



**ALGORITMIA E ESTRUTURAS DE DADOS**

**A BIBLIOTECA PILLOW**

**LICENCIATURA EM**

**TECNOLOGIAS E SISTEMAS DE INFORMAÇÃO PARA A WEB**

**#ESMAD #P.PORTO**



# Pillow

This is the home of Pillow, the friendly PIL fork.

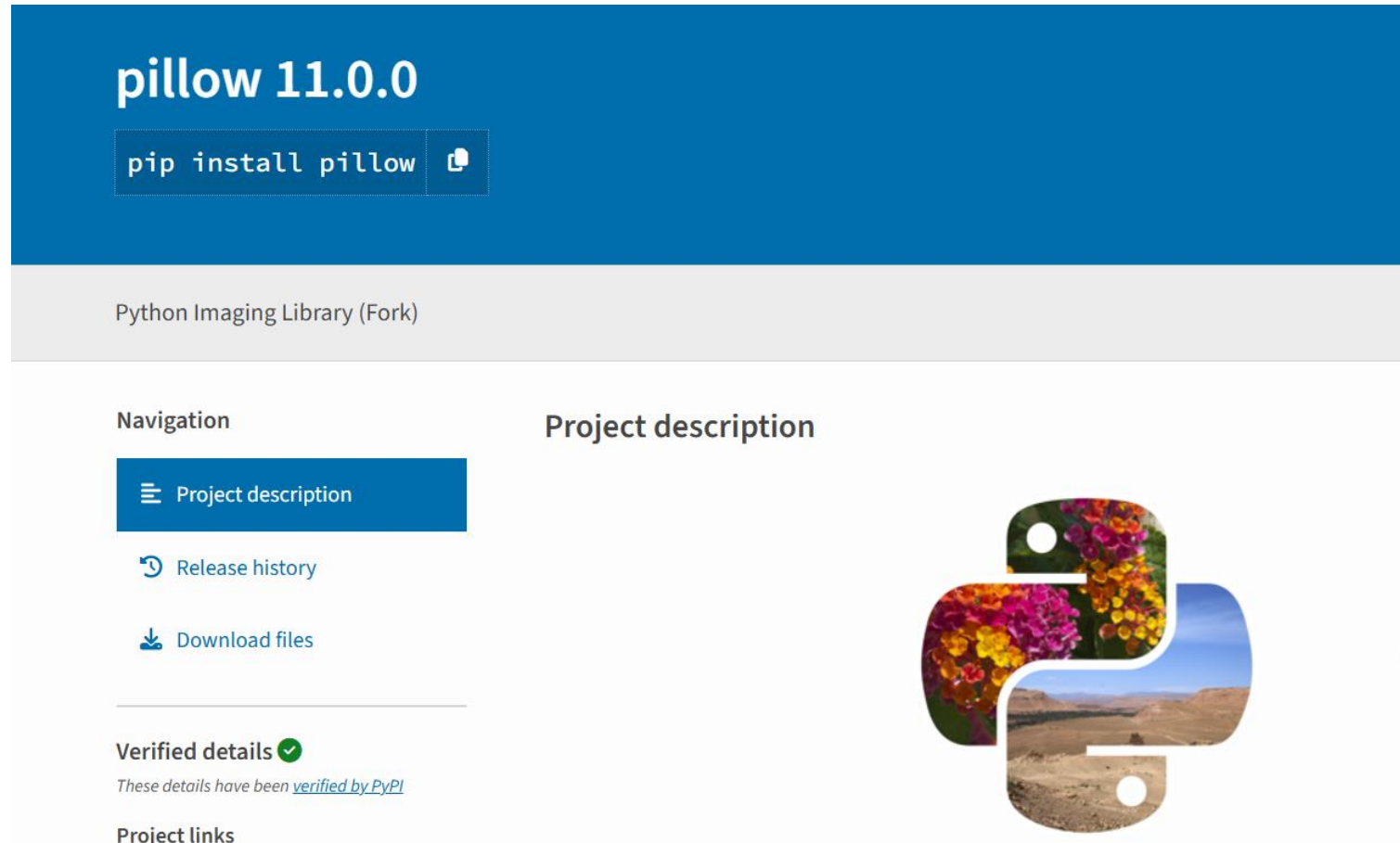
PIL is an acronym for Python Imaging Library. If you have ever worried or wondered about the future of PIL, please stop. We're here to save the day.

[Get from PyPI](#)

[Read the docs](#)


## ❖ Biblioteca pillow

- ❑ Biblioteca de processamento de imagens em python
  - ❑ Manipular janelas, imagens e pixéis
  - ❑ Suporta diferentes formatos como PNG, JPEG, TIFF, GIF, EPS






The screenshot displays the official PyPI page for the Pillow library, version 11.0.0. The top section features a blue header with the library name and version, a 'pip install pillow' button, and a copy icon. Below this, a light gray bar identifies it as the 'Python Imaging Library (Fork)'. The main content area is divided into two columns. The left column, titled 'Navigation', contains links for 'Project description' (highlighted with a blue bar), 'Release history', and 'Download files'. The right column, titled 'Project description', features a large, stylized Python logo where the two snakes are filled with vibrant, colorful flowers. Below the navigation links, a 'Verified details' section with a green checkmark icon states that the details have been verified by PyPI. The bottom of the page shows the start of a 'Project links' section.

**pillow 11.0.0**


`pip install pillow` 


Python Imaging Library (Fork)

**Navigation**

-  **Project description**
-  Release history
-  Download files

**Project description**



**Verified details** 

*These details have been [verified by PyPI](#)*

**Project links**

## ❖ Biblioteca pillow

### ❑ Instalar biblioteca pillow:

1. Aceder à linha de comando:



```
Linha de comandos × + v
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. Todos os direitos reservados.
C:\Users\mario>
```

2. Instalar biblioteca


```
>py -m pip install Pillow
```

3. Opcional: fazer upgrade do instalador pip

```
>py -m pip install Pillow --upgrade pip
```

## ❖ Biblioteca pillow

- ❑ Importar a biblioteca no VS Code:
- ❑ A biblioteca contém uma classe denominada Image, que inclui um vasto conjunto de métodos para processamento de imagens

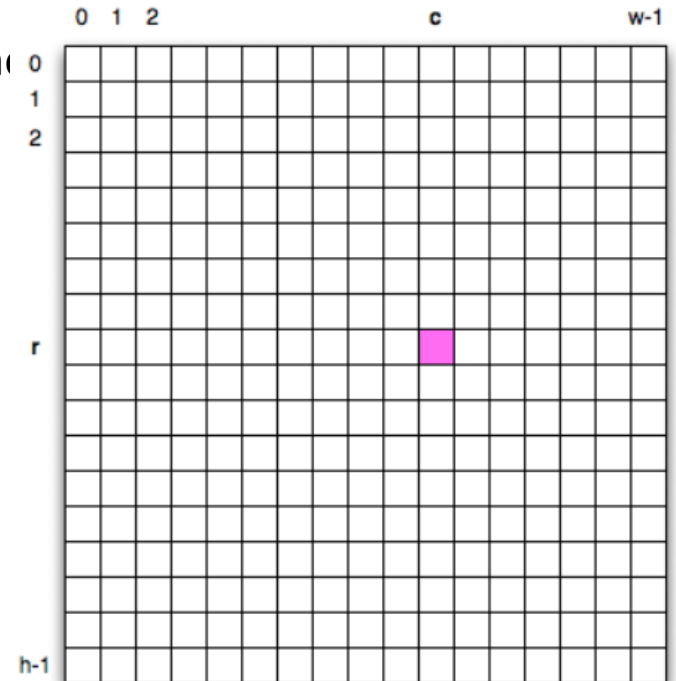


```
1  from PIL import Image
2
```

## ❖ Biblioteca pil - Python Imaging Library

### ❑ Pixels de uma imagem

- ❑ Uma imagem digital é uma coleção finita de **pixels** .
- ❑ Esses pixels são organizados numa tabela bidimensional. Cada pixel representa informação de imagem que está disponível. Esses pixels aparecem como pequenos quadrados.
- ❑ Cada imagem tem sua própria largura e sua própria altura.
- ❑ A largura é o número de colunas e a altura é o número de linhas. Podemos identificar os pixels usando o número da coluna e o número da linha
- ❑ Na figura ao lado, o pixel de interesse é encontrado na coluna  $c$  e na linha  $r$  .



## ❖ Biblioteca pil

### ❑ O modelo de cores **RGB – Red, Green, Blue**

❑ Cada pixel da imagem representa uma única cor.

❑ A cor específica depende de uma fórmula que mistura várias quantidades de três cores básicas: **(vermelho, verde e azul)**.




Essa técnica para criar cores é conhecida como RGB Color Model .

❑ A quantidade de cada cor, também chamada de intensidade da cor, é o que nos permite ter um controlo exato sobre a cor resultante.




❑ O valor mínimo de intensidade para uma cor básica é 0.  
A intensidade máxima de uma cor é 255.

❑ Por exemplo, se a intensidade do vermelho for 0, então não há vermelho no pixel. Se a intensidade ao azul for 255, então esse pixel contém a maior quantidade possível de azul

#### Example

Color	RGB	Color
	rgb(255,0,0)	Red
	rgb(0,255,0)	Green
	rgb(0,0,255)	Blue

#### Example

Color	RGB	Color
	rgb(0,0,0)	Black
	rgb(128,128,128)	Gray
	rgb(255,255,255)	White

## ❖ Biblioteca pil

❑ O modelo de cores RGB – Red, Green, Blue

Cor	Vermelho	Verde	Azul
Vermelho	255	0	0
Verde	0	255	0
Azul	0	0	255
Branco	255	255	255
Preto	0	0	0
Amarelo	255	255	0
Magenta	255	0	255

Colorname	RGB
AliceBlue	240,248,255
AntiqueWhite	250,235,215
Aqua	0,255,255
Aquamarine	127,255,212
Azure	240,255,255
Beige	245,245,220
Bisque	255,228,196
Black	0,0,0
BlanchedAlmond	255,235,205
Blue	0,0,255
BlueViolet	138,43,226
Brown	165,42,42
BurlyWood	222,184,135
CadetBlue	95,158,160
Chartreuse	127,255,0

<https://www.w3schools.com/colors/default.asp>

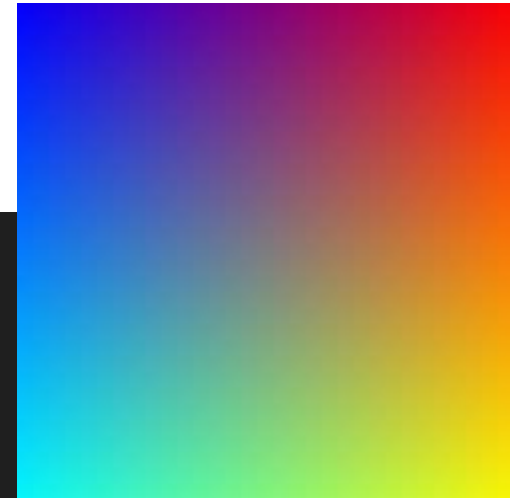
[https://www.w3schools.com/colors/color\\_tryit.asp?color=MidnightBlue](https://www.w3schools.com/colors/color_tryit.asp?color=MidnightBlue)



## ❖ Biblioteca pil

### ❑ Criar uma imagem

```
1  from PIL import Image
2  """
3  Cria uma imagem de 250x250 pixels
4  """
5  pathImages = ".\\images\\"          # Opcional: caminho da pasta atual de imagens
6
7  newSize = (250, 250) # Tuplo com largura (width) e altura (height) da imagem
8  imagem = Image.new(size=newSize, mode = "RGB", color= "white") # Nova imagem
9
10 pixelMap= imagem.load()             # "loads" the pixel para uma lista bidimensional
11 for i in range(imagem.width):
12     for j in range(imagem.height):
13         red = i
14         green = j
15         blue = 255-i
16         pixelMap[i,j] = (red, green, blue)
17 imagem.show()                       # mostra a imagem
18 imagem.save(pathImages+'teste.jpg') # Grava imagem no ficheiro pretendido
```



## ❖ Biblioteca pil

### ❑ Criar uma imagem

```
1 from PIL import Image
2 """
3 Cria uma imagem de 240x240 pixels
4 """
5 pathImages = ".\\images\\"          # Opcional: caminho da pasta atual de imagens
6
7 newSize = (240, 240) # Tuplo com largura (width) e altura (height) da imagem
8 imagem = Image.new(size=newSize, mode = "RGB", color= "white") # Nova imagem
9
10 pixelMap= imagem.load()             # "loads" the pixel para uma lista bidimensional
11 for i in range(imagem.width):
12     for j in range(imagem.height):
13         if j <80:
14             pixelMap[i,j] = (0, 0, 255)          # Cor?
15         elif j < 160:
16             pixelMap[i,j] = (255, 255, 255)      # Cor?
17         else:
18             pixelMap[i,j] = (255, 0, 0)          # Cor?
19 imagem.show()                          # mostra a imagem
20 imagem.save(pathImages+'teste1.jpg')          # Grava imagem no ficheiro pretendido
```

## ❖ Biblioteca pil

### ❑ Criar uma imagem

```
1 from PIL import Image
2 """
3 Cria uma imagem de 240x240 pixels
4 """
5 pathImages = ".\\images\\"          # Opcional: caminho da pasta atual de imagens
6
7 newSize = (240, 240) # Tuplo com largura (width) e altura (height) da imagem
8 imagem = Image.new(size=newSize, mode = "RGB", color= "white") # Nova imagem
9
10 pixelMap= imagem.load()           # "loads" the pixel para uma lista bidimensional
11 for i in range(imagem.width):
12     for j in range(imagem.height):
13         if j < 80:
14             pixelMap[i,j] = (0, 0, 255)          # Cor?
15         elif j < 160:
16             pixelMap[i,j] = (255, 255, 255)      # Cor?
17         else:
18             pixelMap[i,j] = (255, 0, 0)          # Cor?
19 imagem.show()                          # mostra a imagem
20 imagem.save(pathImages+'teste1.jpg')        # Grava imagem no ficheiro pretendido
```



## ❖ Biblioteca pillow

### ❑ Abrir uma imagem a partir de um ficheiro

```
1 from PIL import Image
2 """
3 abrir uma imagem a partir de um ficheiro
4 """
5 pathImages = ".\\images\\" # caminho da pasta atual de imagens
6 imagem1 = Image.open(pathImages+'img2.jpg') # path+ nome do ficheiro de imagem
7 imagem1.show() # Abrir imagem
8
```



## ❖ Biblioteca pillow

❑ Imagem: abrir | mostrar | guardar



```
1 from PIL import Image
2 """
3 abrir uma imagem a partir de um ficheiro
4 """
5 pathImages = ".\\images\\" # caminho da pasta atual de imagens
6 imagem1 = Image.open(pathImages+'img2.jpg') # path+ nome do ficheiro de imagem
7 imagem1.show() # Abrir imagem
8 imagem1.save(pathImages+'Papagaio.jpg') # Grava imagem noutra ficheiro
9
```

## ❖ Biblioteca pillow

### ❑ Atributos da imagem: size | mode | width | height

```
1 from PIL import Image
2 """
3 abrir uma imagem a partir de um ficheiro
4 """
5 pathImages = ".\\images\\" # caminho da pasta atual de imagens
6 imagem1 = Image.open(pathImages+'img2.jpg') # path+ nome do ficheiro de imagem
7 imagem1.show() # Abrir imagem
8 imagem1.save(pathImages+'Papagaio.jpg') # Grava imagem noutro ficheiro
9
10
11 print(imagem1.size, imagem1.mode, imagem1.format) # Alguns atributos da imagem
12 print(imagem1.width, imagem1.height)
```

```
(292, 173) RGB JPEG
292 173
```

## ❖ Biblioteca pillow

❑ Image: **resize**



```
1  # metodo resize()
2  pathImages = ".\\images\\"          # caminho da pasta atual de imagens
3  imagem2 = Image.open(pathImages+'img2.jpg') # path+ nome do ficheiro de imagem
4  newSize = (int(imagem2.width/2), int(imagem2.height/2))
5  resizedImage = imagem2.resize(newSize)
6  resizedImage.show()
7
8  newSize = (int(imagem2.width+1.5), int(imagem2.height*1.5))
9  resizedImage = imagem2.resize(newSize)
10 resizedImage.show()
```





## ❖ Biblioteca pillow

### ❑ Image: rotate | crop



```
1 pathImages = ".\\images\\" # caminho da pasta
2 imagem1 = Image.open(pathImages+'img1.jpg') # path+ nome do fi
3 # rotate
4 rotatedImage = imagem1.rotate(180, expand=True)
5 rotatedImage.show()
```



```
1 pathImages = ".\\images\\" # caminho da pasta
2 imagem1 = Image.open(pathImages+'img1.jpg') # path+ nome do fic
3 imageCropped = imagem1.crop((20,0,130,130)) # left, upper, right
4 imageCropped.show()
5 imageCropped.save(pathImages+'croppedImage.jpg')
```





## ❖ Biblioteca pillow

### ❑ Image: **editar pixels**



```
1  from PIL import Image
2  """
3  Criar imagem com uma moldura superior e inferior de 20 pixels, azul
4  """
5  pathImages = ".\\images\\"          # caminho da pasta atual de imagens
6  imagem1 = Image.open(pathImages+'img1.jpg')
7  pixelMap = imagem1.load()          # load de pixel data (lista bidimensional)
8
9  for i in range(imagem1.width):      # largura
10     for j in range(imagem1.height):  # altura
11         if j <20 or j > imagem1.height-20:
12             pixelMap[i,j] = (0,0,255) # Azul
13
14  imagem1.show()
15  imagem1.save(pathImages+'img1Moldura.jpg')
```

## ❖ Biblioteca pillow

### ❑ Image: editar pixels

```
1  from PIL import Image
2  """
3  Obter o Grayscale de uma imagem
4  """
5  pathImages = ".\\images\\" # caminho da pasta atual de imagens
6  imagem1 = Image.open(pathImages+'img2.jpg')
7  pixelMap = imagem1.load() # load de pixel data (lista bidimensional)
8  imagem1.show()
9
10 for i in range(imagem1.width): # largura
11     for j in range(imagem1.height): # altura
12         p = pixelMap[i,j] # pixel específico
13         red= p[0] # RGB de vermelho
14         green = p[1] # RGB de verde
15         blue = p[2] # RGB de azul
16         red = green = blue = int((red + green + blue) / 3)
17         pixelMap[i,j] = (red, green, blue) # Forma mais simples de grayScale
18
19  imagem1.show()
```

