Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

Институт «Электронных и информационных систем»

Кафедра «Информационных технология и систем»

Лабораторная работа №1

**«СОЗДАНИЕ СРЕДСТВА ПРОСМОТРА ИЗОБРАЖЕНИЙ»**

Отчёт по дисциплине:

«Распознавание образов и обработка изображений»

Принял преподаватель:

\_\_\_\_\_\_\_\_\_\_ / Назаров А. Г.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Выполнил студент группы 8091:

\_\_\_\_\_\_\_\_\_\_ / Григорьев Д. И.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

Великий Новгород

2022

**Цель лабораторной работы**

1. Подготовка рабочего программного окружения для выполнения лабораторных работ;
2. Разработать и реализовать программу просмотра изображений.

**Программное окружение**

Для выполнения данной лабораторной работы было использовано следующее окружение:

1. Язык программирования: Python;
2. Операционная система: Windows 11;
3. Среда разработки: Visual Studio Code;
4. Использованные пакеты: PyQt6 (6.2.3).

**Ответы на вопросы**

**Q:** We use a message box to tell users that they are already viewing the first or last image while they are trying to see the previous one before the first image, or the next one after the last image. But there is another way to deal with this situation—disable prevAction when users are viewing the first image and disable nextAction when users are viewing the last image. How is this implemented?

**A:** Отключение кнопок происходит вызовом метода setEnabled.

**Q:** Our menu items or tool buttons only contain text. How could we add an icon image to them?

**A:** Для добавления иконки можно использовать Qt Creator (вкладка Дизайн), а также метод setIcon.

**Q:** We use QGraphicsView.scale to zoom in or out of an image view, but how do you rotate an image view?

**A:** Чтобы поворачивать картинки нужно использовать QGraphicsView.rotate(angle).

**Q:** What does moc do? What actions do the SIGNAL and SLOT macros perform?

**A:** Moc преобразует код с расширениями QT в C++ код, который может быть скомпилирован стандартными C++ компиляторами. SIGNAL и SLOT создают сигнал и слот соответственно.

**Исходный код:**

import sys

from PyQt6.QtCore import QSize, Qt, QFile, QRectF, QRegularExpression, QFileInfo, QDir

from PyQt6.QtWidgets import QApplication, QMainWindow, QGraphicsScene, QGraphicsView, QLabel, QFileDialog, QMessageBox

from PyQt6.QtGui import QAction, QPixmap

class MainWindow(QMainWindow):

    def \_\_init\_\_(self):

        super().\_\_init\_\_()

        self.setWindowTitle("Image Viewer")

        self.resize(QSize(800, 600))

        menubar = self.menuBar()

        self.file\_menu = menubar.addMenu('&File')

        self.view\_menu = menubar.addMenu('&View')

        self.file\_tool\_bar = self.addToolBar('File')

        self.view\_tool\_bar = self.addToolBar('View')

        self.image\_scene = QGraphicsScene(self)

        self.image\_view = QGraphicsView(self.image\_scene)

        self.setCentralWidget(self.image\_view)

        self.main\_status\_bar = self.statusBar()

        self.main\_status\_label = QLabel(self.main\_status\_bar)

        self.main\_status\_bar.addPermanentWidget(self.main\_status\_label)

        self.main\_status\_label.setText('Image Information will be here!')

        self.\_\_create\_actions()

    def \_\_create\_actions(self):

        self.open\_action = QAction('&Open', self)

        self.save\_as\_action = QAction('&Save as', self)

        self.exit\_action = QAction('&Exit', self)

        file\_actions = [self.open\_action, self.save\_as\_action, self.exit\_action]

        self.file\_menu.addActions(file\_actions)

        self.zoom\_in\_action = QAction('Zoom In', self)

        self.zoom\_out\_action = QAction('Zoom Out', self)

        self.prev\_action = QAction('&Previous Image', self)

        self.next\_action = QAction('&Next Image', self)

        view\_actions = [self.zoom\_in\_action, self.zoom\_out\_action, self.prev\_action, self.next\_action]

        self.view\_menu.addActions(view\_actions)

        self.file\_tool\_bar.addAction(self.open\_action)

        self.view\_tool\_bar.addActions(view\_actions)

        self.exit\_action.triggered.connect(self.close)

        self.open\_action.triggered.connect(self.\_\_open\_image)

        self.save\_as\_action.triggered.connect(self.\_\_save\_as)

        self.zoom\_in\_action.triggered.connect(self.\_\_zoom\_in)

        self.zoom\_out\_action.triggered.connect(self.\_\_zoom\_out)

        self.prev\_action.triggered.connect(self.\_\_prev\_image)

        self.next\_action.triggered.connect(self.\_\_next\_image)

        self.\_\_setup\_shortcuts()

    def \_\_open\_image(self):

        self.dialog = QFileDialog(self)

        self.dialog.setWindowTitle('Open Image')

        self.dialog.setFileMode(QFileDialog.FileMode.ExistingFile)

        self.dialog.setNameFilter(self.tr('Images (\*.png \*.bmp \*.jpg)'))

        file\_paths = []

        if self.dialog.exec():

            file\_paths = self.dialog.selectedFiles()

            self.\_\_show\_image(file\_paths[0])

    # Не работает :\

    def \_\_show\_image(self, path):

        self.image\_scene.clear()

        self.image\_view.resetTransform()

        self.image = QPixmap(path)

        self.cur\_img = self.image\_scene.addPixmap(self.image)

        self.image\_scene.update()

        self.image\_view.setSceneRect(QRectF(self.image.rect()))

        status = f'{path}, {self.image.width()}x{self.image.height()}, {QFile(path).size()} Bytes'

        self.main\_status\_label.setText(status)

        self.current\_image\_path = path

    def \_\_save\_as(self):

        try:

            self.cur\_img

        except Exception:

            QMessageBox.information(self, 'Information', 'Nothing to save')

            return

        self.dialog = QFileDialog(self)

        self.dialog.setWindowTitle('Save Image As...')

        self.dialog.setFileMode(QFileDialog.FileMode.AnyFile)

        self.dialog.setAcceptMode(QFileDialog.AcceptMode.AcceptSave)

        self.dialog.setNameFilter(self.tr('Images (\*.png \*.bmp \*.jpg)'))

        file\_names = []

        if self.dialog.exec():

            file\_names = self.dialog.selectedFiles()

            if QRegularExpression('.+\\.(png|bmp|jpg)').match(file\_names[0]):

                self.cur\_img.pixmap().save(file\_names[0])

            else:

                QMessageBox.information(

                    self, 'Information', 'Save Error: bad format or filename.')

    def \_\_zoom\_in(self):

        self.image\_view.scale(1.2, 1.2)

    def \_\_zoom\_out(self):

        self.image\_view.scale(1/1.2, 1/1.2)

    def \_\_prev\_image(self):

        try:

            self.cur = QFileInfo(self.current\_image\_path)

        except Exception:

            QMessageBox.information(self, 'Information', 'No image selected.')

            return

        self.dir = QDir(self.cur.absoluteDir())

        self.name\_filters = ['\*.png', '\*.bmp', '\*.jpg']

        self.file\_names = self.dir.entryList(

            self.name\_filters, QDir.Filter.Files, QDir.SortFlag.Name)

        idx = self.file\_names.index(self.cur.fileName())

        if idx > 0:

            self.\_\_show\_image(self.dir.absoluteFilePath(

                self.file\_names[idx - 1]))

        else:

            QMessageBox.information(

                self, 'Information', 'Current image is the last one.')

    def \_\_next\_image(self):

        try:

            self.cur = QFileInfo(self.current\_image\_path)

        except Exception:

            QMessageBox.information(self, 'Information', 'No image selected.')

            return

        self.dir = QDir(self.cur.absoluteDir())

        self.name\_filters = ['\*.png', '\*.bmp', '\*.jpg']

        self.file\_names = self.dir.entryList(

            self.name\_filters, QDir.Filter.Files, QDir.SortFlag.Name)

        idx = self.file\_names.index(self.cur.fileName())

        if idx < len(self.file\_names) - 1:

            self.\_\_show\_image(self.dir.absoluteFilePath(

                self.file\_names[idx + 1]))

        else:

            QMessageBox.information(

                self, 'Information', 'Current image is the last one.')

    def \_\_setup\_shortcuts(self):

        shortcuts = [Qt.Key.Key\_Plus, Qt.Key.Key\_Equal]

        self.zoom\_in\_action.setShortcuts(shortcuts)

        shortcuts = [Qt.Key.Key\_Minus, Qt.Key.Key\_Underscore]

        self.zoom\_out\_action.setShortcuts(shortcuts)

        shortcuts = [Qt.Key.Key\_Up, Qt.Key.Key\_Left]

        self.prev\_action.setShortcuts(shortcuts)

        shortcuts = [Qt.Key.Key\_Down, Qt.Key.Key\_Right]

        self.next\_action.setShortcuts(shortcuts)

app = QApplication(sys.argv)

window = MainWindow()

window.show()

app.exec()