print(f'Meu nome é {nome}')
Meu nome é Matheus
```

Quando colocamos a letra `f` antes das aspas, informamos ao Python que estamos utilizando uma `f-string`. Dessa forma o Python consegue, em tempo de execução, captar a expressão que está entre chaves (`{ }`) e avaliá-la.

Além de variáveis, podemos passar também de funções e métodos:

```
>>> nome = 'Matheus'
>>> print(f'Meu nome é {nome.lower()}')
Meu nome é matheus
```