**Редактор изображений**

Изначально проект задумывался как графический редактор. В итоге получился мини-редактор логотипов. Была мысль использовать некоторые из изученных уже изученных функций библиотеки PIL.

Проблемы были неизбежны, так как не все функции имеют возможность работать с любыми разрешениями, таким образом функция изменения прозрачности не работает с файлами разрешения «.jpg», но работает с файлами «.png», так как у них определен альфа канал, отвечающий за прозрачность.

Были созданы несколько «.py» файлов:

1. **main**: основной файл, запускающий программу
2. **image\_processing**: файл, содержащий класс Image, в котором описаны функции по изменению изображения, конвертированию цвета из RGB в hex, конвертированию изображения в байты, сохранению изображения
3. **widgets**: содержит класс Canvas, создающий объект класса, на котором размещается изображение, содержит класс TabsW, содержащий список объектов Canvas, позволяющий обратиться к конкретному объекту QtabWidget при помощи метода currentIndex
4. **windows**: содержит классы всех окон
5. **declarations**: в этом файле находятся константы, также logger, записывающий все действия в log.txt
6. **database:** имеет 1 класс DataBase, в котором переопределены магические методы \_\_init\_\_, \_\_enter\_\_, \_\_exit\_\_.

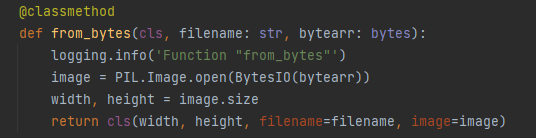


Используемые технологии:

1. **Изученные**: декораторы классов, импортирование, функции, условия, циклы, проектирование классов, возвращение,

f строки, переопределение методов, магические методы, разработка QT дизайна в designer, работа с текстовыми файлам, работа с фотографиями, работа с базами данных, библиотеки (PIL, pyQT5, datetime).

1. **Неизученные**: иконки у виджетов, библиотеки (os, logging, io), подсказки типов, создание новых диалогов, использование функций для взаимодействия с контекстными менеджерами.

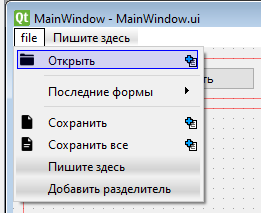


Используемые виджеты:

1. **Изученные**: QMainWindow, QVBoxLayout, QWidget, QHBoxLayout, QPushButton, QFormLayout, Qlabel, QStatusBar, QSpinbox,

QSlider, QFileDialog, QInputDialog

1. **Неизученные:** QMenu, QAction, QtabWidget

****

Используемые функции:

1. **Статические**: main (запуск программы), except\_hook (определение ошибки при аварийном завершении работы), convert\_color (переводит цвет из RGB в hex), image\_to\_bytes (переводит изображение в байты), image\_from\_bytes(переводит байты в изображение), callback (коллбэк, нужный для реализации комбинаций ctrl + z, ctrl + shift + z), showExeptDialog (вызывается при попытке обработать jpg изображение функциями, непредназначенными для этого)
2. **Класс Image:** \_\_init\_\_, from\_filename (создает объект класса Image из имени передаваемого файла), width (возвращает ширину изображения), height (возвращает высоту изображения), filename (возвращает имя файла), функции рисования: (draw\_line, draw\_ellipse, draw\_square, draw\_polygon), функции поворота: (rotate\_left, rotate\_right), resize\_image (изменяет размер изображения), quantize\_image (квантирует изображение), blur\_image (размывает изображение), функции отражения: (vertical\_reflection, horizontal\_reflection),change\_alpha (изменяет альфа канал изображения), функции изменения цветового канала для png и jpg (only\_red\_png, only\_green\_png, only\_blue\_png, to\_default\_color\_png, only\_red\_jpg, only\_green\_jpg, only\_blue\_jpg, to\_default\_color\_jpg ), save (сохранить), update (обновить базу данных), set\_new\_image (заменяет предыдущий объект Image на новый)
3. **Класс CanvasW:** \_\_init\_\_,initUI, mouseEvent (переопределяет данный метод, для построения графических примитивов), mouseReleaseEvent (запоминает координаты для построения графических примитивов), undo (реализует ctrl + z), redo (реализует ctrl + shift + z)
4. **Класс TabsW:** addTab (добавляет QTabWidget), get\_canvas (не позволяет рисовать на пустом поле)
5. **Класс MainWindow:** \_\_init\_\_, initUI, CloseEvent (закрывает скрытые окна и очищает часть базы данных), выбор цвета для рисования: choose\_main\_color, choose\_secondary\_color, ряд переопределяемых функций для привязки к кнопкам на главном окне: draw\_on\_image, change\_color, rotate\_image, reflect\_an\_image, change\_alpha, quantize\_an\_image, blur\_an\_image, open\_an\_image, open\_recent, save\_an\_image, undo, redo, save\_all
6. **Классы второстепенных окон, вызываемых для получения дополнительной информации для функций из image\_processing:** DrawWindow, ReflectWindow, AlphaWindow, ColorWindow, SizeDialog, RotateWindow