algoritmics

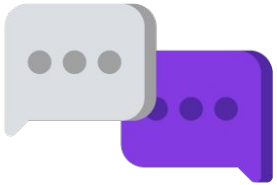
Module 4. Lesson 2.

The Easy Editor app. Part 1

Link to the methodological guidelines

Discussion:

Project planning

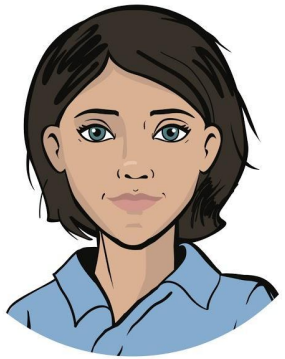


Starting a big order!

Last time, a representative of the Ministry for Social Development turned to ProTeam specialists.
He is making a software package for the elderly people.

One of the apps should be an **Easy Editor photo editor**.

To deliver the app on time, we need to plan our work on the order!



Emily,
Project Manager

Discussion of tasks

Technical specification

The **goal** is to program the Easy Editor app.

Expected app view:

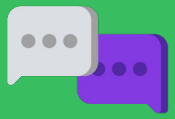
Discussion of tasks

Technical specification

The **goal** is to program the Easy Editor app.

Requirements:

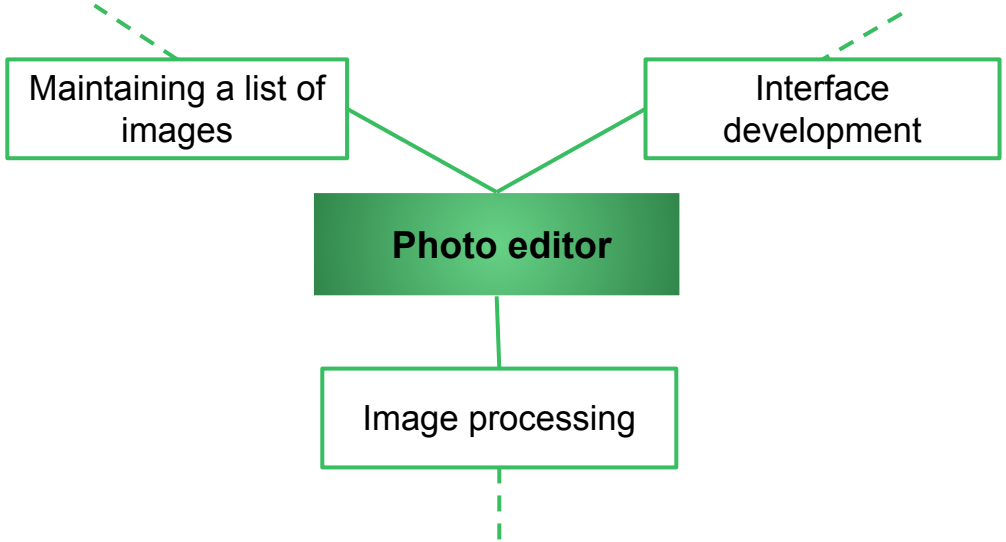
- ❏ The interface must be like in the picture.
- ❏ Ability to select a folder with images on a computer.
- ❏ Processing tools:
  - “Make it black and white”.
  - “Rotate left (90°)”.
  - “Rotate right (90°)”.
  - “Sharpen”.
  - “Mirror (left to right)”.
- ❏ Saving edited photos (copies of the original) to a new Modified subfolder.



Discussion of tasks

Planning work on the project

Let’s build the project **mind map** :

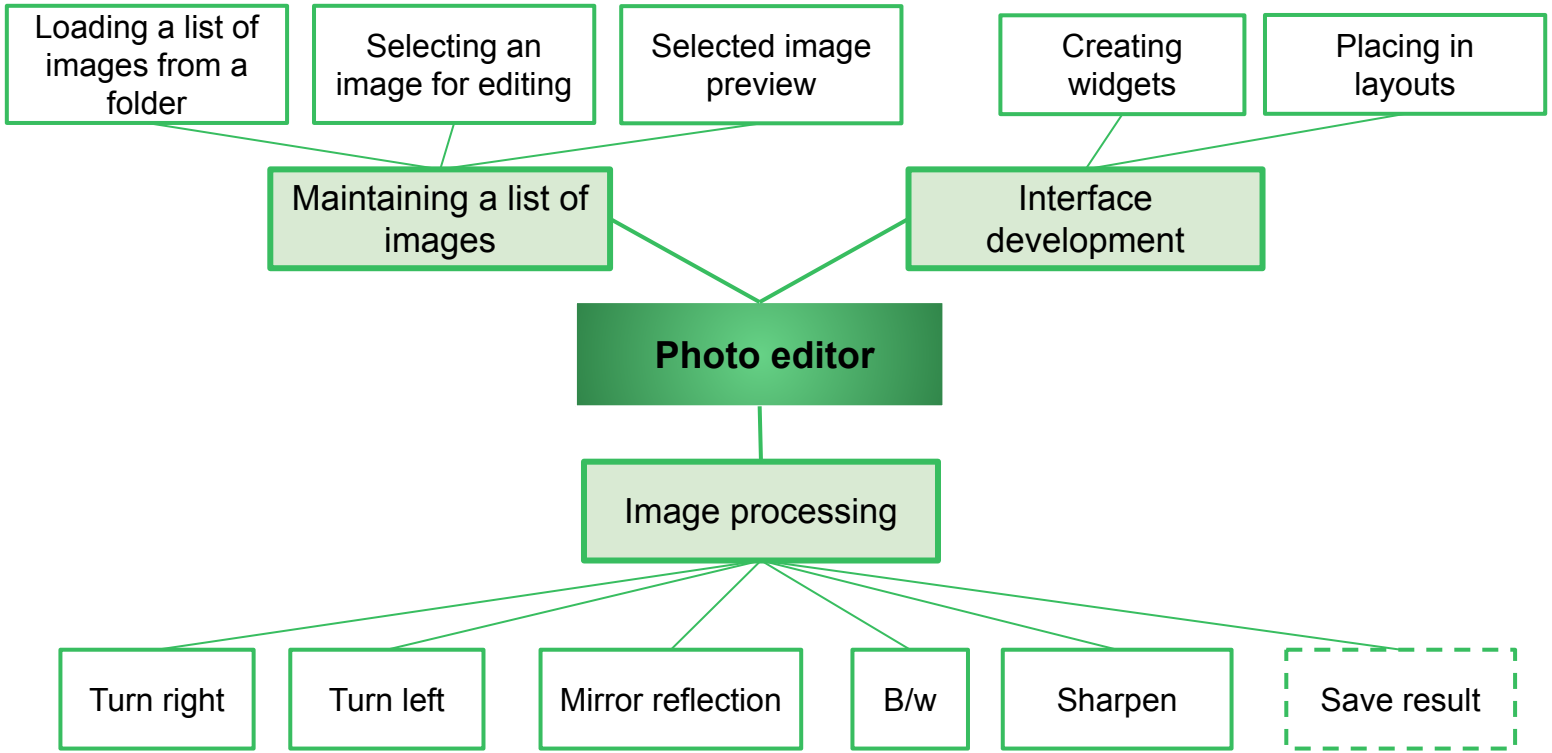


Developer Cole has already identified three main areas.
What blocks can be added to the flowchart?

Discussion of tasks

Planning work on the project

Project **mind map:**



Discussion of tasks

Planning work on the project

Checklist based on the **mind map**:

1. *Create an interface* for the app.
2. *Ensure loading* images from the required folder.
3. *Show a preview* of the image selected in the list.
4. *Program editing* of a photo:
    - creating a modified copy;
    - showing a preview of the modified copy;
    - saving to the Modified subfolder.

Discussion of tasks

Planning work on the project

Checklist based on the **mind map**:

Today

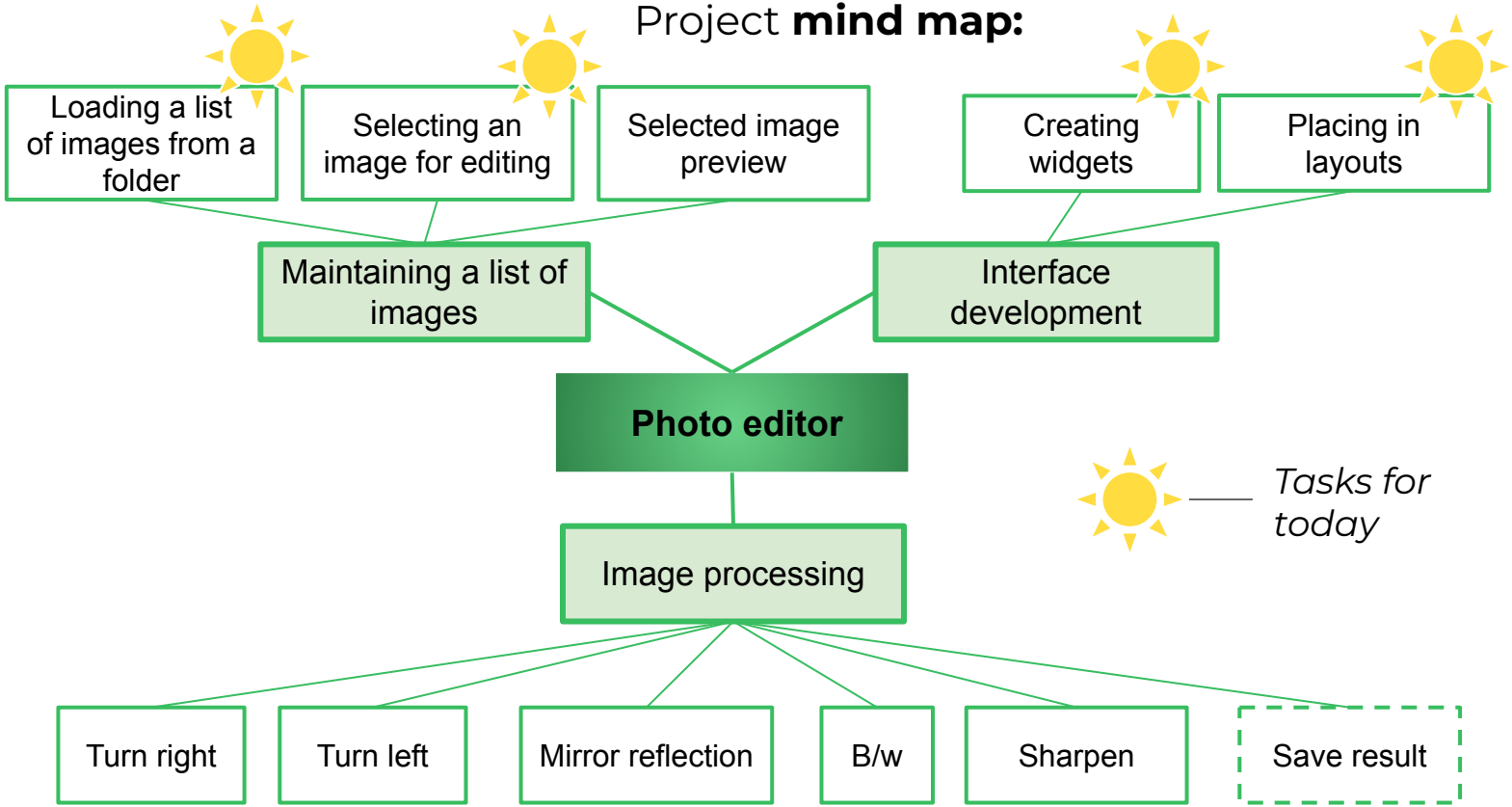
1. *Create an interface for* the app.
2. *Ensure loading* images from the required folder.
3. *Show a preview* of the image selected in the list.
4. *Program editing* of a photo:

    - creating a modified copy;
    - showing a preview of the modified copy;
    - saving to the Modified subfolder.

Discussion of tasks

Planning work on the project

Project **mind map:**



Tasks for today

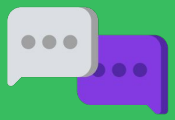
Discussion of tasks

The goal of the working day is

to create an interface for the Easy Editor app and configure uploading images from any folder on a computer.

Today you will:

- Recall how to build interfaces using PyQt.
- Start exploring the capabilities of the os module for working with an operating system.
- Program your own DirList class to load and display a list of images.



Discussion of tasks

Qualification

Demonstrate your knowledge

of scopes and UI elements

Qualification

What formats of graphic files do you know?

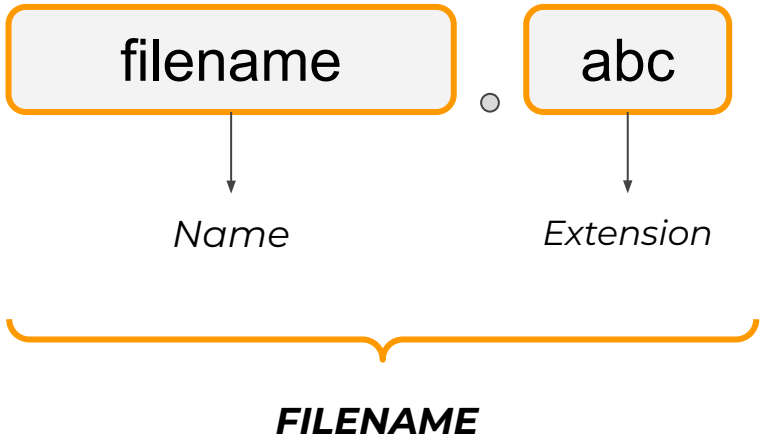
How does the file format differ from the extension ?

Qualification

Graphic file formats:

- JPG.
- PNG.
- BMP.
- SVG.
- EPS.

etc.



Qualification

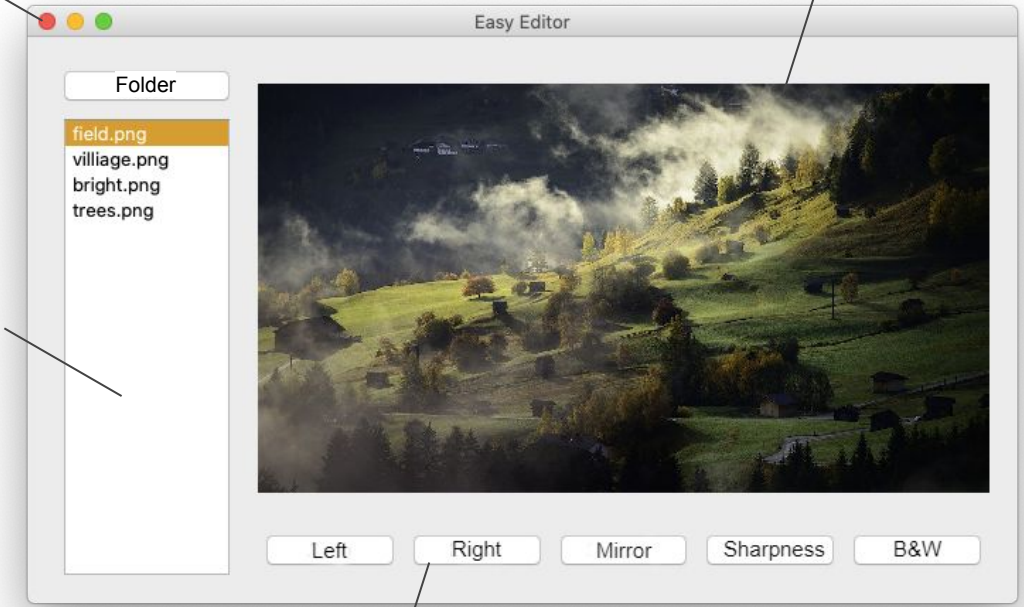
Show and name all the widgets in the picture:

Qualification

Widget names:

App window — **QWidget**

Static label/image — **QLabel**



Selectable list — **QListWidget**

Button — **QPushButton**

Qualification

How to create an empty application window?

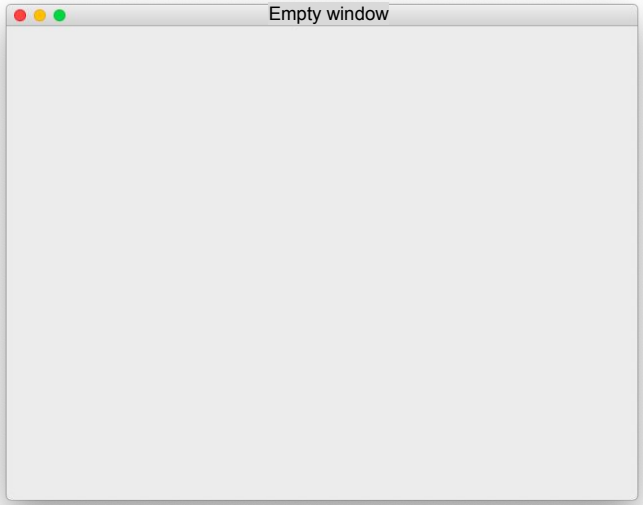
List the PyQt5 modules and commands required.

If you are confused, use the theoretical documentation!

Qualification

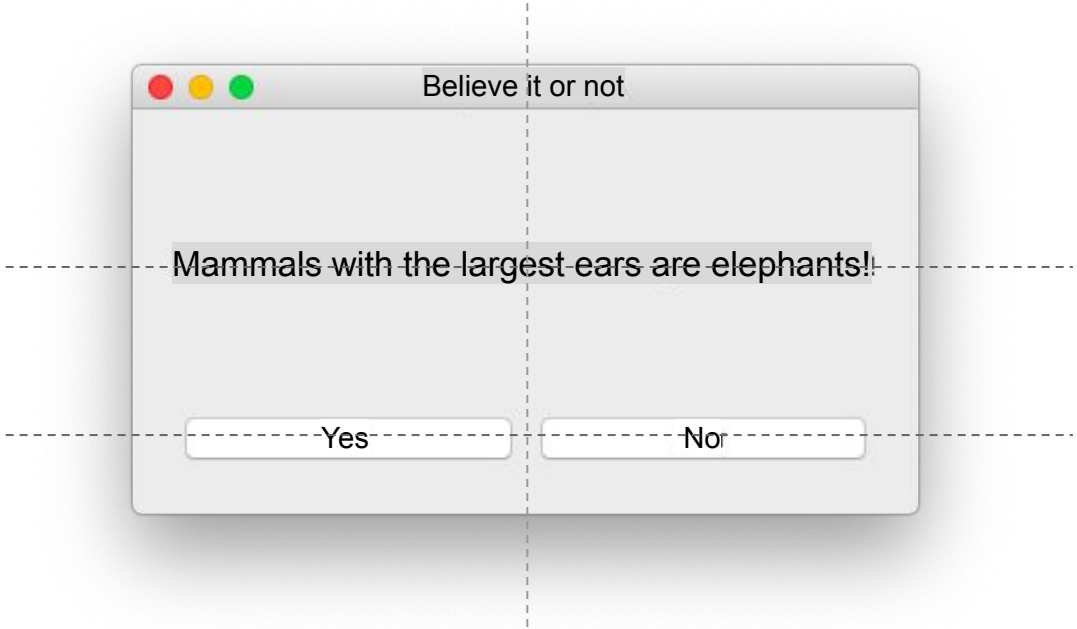
Creating an empty application window:

```
from PyQt5.QtWidgets import QApplication, QWidget

app = QApplication([])
main_win = QWidget()
main_win.setWindowTitle('Empty window')

main_win.show()
app.exec_()
```



Qualification

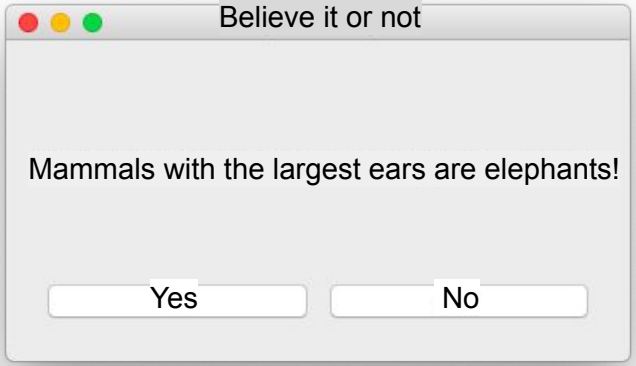
How do I create and position widgets in an empty window like in the picture?



If you are confused, use the theoretical documentation!

Qualification

```
from PyQt5.QtWidgets import QApplication, QWidget, QPushButton, QLabel, QVBoxLayout, QHBoxLayout
from PyQt5.QtCore import Qt

app = QApplication([])
main_win = QWidget()
main_win.resize(300, 200)
main_win.setWindowTitle('Believe it or not')
statement = QLabel('Mammals with the largest ears are elephants!')
btn_yes = QPushButton('Yes')
btn_no = QPushButton('No')

line1 = QHBoxLayout()
line2 = QHBoxLayout()
line1.addWidget(statement, alignment = Qt.AlignCenter)
line2.addWidget(btn_yes)
line2.addWidget(btn_no)
line3 = QVBoxLayout()
line3.addLayout(line1)
line3.addLayout(line2)
main_win.setLayout(line3)
main_win.show()
app.exec_()
```



Qualification

What is a global variable ?

How to create it?

Qualification

Global variable

is a variable that is visible (accessible) from any part of the program.

Local variable

is a variable that is declared and used only inside functions.

```
def setName(name, format):
    filename = name + '.' + format
    return filename

filename = setName(name, format)
print(filename)
```

Global variable

is declared in the main part of the program. Its value can be obtained and changed in any part of the program.

```
filename = ''
def setName(name, format):
    global filename
    filename = name + '.' + format
setName('pic', 'jpg')
print(filename)
```

Qualification

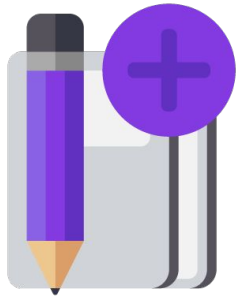
Qualifications confirmed!

Great, you are ready to brainstorming and complete your work task!

Qualification

Brainstorm:

Easy Editor interface



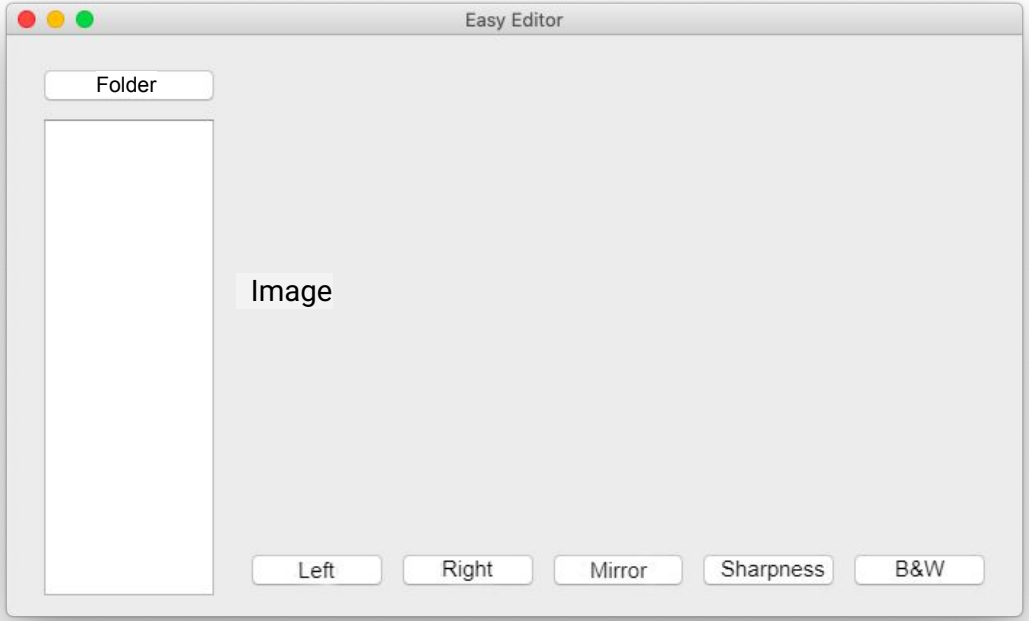
Let’s take a close look at the interface

We will program the list of images later.

Instead of a graphic file from the list, let's make the inscription “Image” for now.

Brainstorming

Let’s take a close look at the interface

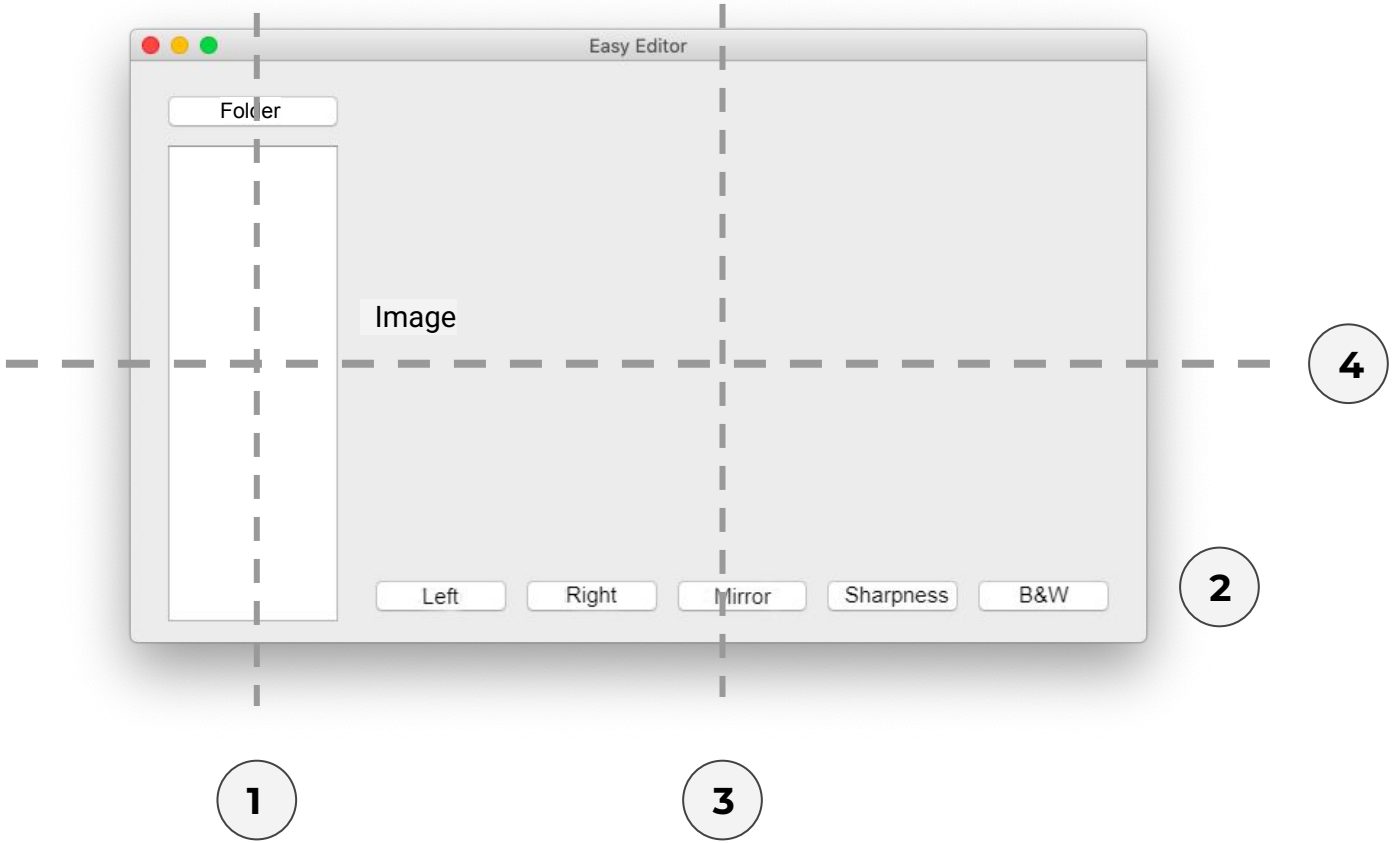


How can we program such an interface?

How to place widgets in layouts?

Brainstorming

Placing widgets



Brainstorming

Creating an interface

```
from PyQt5.QtWidgets import
```

Required PyQt modules

```
app = QApplication([])
win = QWidget()
btn_dir = QPushButton("Folder")
```

And other interface elements

```
win.resize(700, 400)
```

And other properties of elements

Creating widgets

```
row = QHBoxLayout()
```

And other layouts

```
col1.addWidget(btn_dir)
```

Placing widgets in layouts

Placing in layouts

```
win.setLayout(row)
win.show()
app.exec()
```

Placing in layouts

Brainstorming

Your task:

Program the Easy Editor app interface.

The required parameters can be found in the technical specification

Use the technical documentation from previous workdays, if needed.

Cole,
Senior Developer

Brainstorming

VS Code:

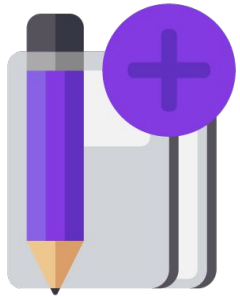
The Easy Editor app

Complete task 1 in VS Code

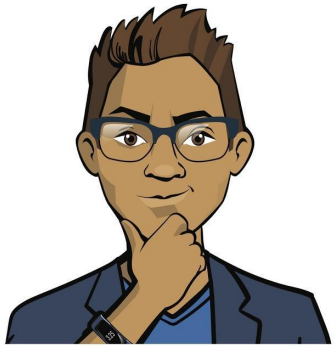
The Easy Editor app

Brainstorming:

Loading a list of graphic files



How to display the names of images from a computer folder in the app interface ?



Brainstorming

How to display the names of images from a computer folder in the app interface ?

Brainstorming

User

Computer

User clicks the “Folder” button

The user selects a folder with files

What actions should the computer take?

User sees a list of filenames

How to display the names of images from a computer folder in the app interface ?

Brainstorming

User

Computer

User clicks the “Folder” button

Open folder selection window

The user selects a folder with files

Load all files from the folder into the program

Select graphic files

Display graphic files in the list widget

User sees a list of filenames

Required functionality

Developer Cole suggested a list of the required functions. Some commands need to be learned before starting work.

1	PyQt window widget: QFileDialog	Open folder selection window
2	Module os: function listdir()	Load names of all files from the folder (into the program)
3	Your own filter() function that selects files	Select graphic files
		Display graphic files in the list widget

Brainstorming

1. Selecting a folder for work

Let's use a new widget from the PyQt5 library – QFileDialog.
It is used to call the folder selection window (File Explorer or Finder)

PyQt5:

Command	*Purpose*
`from PyQt5.QtWidgets import QFileDialog`	Connecting a widget
`workdir = QFileDialog.getExistingDirectory()`	Getting the path to the selected folder from the QFileDialog window

Brainstorming

1. Selecting a folder for work

Let's use a new widget from the PyQt5 library – QFileDialog.
It is used to call the folder selection window (File Explorer or Finder)

PyQt5:

Command	*Purpose*
`from PyQt5.QtWidgets import QFileDialog`	Connecting a widget
`workdir = QFileDialog.getExistingDirectory()`	Getting the path to the selected folder from the QFileDialog window

A **folder path** is a sequence of folder (directory) names and additional characters specifying the path to the folder.

Example: C:\User\Sasha\School\IT – '"IT" folder path

Brainstorming

1. Selecting a folder for work

Let's use a new widget from the PyQt5 library – QFileDialog.
It is used to call the folder selection window (File Explorer or Finder)

PyQt5:

Command	*Purpose*
`from PyQt5.QtWidgets import QFileDialog`	Connecting a widget
`workdir = QFileDialog.getExistingDirectory()`	Getting the path to the selected folder from the QFileDialog window

We don't need to set such paths manually. Python will do it for us.

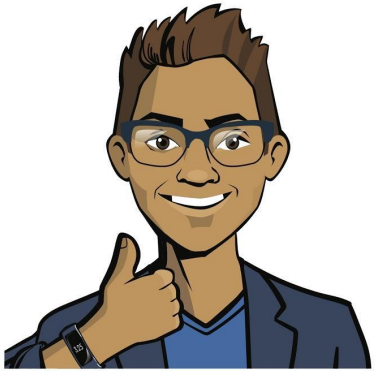
Example: C:\User\Sasha\School\IT – '"IT" folder path

Brainstorming

2. Get names of all files from the folder and load them into the program

Let’s use the os module from the Python standard library.

Out of many os functions, we'll use commands to access files and folders along the path.



Brainstorming

Module os

is in the Python standard library and contains functions for working with the operating system.

os:

Command	*Purpose*
`import os`	Connecting the os module
`filenames = os.listdir(workdir)`	Getting files from the folder (argument – path)

The folder path was obtained at the previous step

Brainstorming

Module os

is in the Python standard library and contains functions for working with the operating system.

os:

Command	*Purpose*
`import os`	Connecting the os module
`filenames = os.listdir(workdir)`	Getting files from the folder (argument – path)

‘robotics_article.doc’
‘turtles game.py’
‘vacation2020.jpg’
‘crazy_cat.gif’

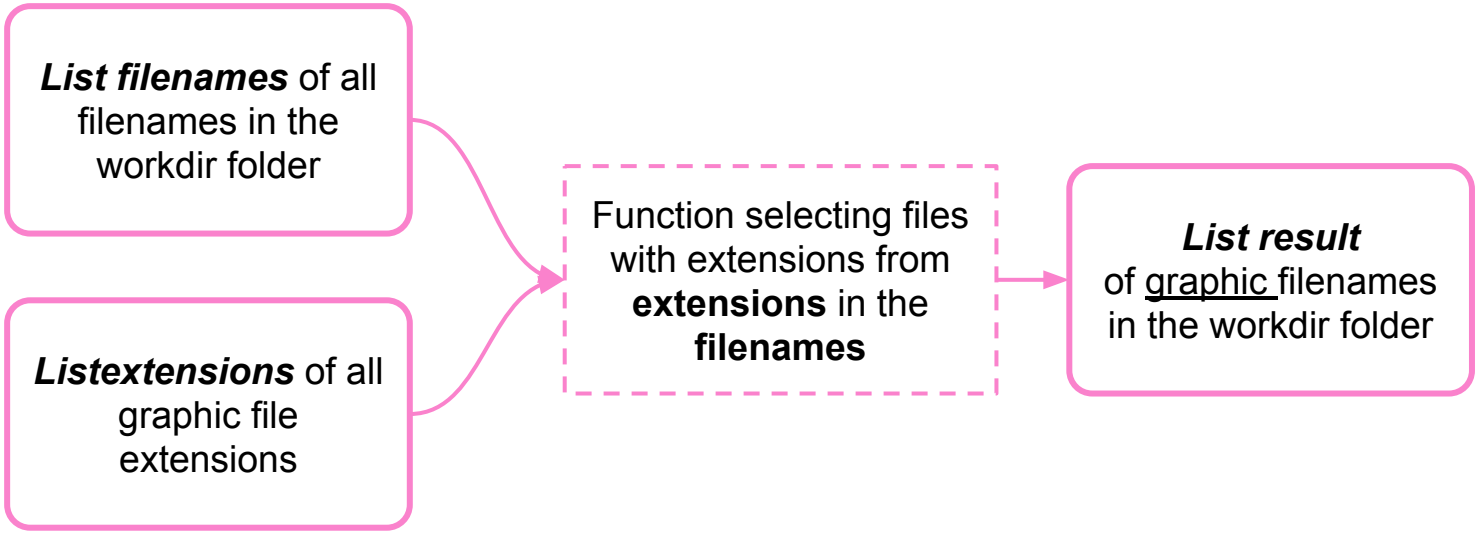
Names of different types of files were loaded.

We need to keep the graphic files only.

Brainstorming

3. Selecting graphic files

Let’s select files with graphic extensions only.
It is convenient to put all valid extensions in the **extensions** list.



Brainstorming

3. Selecting graphic files

Let's check which filenames have graphic file extensions (.jpg, .png, etc.)

```
def filter(filenames, extensions):
```

- Create an empty list result for filenames.
- *For every* filename from the filenames list:
  - *And every* extension from the extensions list:
    - **If** a name has this extension,
      then add it to the results list.
- Return the result list.

Brainstorming

3. Selecting graphic files

Let's check which filenames have graphic file extensions (.jpg, .png, etc.)

```
def filter(filenames, extensions):
```

- ❑ Create an empty list result for filenames.
- ❑ *For every* filename from the filenames list:
  - ❑ *And every* extension from the extensions list:

```
            if filename.endswith(extension):
                result.append(filename)
```

- ❑ Return the result list.

Method**endswith(<string>)**.

Returns **True** ,if the filename ends with ext, and **False**, if not.

Brainstorming

4. Displaying the name list in the widget

We have already implemented a similar task for the Smart Notes project (for a list of note titles)

```
def showFilenamesList():
```

- ❑ Select the work folder (workdir).
- ❑ Set the list of valid extensions.
- ❑ Load the folder filenames and keep only with extensions extentions.
- ❑ Clear the list widget (in case there are filenames of another folder there).
- ❑ Add the selected filenames to the widget one by one.

Brainstorming

Let's include code fragments into the program:

```
import os
from PyQt5.QtWidgets import QFileDialog
```

Connecting the rest of PyQt objects and creating an interface

```
def chooseWorkdir():
```

Function for selecting a work folder

```
def filter(files, extensions):
```

Function for selecting filenames by extensions

```
def showFilenamesList():
```

Function handling the "Folder" button click.

Responsible for choosing a folder, selecting files, and displaying them in the widget.

Uses chooseWorkdir() and filter()

```
btn_dir.clicked.connect(showFilenamesList)
```

Let's include code fragments into the program:

```
import os
from PyQt5.QtWidgets import QFileDialog
```

Connecting the rest of PyQt objects and creating an interface

```
def chooseWorkdir():
```

Function for selecting a work folder

```
def filter(files, extensions):
```

Function for selecting filenames by extensions

```
def showFilenamesList():
```

Function handling the "Folder" button click.

Responsible for choosing a folder, selecting files, and displaying them in the widget.

Uses chooseWorkdir() and filter()

```
btn_dir.clicked.connect(showFilenamesList)
```

Comment:

To work, we need not only the list of files (available in the widget) but also the work folder.

Now, the folder name is removed after showFilenamesList() finishes the operation

To save the folder, let's define it as a ***global variable***.

```
workdir = ''

def chooseWorkdir():
    global workdir
    workdir = QFileDialog...
```

```
import os
from PyQt5.QtWidgets import QFileDialog
```

Connecting the rest of PyQt objects and creating an interface

```
workdir = '' # introducing a global variable
def chooseWorkdir():
      global workdir # calling the global variable
      workdir = QFileDialog.getExistingDirectory()

def filter(files, extensions):
```

Function for selecting filenames by extensions

```
def showFilenamesList():
```

Function handling the “Folder” button click.

Responsible for choosing a folder, selecting files, and displaying them in the widget.

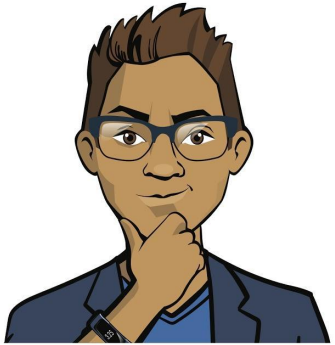
Uses chooseWorkdir() and filter()

```
btn_dir.clicked.connect(showFilenamesList)
```

The command word **global** tells the interpreter that the result should be stored in a global variable.

Task:

Write functions to select the working directory and display names of graphic files from it in the widget.



Brainstorming

Module 4. Lesson 2. The Easy Editor project. P. 1

VS Code:

The Easy Editor app

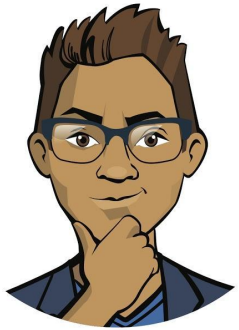
Complete task 2 in VS Code

The Easy Editor app

Wrapping up the workday

To complete, pass a technical interview:

1. Which module contains functions for working with the computer's operating system?
2. What variable is called global? Why is the workdir variable defined globally?



Cole,
Senior Developer

Emily,
Project Manager

Great job!

Dear colleagues!

You’ve done a great job today.

Next workday we will continue working on the Easy Editor app and use it to process the first photos!

Wrapping up
How tothe workday