


ХОЧУ ПОМОЧЬ  
ПРОЕКТУ





practicum.yandex.ru

Углубленное изучение JavaScript и Node.js.

Для опытных разработчиков готовых выйти на новый уровень за 5 месяцев. Практика

Узнать больше

РЕКЛАМА · 16+  

Страница 3. / Библиотека Pillow в Python

## Обработка изображений в Python

Python довольно мощные возможности обработки изображений, идентифицирует и читает большое количество форматов. Pillow поддерживает большинство записи ограничена наиболее часто используемыми форматами. Pillow идеально подходит для обработки изображений. Ее можно использовать для создания эскизов, преобразования между форматами изображений. . Д.

### Библиотека Pillow содержит основные функции обработки изображений:

- содержит ряд готовых операций для изображениями L и RGB (автоконтраст, обрезка, масштабирование, вращение, произвольные аффинные преобразования и .т.д.)
- содержит набор предопределенных фильтров улучшения изображения (BLUR, DETAIL, SHARPEN и т.д.).
- при сохранении файлов JPEG может использовать настройки качества изображения, эквивалентные настройкам Photoshop.
- умеет делать копирование содержимого экрана (скриншот экрана).
- поддерживает фильтрацию с помощью набора встроенных ядер свертки и преобразования цветового пространства.
- метод гистограммы позволяет извлечь некоторую статистику из изображения, которую можно использовать для автоматического улучшения изображения и для статистического анализа.
- умеет загружать шрифты TrueType или OpenType с последующим созданием объекта шрифта с заданным размером для выполнения надписей на изображении.
- имеет простую 2D-графику для создания новых изображений, ретуширования существующих, а также для создания графики на лету для использования в Интернете.

### Содержание:

- [Установка Pillow в виртуальное окружение;](#)
- [Модель и режим изображений;](#)
- [Система координат;](#)
- [Прозрачность изображений;](#)
- [Пример использования модуля Pillow.](#)

### Установка Pillow в виртуальное окружение:

```
# создаем виртуальное окружение, если нет
$ python3 -m venv .venv --prompt VirtualEnv
# активируем виртуальное окружение
$ source .venv/bin/activate
# обновляем `pip`
(VirtualEnv):~$ python3 -m pip install -U pip
# ставим модуль `Pillow`
(VirtualEnv):~$ python3 -m pip install Pillow -U
```

### Модель и режим изображений.

Изображение может состоять из одной или нескольких полос/каналов с данными. Pillow позволяет хранить несколько каналов в одном изображении при условии, что все они имеют одинаковые размеры и глубину. Например, изображение PNG может иметь полосы R, G, B и A для значений прозрачности красного, зеленого, синего и альфа-канала. Многие операции воздействуют на каждую полосу отдельно, например, гистограммы. Полезно думать, что каждый пиксель имеет одно значение для каждой полосы. Чтобы получить количество и названия каналов на изображении, используйте метод [Image.getbands\(\)](#).

Режим изображения mode - это строка, определяющая тип и глубину пикселя изображения. Каждый пиксель использует полный диапазон битовой глубины. Таким образом, 1-битный пиксель имеет диапазон от 0 до 1, 8-битный пиксель имеет диапазон от 0 до 255, 32-разрядный целочисленный пиксель имеет диапазон INT32, а 32-битный пиксель с плавающей запятой имеет диапазон

FLOAT32.

Библиотека Pillow поддерживает следующие стандартные режимы:

- 1 (черно-белые 1-битные пиксели, хранятся по одному пикселю на байт)
- L (черно-белые 8-битные пиксели)
- P (8-битные пиксели, сопоставленные с любым другим режимом с помощью цветовой палитры)
- RGB (3x8-битные пиксели, истинный цвет)
- RGBA (4x8-битные пиксели, истинный цвет с маской прозрачности)
- CMYK (4x8-битные пиксели, цветоделение)
- YCbCr (3x8-битные пиксели, формат цветного видео)
  - Обратите внимание, это относится к стандарту JPEG, а не к стандарту ITU-R BT.2020.
- LAB (3x8-битные пиксели, цветовое пространство Lab)
- HSV (3x8-битные пиксели, оттенок, насыщенность, значение цветового пространства)
  - Диапазон оттенка 0-255 - это масштабированная версия 0 градусов <= Оттенок < 360 градусов
- I (32-разрядные целочисленные пиксели со знаком)
- F (32-битные пиксели с плавающей запятой)

Pillow пока не поддерживает многоканальные изображения с глубиной более 8 бит на канал.

Прочитать режим изображения можно через атрибут [Image.mode](#). Это строка, содержащая одно из указанных выше значений.

## Система координат.

Библиотека Pillow использует декартову пиксельную систему координат с (0,0) в верхнем левом углу. Обратите внимание, что координаты относятся к подразумеваемым углам пикселей. Центр пикселя, адресованного как (0, 0), на самом деле лежит в (0,5, 0,5).

Координаты обычно передаются в библиотеку в виде двух кортежей (x, y). Прямоугольники представляются как 4-х элементный [кортеж](#), причем левый верхний угол дается первым. Например, прямоугольник, покрывающий все изображение размером 800x600 пикселей, записывается как (0, 0, 800, 600).

Размер изображения можно прочитать через атрибут [Image.size](#). Это кортеж из двух элементов, содержащий размер по горизонтали [Image.width](#) и вертикали в пикселях [Image.height](#).

## Прозрачность изображений.

Если изображение не имеет альфа-канала, то прозрачность может быть указана в атрибуте [Image.info](#) с ключом transparency.

В большинстве случаев значение transparency представляет собой одно целое число, описывающее, какое значение пикселя является прозрачным в изображении в режимах 1, L, I или P. Однако изображения PNG могут иметь три значения, по одному для каждого канала в изображении в режиме RGB, или могут иметь строку байтов для изображения в режиме P (альфа-значение для каждой записи палитры).

## Ориентация изображений.

Общим элементом атрибута [Image.info](#) изображений в формате JPG и TIFF является тег ориентации EXIF. Это инструкция о том, как следует ориентировать данные изображения при просмотре в программе-просмотрщике изображений. Например, тег может указывать, что изображение повернуто на 90 градусов или зеркально отражено. Чтобы применить эту информацию к сохраняемому изображению, можно использовать [Image.exif.transpose\(\)](#).

## Пример использования модуля Pillow.

Прочитать изображение по URL можно следующим образом:

```
from PIL import Image
from urllib.request import urlopen
url = "https://python-pillow.org/images/pillow-logo.png"
# открываем URL
img = Image.open(urlopen(url))
# сохраняем изображение
img.save('pillow-logo.png')
```

Следующий код делает скриншот экрана и выполняет некоторые преобразования с полученным изображением. Все преобразования детально прокомментированы.

Вверх

```
from PIL import (Image, ImageEnhance, ImageOps,
                  ImageGrab, ImageDraw, ImageFont)
```

```
import time

# делаем задержку в 2 секунды, что бы
# успеть переключиться на нужное окно
time.sleep(2)
# делаем скриншот
im_orig = ImageGrab.grab()
# получаем размеры скриншота
print(f'Размеры: {im_orig.size}')
# получаем режим изображения скриншота
print(f'Режим: {im_orig.mode}')

# обрезаем по 100px с каждого края
im_orig = ImageOps.crop(im_orig, border=100)
# добавляем черную границу в 2px
im_orig = ImageOps.expand(im_orig, border=2, fill='#000000')
# и с верху добавляем белую границу в 10px
im_orig = ImageOps.expand(im_orig, border=10, fill='#ffffff')

# делаем надпись на скриншоте
idraw = ImageDraw.Draw(im_orig)
# текст
text = 'DOCS-PYTHON.RU'
# подключаем Font и задаем высоту в пикселях
font = ImageFont.truetype("/usr/share/fonts/truetype/freefont/FreeSans.ttf", size=18)
# вычисляем длину надписи
textlength = idraw.textlength(text, font)
# вычисляем положение надписи на скриншоте, например по ширине
# ширина скриншота - длина надписи - граница 2px + 10px
size = (im_orig.size[0]-textlength-12, im_orig.size[1]-32)
# накладываем текст на скриншот
idraw.text(size, text, font=font, fill='green')

# сохраняем обработанный скриншот
im_orig.save('scrshoot.png')
# сохраняем скриншот в формате `JPG`
# с оптимизацией и заданным качеством
im_orig.save('scrshoot.jpg', optimize=True, quality=90)

# Открываем изображение скриншота
# (для примера)
with Image.open('scrshoot.png') as im_orig:
    # получаем формат файла
    print(f'Формат файла: {im_orig.format}')

    # ВНИМАНИЕ! все дальнейшие преобразования
    # проделываем с оригинальным изображением `im_orig`
    # результат каждого преобразования будем сохранять

    # обесцвечиваем изображение с оттенками серого
    im = ImageOps.grayscale(im_orig)
    # сохраняем
    im.save('scrshoot_grayscale.png')
    # накладываем сепию 'black='#523224''
    im = ImageOps.colorize(im, black='#523224', white='#ffffff')
    im.save('scrshoot_colorize.png')

    # отображаем картинку зеркально
    im = ImageOps.mirror(im_orig)
    im.save('scrshoot_mirror.png')

    # `режим сканера` с порогом 150
    # т.е. применяем к каждому пикселю картинки функцию `fn`
    threshold = 150
    fn = lambda x : 255 if x > threshold else 0
    # метод image.convert('L') - предварительно обесцвечивает картинку
    im = im_orig.convert('L').point(fn, mode='1')
    im.save('scrshoot_scan.png')

    # увеличим резкость
    im = ImageEnhance.Sharpness(im_orig).enhance(2)
```

Вверх

```
im.save('scrshoot_Sharpness.png')

# увеличим контрастность
im = ImageEnhance.Contrast(im_orig).enhance(2)
im.save('scrshoot_Contrast.png')

# увеличим насыщенность
im = ImageEnhance.Color(im_orig).enhance(2)
im.save('scrshoot_Color.png')

# изменим размер изображения так, что бы оно вписалось в size
size = (700, 700)
# соотношения сторон сохраняются
im = ImageOps.contain(im_orig, size, method=Image.BICUBIC)
im.save('scrshoot_contain.png')

# создание `thumbnail`
size = (150, 150)
im_orig.thumbnail(size)
im_orig.save('scrshoot_thumbnail.jpg')
```

Содержание раздела:

- [КРАТКИЙ ОБЗОР МАТЕРИАЛА.](#)
- [Функция Image.open\(\) модуля Pillow](#)
- [Функция Image.new\(\) модуля Pillow](#)
- [Функция Image.frombytes\(\) модуля Pillow](#)
- [Функция Image.frombuffer\(\) модуля Pillow](#)
- [Функция Image.fromarray\(\) модуля Pillow](#)
- [Создание скриншота с использованием Pillow](#)
- [Объект Image модуля Pillow](#)
- [Обработка GIF изображений в Pillow](#)
- [Параметры для JPG, ICO и WebP модуля Pillow](#)
- [Извлечение EXIF-тегов модулем Pillow](#)
- [Встроенные фильтры улучшения модуля Pillow](#)
- [Фильтры передискретизации модуля Pillow](#)
- [Подмодуль ImageDraw модуля Pillow](#)
- [Функция Image.alpha\\_composite\(\) модуля Pillow](#)
- [Функция Image.blend\(\) модуля Pillow](#)
- [Функция Image.composite\(\) модуля Pillow](#)
- [Функция Image.merge\(\) модуля Pillow](#)
- [Функция Image.eval\(\) модуля Pillow](#)
- [Регулировка яркости, контрастности, резкости и насыщенности: Pillow](#)
- [Автоматическая регулировка контрастности, Pillow](#)
- [Тонирование черно-белого фото модулем Pillow](#)
- [Масштабирование изображений с модулем Pillow](#)
- [Добавить/обрезать рамку изображения, модуль Pillow](#)
- [Перевернуть/отразить изображение, модуль Pillow](#)
- [Обесцвечивание/инверсия изображения, модуль Pillow](#)
- [Операции с каналами изображений, модуль Pillow](#)