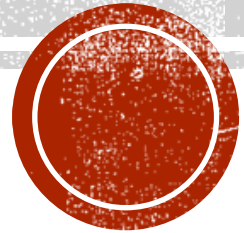
PILLOW



ЗАГРУЗКА ИЗОБРАЖЕНИЯ

from PIL import Image

```
img = Image.open("foto.gif")

# Открываем файл в бинарном режиме
f = open("foto.gif", "rb")
# Передаем объект файла
img = Image.open(f)
# Получаем размер изображения
img.size
(800, 600)
f.close() # Закрываем файл
```

МАТРИЦА ПИКСЕЛЕЙ

```
img = Image.open("foto.jpg")
obj = img.load()
obj[25, 45] # Получаем цвет пикселя
(122, 86, 62)
# Задаем цвет пикселя (красный)
obj[25, 45] = (255, 0, 0)
```

```
img = Image.open("foto.jpg")
# Получаем цвет пикселя
img.getpixel((25, 45))
(122, 86, 62)
# Изменяем цвет пикселя
img.putpixel((25, 45), (255, 0, 0))
# Получаем цвет пикселя
img.getpixel((25, 45))
(255, 0, 0)
img.show() # Просматриваем изображение
```

ИНВЕРСИЯ ИЗОБРАЖЕНИЯ

```
from PIL import Image,ImageDraw
```

```
img=Image.open("img1.jpg")
img.show()
img.save("img.png")
img=Image.open("img.png")
```

```
img.load()
draw = ImageDraw.Draw(img)
for x in range(img.width):
    for y in range(img.height):
        r=img.getpixel((x,y))[0]
        g=img.getpixel((x,y))[1]
        b=img.getpixel((x,y))[2]
        draw.point((x,y),(255-r,255-g,255-b))
img.show()
```

ПРЕОБРАЗОВАНИЕ ИЗОБРАЖЕНИЯ В ПОЛУТОНОВОЕ

```
img.load()
draw = ImageDraw.Draw(img)
for x in range(img.width):
    for y in range(img.height):
        r=img.getpixel((x,y))[0]
        g=img.getpixel((x,y))[1]
        b=img.getpixel((x,y))[2]
        sr=(r+g+b)//3
        draw.point((x,y),(sr,sr,sr))
img.show()
```

СОХРАНЕНИЕ ИЗОБРАЖЕНИЙ

```
# В формате JPEG
img.save("tmp.jpg")
# В формате BMP
img.save("tmp.bmp", "BMP")
f = open("tmp2.bmp", "wb")
# Передаем файловый объект
img.save(f, "BMP")
f.close()
```

ИЗМЕНЕНИЕ ФОРМАТА ИЗОБРАЖЕНИЯ

```
import os
import sys
from PIL import Image
jpg_images = [image for image in os.listdir() if image.endswith('.jpg')]
for jpg_image in jpg_images:
    try:
        new_name = jpg_image.split('.')[0] + '.png'
        Image.open(jpg_image).save(new_name)
    except IOError as error:
        print('Couldn\'t read {} '.format(jpg_image))
```

СОЗДАНИЕ ИЗОБРАЖЕНИЯ

```
img = Image.new("RGB", (100, 100))
img.show() # Черный квадрат
img = Image.new("RGB", (100, 100), (255, 0,
  0))
img.show() # Красный квадрат
img = Image.new("RGB", (100, 100), "green")
img.show() # Зеленый квадрат
img = Image.new("RGB", (100, 100), "#f00")
img.show() # Красный квадрат
```

ИЗМЕНЕНИЕ РАЗМЕРА ИЗОБРАЖЕНИЯ

```
Image.open("foto.jpg")
img.size # Исходные размеры изображения
(800, 600)
img.thumbnail((400, 300), Image.ANTIALIAS)
img.size # Изменяется само изображение
(400, 300)

img = Image.open("foto.jpg")
img.thumbnail((400, 100), Image.ANTIALIAS)
img.size # Размер изменяется пропорционально
(133, 100)
```

ПОВОРОТ ИЗОБРАЖЕНИЯ

```
img = Image.open("foto.jpg")
img.size # Исходные размеры изображения
(800, 600)
img2 = img.rotate(90)# Поворот на 90 градусов
img2.size
(600, 800)
img3 = img.rotate(45, Image.NEAREST)
img3.size # Размеры сохранены, изображение
  обрезано
(800, 600)
img4 = img.rotate(45, expand=True)
img4.size # Размеры увеличены, изображение полное
(991, 990)
```

ОБРЕЗКА

```
image = Image.open('jelly.jpg')

cropped = image.crop((0, 80, 200, 400))

cropped.save('/path/to/photos/cropped_jelly.png')
```

ЗАЛИВКА

```
# Закрасим область красным цветом
img = Image.open("foto.jpg")
img.paste( (255, 0, 0), (0, 0, 100, 100) )
img.show()

# Зальем все изображение зеленым цветом
img = Image.open("foto.jpg")
img.paste( (0, 128, 0), img.getbbox() )
img.show()
```

КОНВЕРТАЦИЯ ЦВЕТОВОЙ МОДЕЛИ

```
img = Image.open("foto.jpg")
img.mode
'RGB'
img2 = img.convert("RGBA")
img2.mode
'RGBA'
img2.show()
```

РИСОВАНИЕ

```
# Подключаем модули
from PIL import Image, ImageDraw
img = Image.new("RGB", (300, 300), (255, 255, 255))
# Создаем экземпляр класса
draw = ImageDraw.Draw(img)
```

```
from PIL import Image, ImageDraw
img = Image.new("RGB", (300, 300), (255,
  255, 255))
draw = ImageDraw.Draw(img)
for n in  range(5, 31):
    draw.point( (n, 5), fill=(255, 0, 0) )
img.show()
```

ЛИНИЯ

```
draw.line( (0, 0, 0, 300), fill=(0, 128, 0) )
draw.line( (297, 0, 297, 300), fill=(0, 128,
  0), width=3 )
img.show()
```

ПРЯМОУГОЛЬНИК

```
draw.rectangle( (10, 10, 30, 30), fill=(0,
  0, 255), outline=(0, 0, 0) )
draw.rectangle( (40, 10, 60, 30), fill=(0,
  0, 128))
draw.rectangle( (0, 0, 299, 299),
  outline=(0, 0, 0))
img.show()
```

МНОГОУГОЛЬНИК

```
draw.polygon((50, 50, 150, 150, 50, 150),
  outline=(0,0,0), fill=(255, 0, 0))
  # Треугольник
draw.polygon( (200, 200, 250, 200, 275, 250,
  250, 300, 200, 300, 175, 250), fill=(255,
  255, 0))
img.show()
```

ЭЛЛИПС

```
draw.ellipse((100, 100, 200, 200), fill=(255,
  255, 0))
draw.ellipse((50, 170, 150, 300), outline=(0,
  255, 255))
img.show()
```

ВЫВОД ТЕКСТА

```
from PIL import Image, ImageDraw,
  ImageFont
img = Image.new("RGB", (300, 300), (255,
  255, 255))
draw = ImageDraw.Draw(img)
font = ImageFont.load_default()
draw.text((10, 10), "Hello", font=font,
  fill="red")
img.show()
```

СОЗДАНИЕ СКРИНШОТА

```
from PIL import Image, ImageGrab
img = ImageGrab.grab()
img.save("screen.bmp", "BMP")
img.mode
'RGB'
img2 = ImageGrab.grab((100, 100, 300, 300))
img2.save("screen2.bmp", "BMP")
img2.size
(200, 200)
```

ФИЛЬТРЫ

```
from PIL import ImageFilter, Image

image = Image.open('jelly.jpg')

blurred_jelly = image.filter(ImageFilter.BLUR) \\размытое изображение

 image.filter(ImageFilter.SHARPEN) – меняет резкость
```

ВОДЯНОЙ ЗНАК

```
from PIL import Image
from PIL import ImageDraw
from PIL import ImageFont


def watermark_text(input_image_path,
                   output_image_path,
                   text, pos):
    photo = Image.open(input_image_path)

    # make the image editable
    drawing = ImageDraw.Draw(photo)
    black = (3, 8, 12)
    font = ImageFont.truetype("Pillow/Tests/fonts/FreeMono.ttf", 40)
    drawing.text(pos, text, fill=black, font=font)
    photo.show()
    photo.save(output_image_path)


if __name__ == '__main__':
    img = 'lighthouse.jpg'
    watermark_text(img, 'lighthouse_watermarked.jpg',
                   text='www.mousevspython.com',
                   pos=(0, 0))
```

```python
from PIL import Image

def watermark_with_transparency(input_image_path,
                                output_image_path,
                                watermark_image_path,
                                position):
    base_image = Image.open(input_image_path)
    watermark = Image.open(watermark_image_path)
    width, height = base_image.size

    transparent = Image.new('RGBA', (width, height), (0,0,0,0))
    transparent.paste(base_image, (0,0))
    transparent.paste(watermark, position, mask=watermark)
    transparent.show()
    transparent.save(output_image_path)


if __name__ == '__main__':
    img = 'lighthouse.jpg'
    watermark_with_transparency(img, 'lighthouse_watermarked3.jpg',
                                'watermark.png', position=(0,0))
```

СКЛЕЙКА ИЗОБРАЖЕНИЙ

```
def concat(images):
    height = images[0][0].size
    total_width = width * len(images[0])
    max_height = height * len(images)
    result = Image.new('RGBA', (total_width, max_height))
     y_offset = 0
     for line in images:
         x_offset = 0
         for element in line:
             result.paste(element, (x_offset, y_offset))
             x_offset += element.size[0]
         y_offset += line[0].size[1]
      return result
```

СРАВНЕНИЕ ИЗОБРАЖЕНИЙ

```
import numpy as np
from PIL import Image

with Image.open("house_left.jpg") as left:
    left.load()

with Image.open("house_right.jpg") as right:
    right.load()

left_array = np.asarray(left)
right_array = np.asarray(right)

difference_array =  right_array - left_array

difference = Image.fromarray(difference_array)
difference.show()
```