algoritmics

Module 4. Lesson 1.

Basics of image processing

Link to guidelines

Discussion:

Image processing

Developers, we have a new order!

The ProTeam specialists were approached by a representative of the Ministry of Social Development. He is preparing a software package for the elderly people.

It should include simple and useful applications for both experienced users and people with poor computer skills.

One of the applications should be Easy Editor.

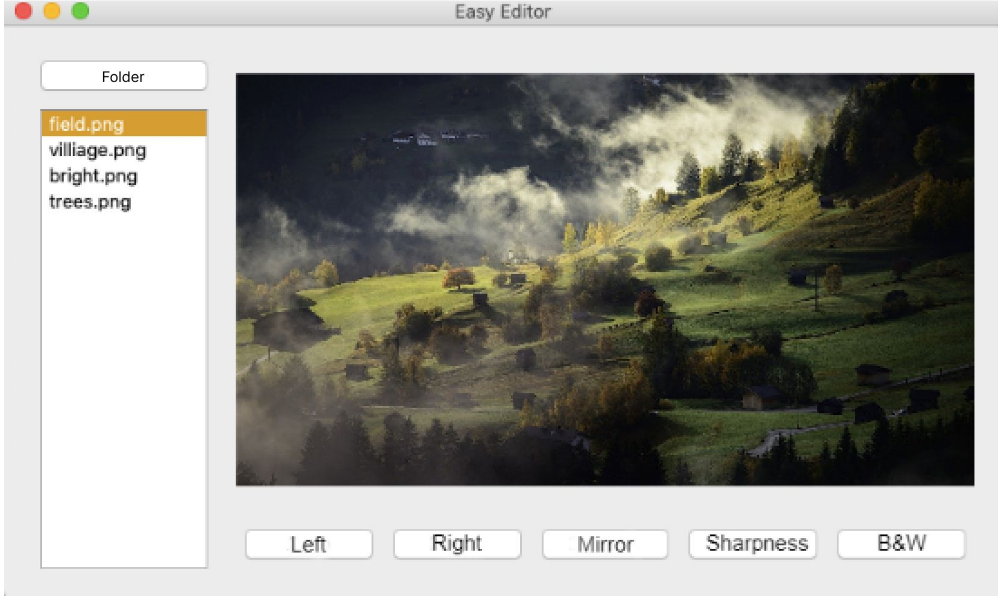
Let's study the technicals specifications in more detail!

Cole,
senior developer

Discussing work tasks

Let’s consider a possible solution

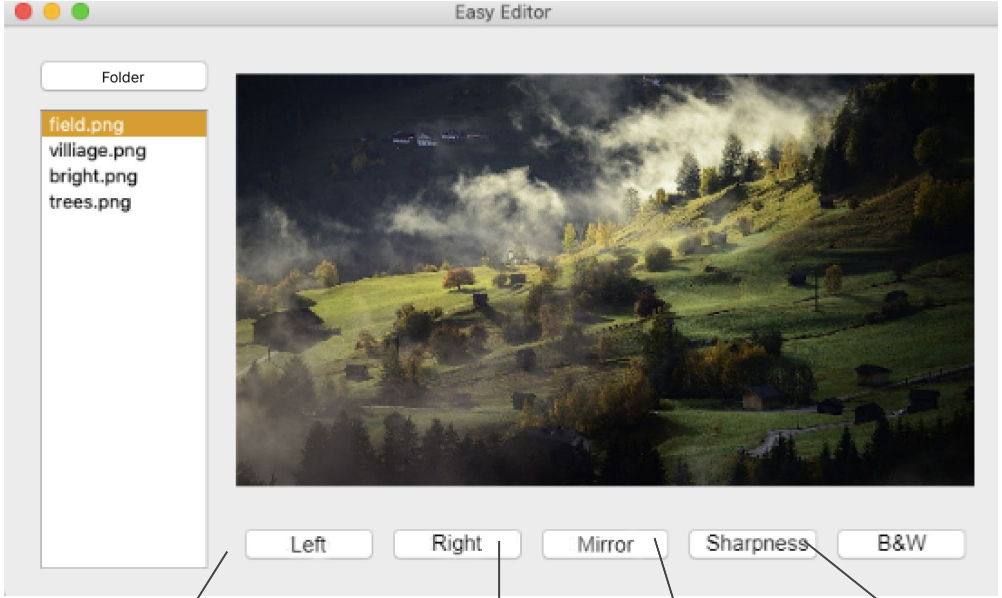
Examine the picture. What features should a photo editor have?



Discussing work tasks

Let’s consider a possible solution

The Easy Editor photo editor should be able to:



Process one or more images from a folder

Rotate the picture left or right 90 degrees

Display the mirror image

Sharpen the image

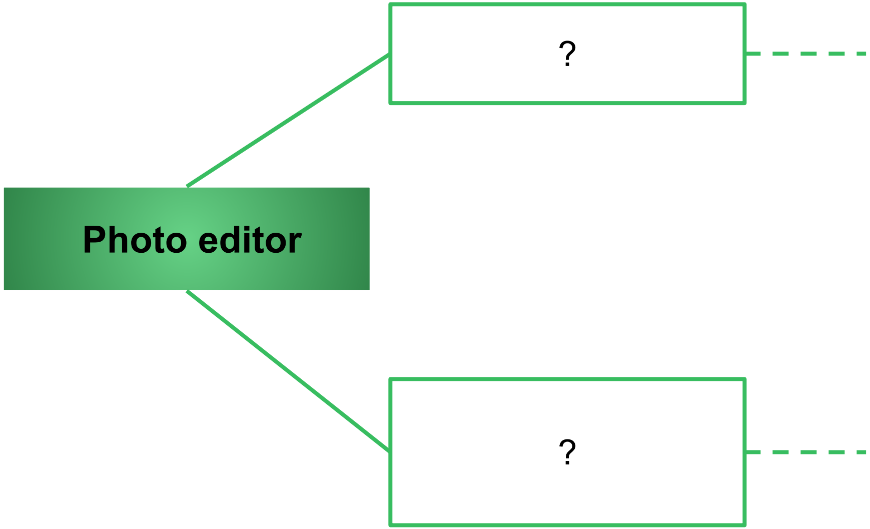
Make the picture black and white

Discussing work tasks

Planning our work on the project

You know two work planning tools: mind maps and checklists.

Let's start composing a **mind map**:

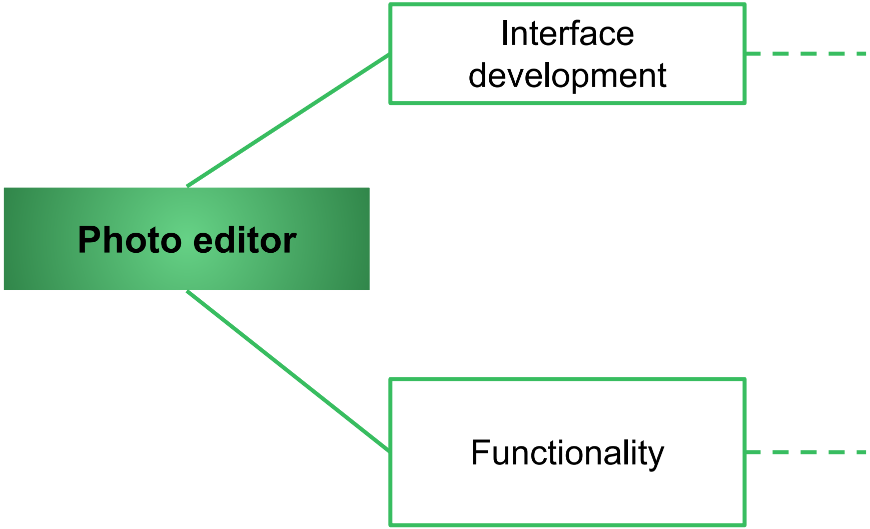


Discussing work tasks

Planning our work on the project

You know two work planning tools: mind maps and checklists.

Let's start composing a **mind map**:



*What **Python tools** will we need?*

Will we need to <u>study new libraries</u>?

Discussing work tasks

The goal of the work day is

to explore the PIL image processing library and prepare for the Photo Editor project.

Today you will :

- explore the capabilities of the PIL library for photo processing;
- recall the object-oriented approach to programming;
- program your own ImageEditor class for photo processing.

Module 4. Lesson 1. Basics of image processing

Qualifications

Demonstrate your knowledge

of working with files and object-oriented programming

Which command **opens** a text **file** for reading?

When will that file be closed?

Qualifications

Open a text file for reading

Command	*Purpose*
`with open("f.txt", "r") as file:`	Open a text file from the project folder for reading

The file will be closed automatically after executing a block of commands described inside with… as…

Qualifications

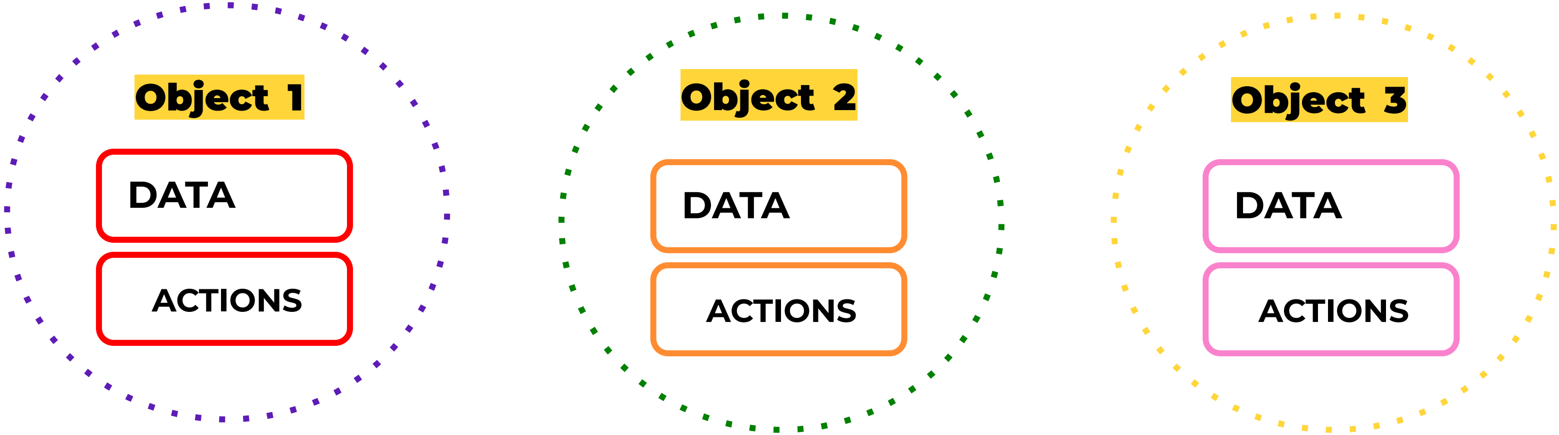
What is an **object**?

Name <u>at least three examples</u> of objects from the programming world.

Qualifications

An object

it is a collection of data and actions that is easy to perceive as a whole.



Object 1

DATA

ACTIONS

Object 2

DATA

ACTIONS

Object 3

DATA

ACTIONS

Turtle:

- *Appearance*,
- *Speed,* etc.

- *Move to a distance,* etc.

Application window:

- *Height*,
- *Title,* etc.

- *Show* the window.
- *Hide* the window.

Text file:

- *Extension*,
- *Volume,* etc.

- *Open,*
- *Add data,* etc.

What is a property?

What is a method?

Task.
You are given a piece of code. What are the names of the objects and their types? List the properties and methods provided in the program.

```
btn_OK = QPushButton('Answer')
btn_OK.setText('Next question')


window = QWidget()
window.setLayout(layout_card)
window.setWindowTitle('Memory Card')
window.show()
```

Qualifications

A property

is a variable inside an object.

A method

is a function inside an object.

Task.
You are given a piece of code. Name the objects and their **types**. List the **properties** and **methods** provided in the program.

```
btn_OK = QPushButton('Answer')
btn_OK.setText('Next question')


window = QWidget()
window.setLayout(layout_card)
window.setWindowTitle('Memory Card')
window.show()
```

Qualifications

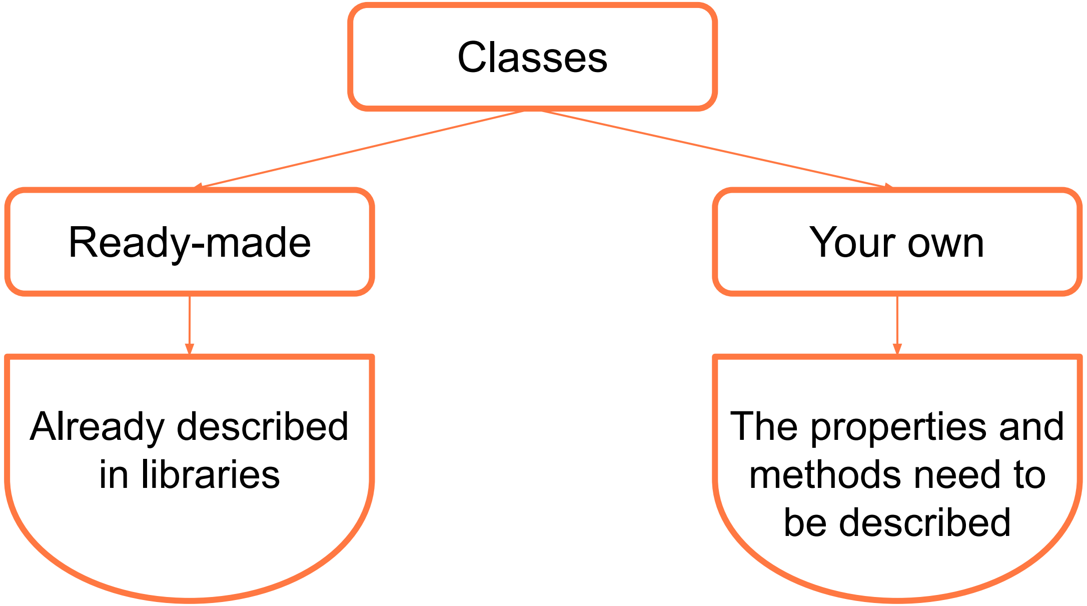
What is a **class**?

How do you create your own class ?

Qualifications

A class is

- a single name for many objects;
- in programming: a general description of how these objects should be arranged.



Classes

Ready-made

Your own

Already described in libraries

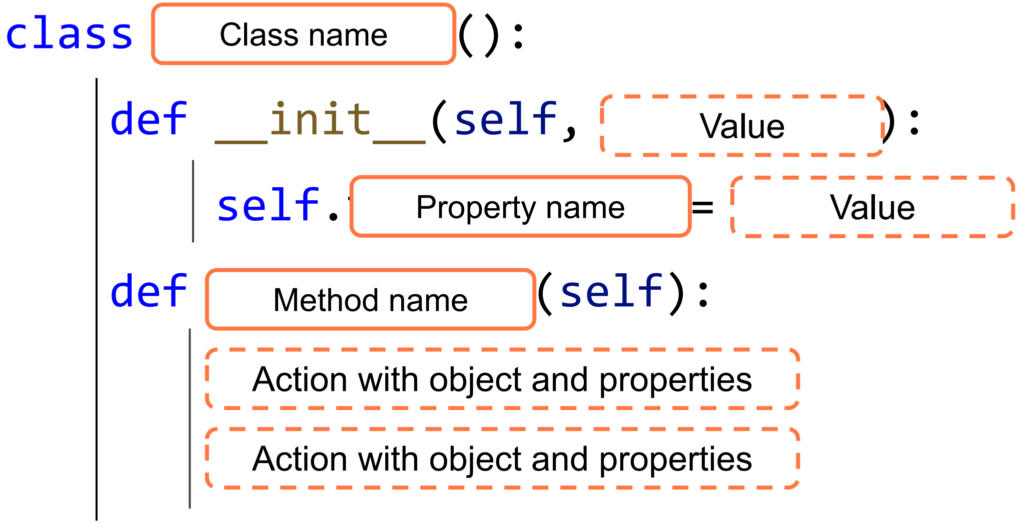
The properties and methods need to be described

Qualifications

Creating classes

To create a class, we need to do the following:

- List <u>in the constructor</u> the **properties** that define the characteristics of an instance of the class;
- list the **methods** for managing an instance.



```
class  Class name  ():
    def __init__(self,  Value  ):
        self. Property name  =  Value 
    def  Method name  (self):
         Action with object and properties 
         Action with object and properties 
```

A special **constructor** function that creates an instance of a class with the specified properties.

Two underscores.

Qualifications

Qualifications confirmed!

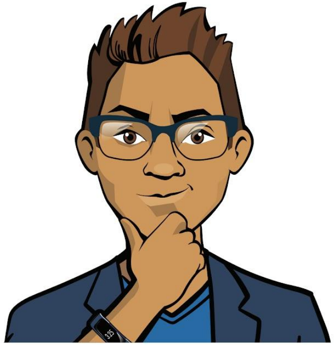
Great, you are ready to brainstorm and work on your tasks!

Brainstorming:

Image processing with PIL

Working with images

Let's recall what raster graphics are and start exploring the PIL library for working with images.
(Python Imaging Library).



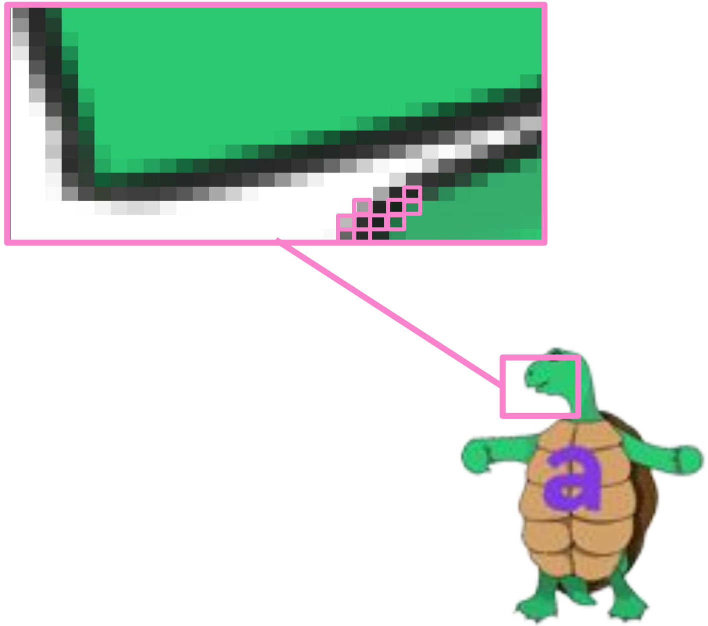
Brainstorming

A pixel is

a minute (indivisible) part of a graphic image

Raster is a set of pixels.

A raster image is a collection of dots (pixels) used to display a picture on a computer screen.



<u>You worked with raster graphics in the turtle module</u>

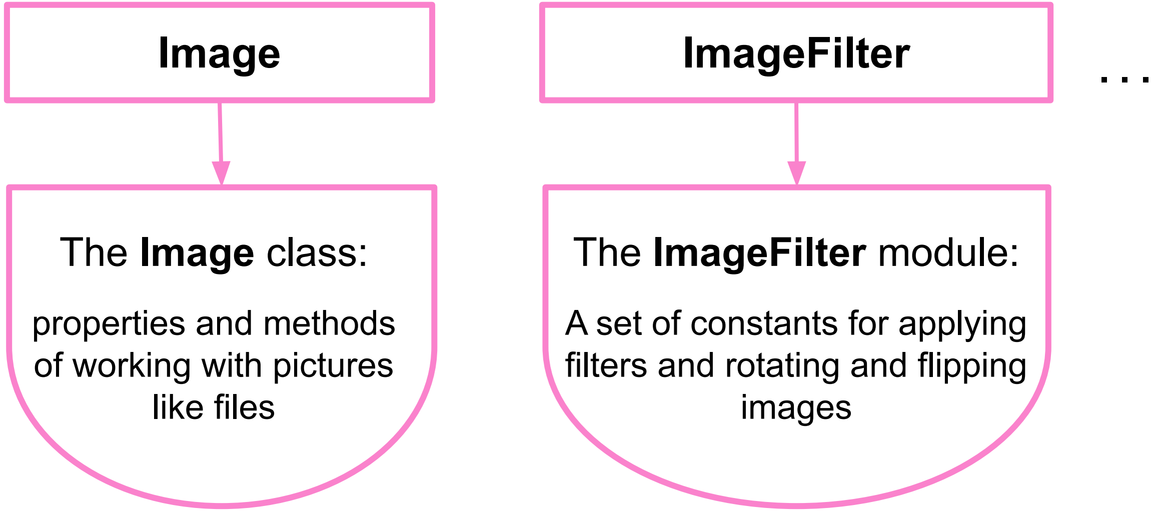
Brainstorming

The Python Imaging Library (PIL)

is a library for working with raster graphics

The PIL library has a hierarchical structure.

We'll need two modules from the framework base: **Image** and **ImageFilter.**



Image

The **Image** class:

properties and methods of working with pictures like files

ImageFilter

The **ImageFilter** module:

A set of constants for applying filters and rotating and flipping images

. . .

Brainstorming

Open an image to work with

To get a picture to work with, we need to import the Image module of the PIL library and open the file using the open() method and with… as operators.

Command	*Purpose*
`from PIL import Image`	Import the Image module from the PIL library

`with Image.open('photo.jpg') as original:`
`#or my_image = Image.open('photo.jpg')`
	Open a graphic file from the project folder

`    original.show()`	Open the image in a separate window

Brainstorming

Image options

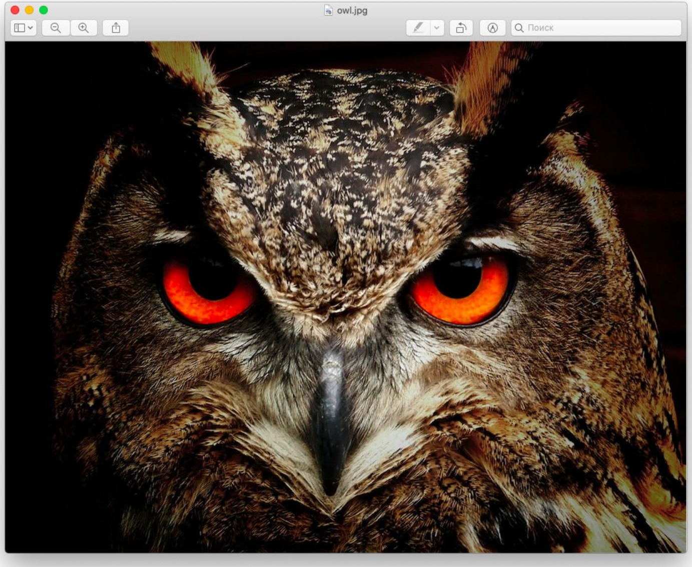
The resulting Image object has a number of properties.

Command	*Purpose*
`original.size`	File size (a pair in "length, width")
`original.format`	File format (jpg, png, bmp, etc.)
`original.mode`	File color type (color, black and white)

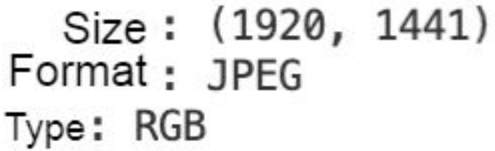
Brainstorming

Let’s look at the task

Task. In the project folder there is a photo called owl.jpg. Write a program that displays the properties of the image to the console and opens it in a separate window.



How do we solve the task?



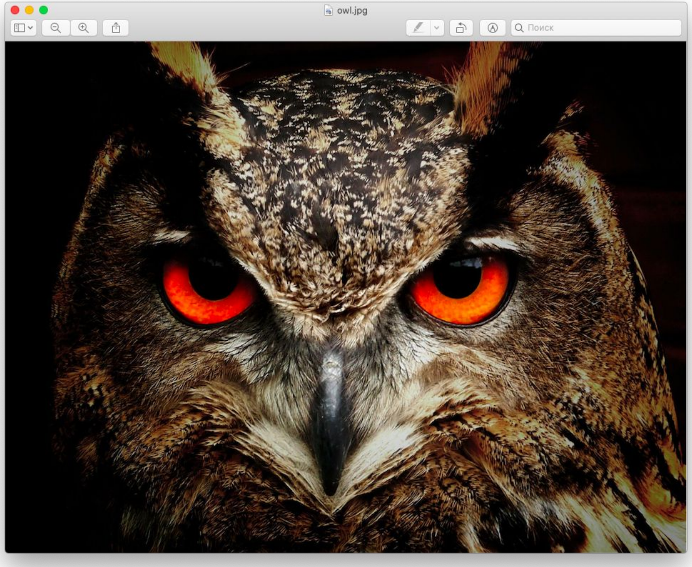
```
  Size : (1920, 1441)
Format : JPEG
Type: RGB
```

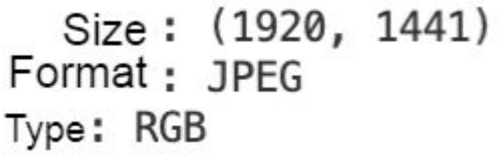
Brainstorming

Let’s look at the task

Task. In the project folder there is a photo called owl.jpg. Write a program that displays the properties of the image to the console and opens it in a separate window.

```
from PIL import Image
with Image.open('owl.jpg') as pic_original:
    print('Size:', pic_original.size)
    print('Format:', pic_original.format)
    print('Type:', pic_original.mode)
    pic_original.show()
```





```
  Size : (1920, 1441)
Format : JPEG
Type: RGB
```

Brainstorming

Image processing

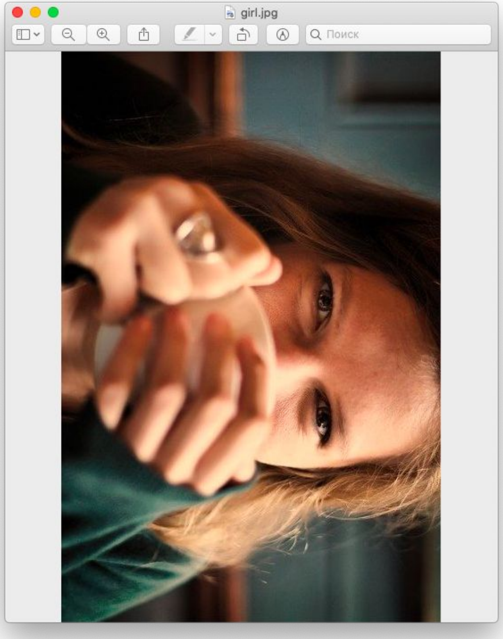
An object of the Image class can be modified using the methods and constants of the ImageFilter module.

Command	*Purpose*
`from PIL import ImageFilter`	Import the module with filters
`pic_gray = original.convert('L')`	Make the image black and white
`pic_blured = original.filter(ImageFilter.BLUR)`	Blur the image
`pic_up = original.transpose(Image.ROTATE_90)`	Rotate image left 90 degrees
`pic_gray.save('gray.jpg')`	Save the image in your project folder with the name gray.jpg

Brainstorming

Let’s look at the task

Task. In the project folder there is a photo called girl.jpg. Write a program that rotates the picture to the left 90 degrees and makes it black and white.



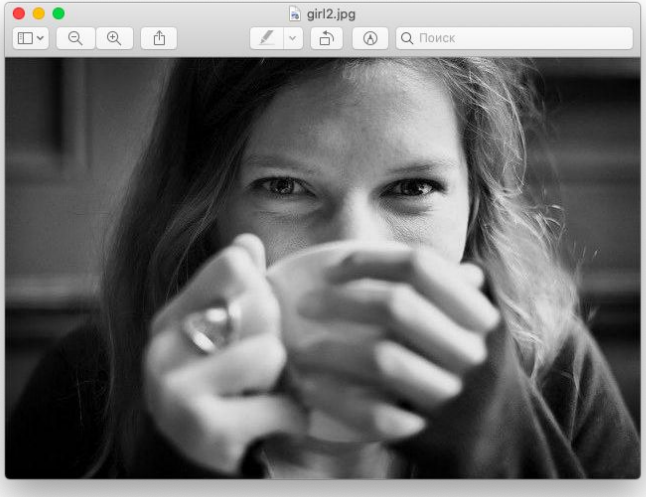
How do we solve the task?

Brainstorming

Let’s look at the task

Task. In the project folder there is a photo called girl.jpg. Write a program that rotates the picture to the left 90 degrees and makes it black and white.

```
from PIL import Image
from PIL import ImageFilter

with Image.open('girl.jpg') as pic_original:
    pic_original.show()

    pic_gray = pic_original.convert('L')
    pic_gray.save('girl1.jpg')
    pic_gray.show()

    pic_up = pic_gray.transpose(Image.ROTATE_90)
    pic_up.save('girl2.jpg')
    pic_up.show()
```



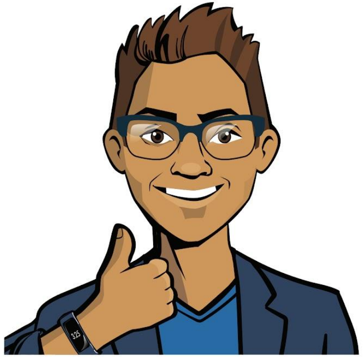
Brainstorming

Conclusions:

- Image processing is performed using the tools of the **PIL library**.

- The **Image module** contains commands for:
  - getting a picture,
  - opening it in a separate window,
  - saving it under a new name,
  - image processing by means of ImageFilter,
  - accessing the picture options.

- The **ImageFilter module** contains constants for image processing.



Brainstorming

Module 4. Lesson 1. Basics of image processing

Platform:

Basics of image processing

Complete the tasks in VS Code

"Graphics: classes"

<code>

Working on the platform

Module 4. Lesson 1. Basics of image processing

Brainstorming:

Image processing with PIL

Class for image processing

Linear processing wouldn't be very inconvenient when working with a lot of photos.

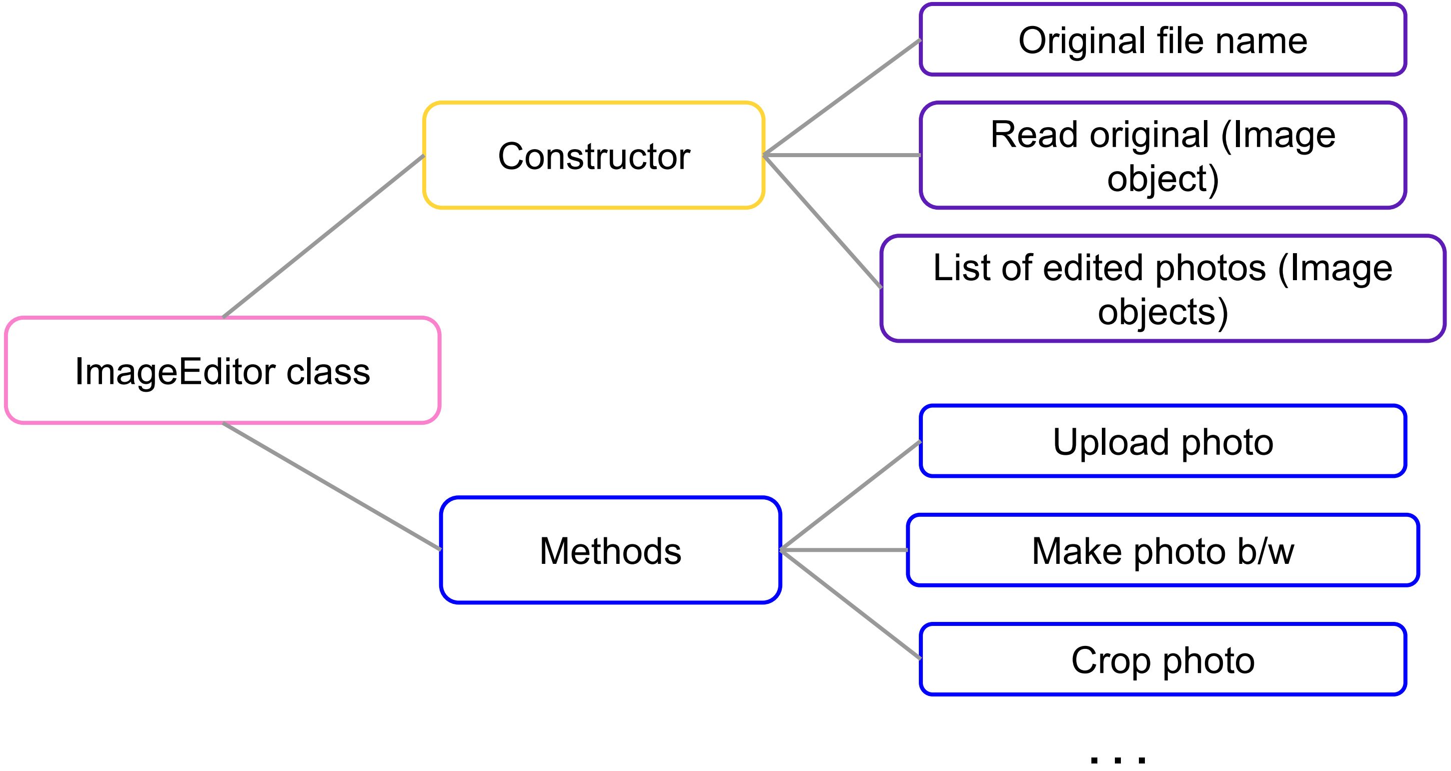
I suggest creating **our own class** with methods that process images.

Cole,
senior developer

Brainstorming

The ImageEditor class

Let's create an ImageEditor class with the following fields and methods:



ImageEditor class

Constructor

Original file name

Read original (Image object)

List of edited photos (Image objects)

Methods

Upload photo

Make photo b/w

Crop photo

...

Brainstorming

Loading an image to work with

When working with multiple files, there is another way to load images.
Compare:

Before:

```
with Image.open('original.jpg') as pic_original:
    pic_original.show()
```

Other way:

```
try:
    original = Image.open('original.jpg')
except:
    print('File not found!')
```

Simplified versions without the object-oriented approach.

Brainstorming

Loading an image to work with

When working with multiple files, there is another way to load images.
Compare:

Before:

```
with Image.open('original.jpg') as pic_original:
   pic_original.show()
```

Other way:

```
try:
    original = Image.open('original.jpg')
except:
    print('File not found!')
```

We will use this method when creating ImageEditor.

Simplified versions without the object-oriented approach.

Brainstorming

The ImageEditor class

Let's start creating a class with a constructor and a method for loading an image.
The class fields have already been defined by the senior developer.

```
class         ?         ():
    def __init__(self,       ?       ):
        self.filename = filename
        self.original = None
        self.changed = list()
    def open(self):
                   ?
                   ?
```

By default, there is nothing there. Later we will add a link to the uploaded original

Brainstorming

The ImageEditor class

Let's start creating a class with a constructor and a method for loading an image.
The class fields have already been defined by the senior developer.

```
class      ?      ():
    def __init__(self,    ?    ):
        self.filename = filename
        self.original = None
        self.changed = list()
    def open(self):
                 ?
                 ?
        ...
```

ImageEditor class fields:

- **file name** (photo.jpg);
- **link to original** photo;
- **list of modified** copies of the original.

What words should be in the blanks? Why?

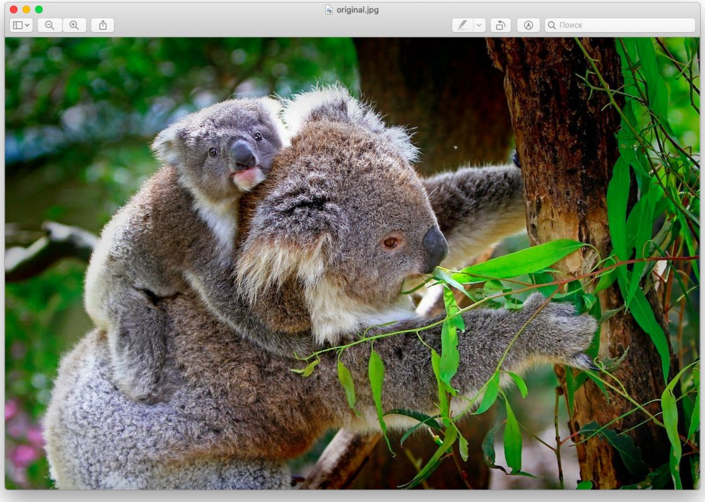
Brainstorming

```python
class ImageEditor():
    def __init__(self, filename):
        self.filename = filename
        self.original = None
        self.changed = list()
    def open(self):
        try:
            self.original = Image.open(self.filename)
        except:
            print('File not found!')
        self.original.show()
```

Brainstorming

Let’s look at the task

Task. Read from the project folder and open the file called original.jpg in a separate window. Use the ImageEditor class.

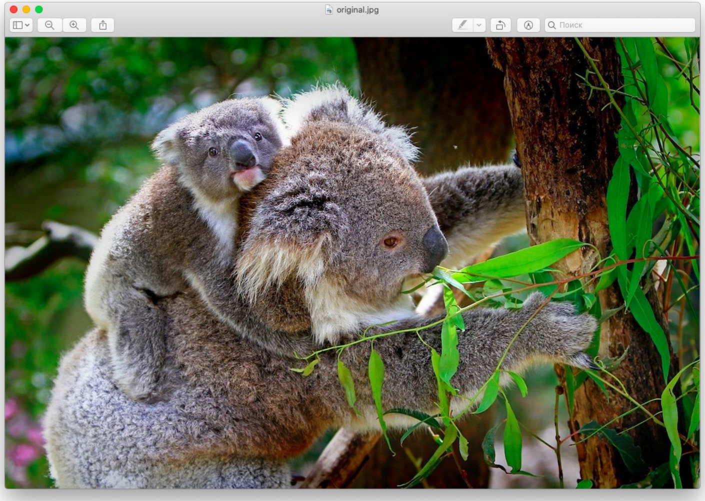


How do we solve the task?

Brainstorming

Possible solution

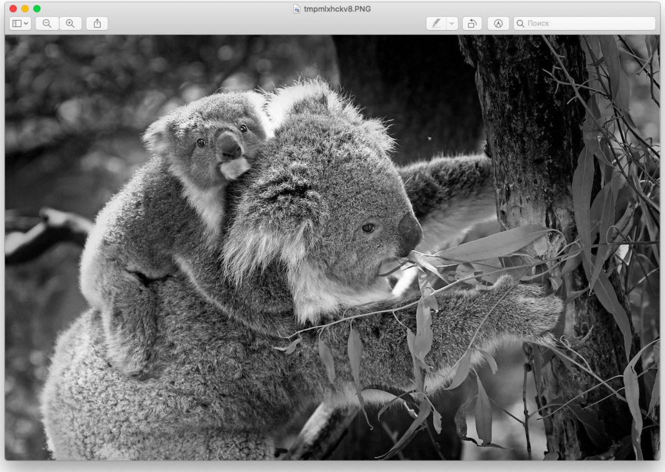
```
from PIL import Image
class ImageEditor():
   def __init__(self, filename):
       self.filename = filename
       self.original = None
       self.changed = list()

   def open(self):
       try:
           self.original = Image.open(self.filename)
       except:
           print('File not found!')
       self.original.show()


MyImage = ImageEditor('original.jpg')
MyImage.open()
```



Brainstorming

Let’s look at the task

Task. Process the original image: make it black and white. Program the processing as a method of the ImageEditor class.

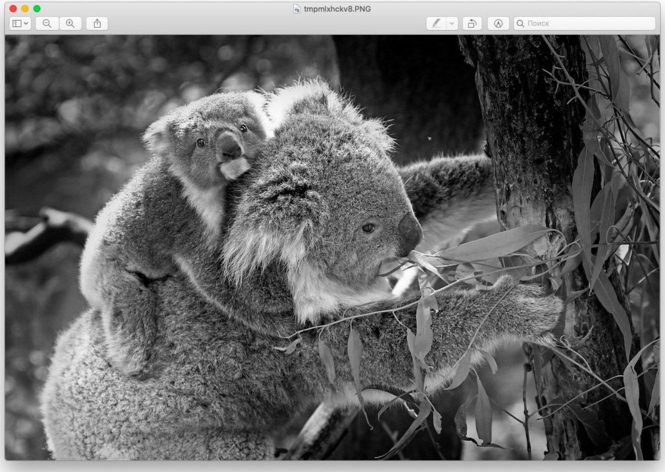


How do we solve the task?

Brainstorming

Possible solution

```python
from PIL import Image
from PIL import ImageFilter
class ImageEditor():
   def __init__(self, filename):
       #body of the class constructor
   def open(self):
       #body of the "load image" method
   def do_bw(self):
       gray = self.original.convert("L")
       self.changed.append(gray)
       gray.save('gray.jpg')

MyImage = ImageEditor('original.jpg')
MyImage.open()
MyImage.do_bw()
```



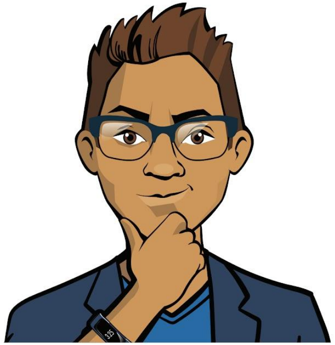
Brainstorming

Before we continue:

1. How could we process a picture differently, for example, blurring it? How will the ImageEditor class change?

2. Suppose we want to make two pictures black and white with the names cat.jpg and dog.jpg.

   How do we supplement the main part of the program?



Brainstorming

Module 4. Lesson 1. Basics of image processing

Platform:

Image processing using classes

Complete the tasks in VS Code

"Graphics: basics"

<code>

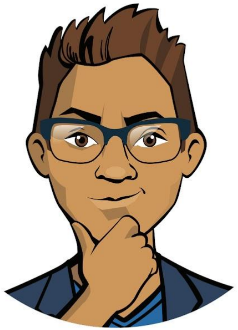
Working on the platform

Module 4. Lesson 1. Basics of image processing

Wrapping up the work day

Let’s wrap up the work day by answering these technical questions:

1. Which library contains image processing tools? What modules does it have?
2. What image processing methods do you know?
3. What is a class? What is the advantage of processing images with the ImageEditor class?



Cole,
senior developer

Emily,
project manager

Wrapping up the work day

Excellent work!

Colleagues,

Today you learned the basics of working with raster graphics using Python.

On our next work day, we will be able to start creating the "Photo Editor" application!

Task to improve efficiency

VS Code

Bonus work tasks

Summing up the work day