Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное
учреждение высшего образования
**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

Кафедра автоматизированных систем управления (АСУ)

РАЗРАБОТКА ГРАФИЧЕСКОГО РЕДАКТОРА

ОТЧЕТ
ПО РЕЗУЛЬТАТАМ
Учебной практики:
(вид практики)
Получение первичных навыков научно-исследовательской работы (рассред.)
(тип практики)

Обучающийся гр. 430-2

_____ А.А. Лузинсан
(подпись) (И.О. Фамилия)

«___» _____ 2021 г.
(дата)

Руководитель практики
от Университета:
Старший преподаватель каф. АСУ,
(должность, ученая степень, звание)

_____ _____ А. Е. Косова
(оценка) (подпись) (И.О. Фамилия)

«___» _____ 2021 г.
(дата)

Томск 2021

Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированных систем управления (АСУ)

УТВЕРЖДАЮ
Зав. кафедрой АСУ
канд. техн. наук, доц.
В. В. Романенко

(подпись)
«___» _____ 2021 г.
(дата)

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

на Учебную практику:
(вид практики)

Получение первичных навыков научно-исследовательской работы (рассред.)
(тип практики)

студент гр. 430-2 факультета ФСУ

Лузинсан Анастасии Александровны
(Ф.И.О. студента)

1. Тема практики: Разработка графического редактора

2. Цель практики: Изучить технологии создания настольных приложений для персональных компьютеров и разобраться в методологии архитектурного построения программ с последующей поддержкой; проанализировать и исследовать виды графических редакторов и область их применения; применить полученные знания и навыки на практике

3. Задачи практики:

1) Рассмотреть возможности фреймворка Qt5 для C++;

2) Углубить свои знания по объектно-ориентированному программированию на C++;

3) Поиск среды разработки: проанализировать, сравнить, выбрать наиболее подходящую;

4) Освоить приложение Qt Creator;

5) Ознакомиться с классами QWidget, QPainter, QApplication как основными классами приложения;

6) Разработать каркас приложения;

7) Составить общий алгоритм работы приложения и взаимодействие;

8) Реализация собственной части проекта;

9) Написать отчет о проделанной работе;

10) Сделать свою часть презентации для защиты.

4. Сроки практики: с «___» _____ 2021 г. по «____» _____ 2021 г.

Совместный рабочий график (план) проведения практики

№ п/п	Перечень заданий	Сроки выполнения
1	Выбор языка программирования и фреймворка	15.09.2021
2	Определение функционала приложения	20.09.2021
3	Описание заголовочного файла класса Canvas	29.09.2021
4	Написание конструктора класса, реализация отрисовки кисти на холсте	04.10.2021
5	Привязывание функционала класса Figure к холсту	18.10.2021
6	Оптимизация метода построения и отрисовки кривой	01.11.2021
7	Создание инструмента «Ластик» и определение методов-свойств кисти	15.11.2021
8	Реализация оставшихся методов-свойств и функций	26.11.2021
9	Комплексное тестирование приложения	29.11.2021

Дата выдачи задания: «___» _____ 2021 г.

Руководитель практики от Университета

Старший преподаватель каф. АСУ _____ А. Е. Косова

(должность, ученая степень, звание) (подпись) (Ф.И.О.)

Задание приняла к исполнению: «____» _____ 2021 г.

Студент гр. 430-2 _____ Лузинсан А. А.
(подпись) (Ф.И.О.)

Оглавление

Введение **7**

1 Описание предметной области **9**

2 Обзор программного обеспечения **11**

2.1 Qt Creator **11**

2.2 Фреймворк Qt **12**

2.2.1 Общая информация о фреймворке 12

2.2.2 Описание используемых классов фреймворка 13

2.3 Язык программирования C++ **15**

2.4 GitHub **16**

3 Обзор нормативной документации **17**

4 Описание структуры разрабатываемой программы **18**

4.1 Функциональность приложения **18**

4.2 Описание класса MainWindow **21**

4.3 Описание класса Canvas **22**

4.3.1 Отрисовка «Кривой» 22

4.3.2 Отрисовка «Фигуры» 23

4.3.3 Отрисовка «Ластика» 24

4.3.4 Отрисовка «Текста» 25

4.3.5 Сеттеры и модификаторы 25

4.4 Описание класса Figure **28**

4.5 Описание класса Manual **28**

4.6 Описание класса Feedback **29**

5 Описание маркетинговых исследований **30**

5.1 Бесплатные векторные редакторы **30**

5.1.1 Gravit Designer 30

5.1.2 Vectr 31

5.1.3 BoxySVG 31

5.2 Бесплатные растровые редакторы 31
5.2.1 GIMP 31
5.2.2 Photo Pos Pro 32
5.2.3 Paint.NET 32
5.3 Бесплатные редакторы ведения конспектов 33
5.3.1 Nebo 33
5.3.2 OneNote 33
5.3.3 MetaMoJi Note Lite 34
5.4 Место нашего продукта на рынке 34
5.4.1 Приложение общественного достояния 35
6 Описание команды разработчиков и их функций в проекте 38
Заключение 39
Список использованных источников 40
Приложения 43
Приложение А 43
Приложение Б 46
Приложение В 54
Приложение Г 55
Приложение Д 56

Введение

Графический редактор позволяет пользователю создавать и редактировать изображения непосредственно на экране компьютера. В случае с рассмотрением векторного графического редактора, то взаимодействие при таком способе происходит именно с объектами, описанными математически. Поэтому создание, отображение и редактирование происходит с достаточно высокой точностью и без потери качества при сжатии и растяжении объекта, а также других преобразований. Это позволяет сохранять информационный смысл после различных трансформациях объекта, что полезно при создании чертежей, схем или различных иллюстраций.

Для облегчения задачи по созданию приложений на сегодняшний день существует достаточно много сред разработки с удобными отладчиками, сборщиками, графическими дизайнерами и т.д. для комфортной работы с фреймворком, который в свою очередь также позволяет использовать уже готовые классы для создания собственных дочерних классов с определёнными и переопределёнными методами, реализовывающими требуемый для разработчика функционал приложения.

Работа по созданию графического редактора – это задача по созданию собственных классов, взаимодействующих друг с другом, которые имеют поля и методы, корректно взаимодействующие между собой и с пользователем.

Графический редактор требуется не только инженерам, но и иллюстраторам, так как он может использоваться для создания и редактирования рисунков, в которых существуют чёткие контуры, такие как: эмблемы, иллюстрации к книге, визитки и плакаты, этикетки, схемы и многое другое. Так как векторные рисунки состоят из отдельных графических объектов, то они легко редактируются. Таким образом каждый их объектов

может быть легко выделен, перемещён, удалён, увеличен или уменьшен, причём способы выделения также могут быть различны. Но в любом случае в основном происходит активация всего объекта-изображения, если не настроено иначе.

По завершению текущей учебной практики у нашей команды должно выйти в релиз приложение, способное реализовывать основные функции по созданию изображения, оформленный в дружелюбный для пользователя интерфейс.

1 Описание предметной области

Графический редактор – программа (или пакет программ), позволяющая создавать, просматривать, обрабатывать и редактировать цифровые изображения (рисунки, картинки, фотографии) на компьютере.

Виды графических редакторов:

1. Векторные редакторы обычно более пригодны для создания разметки страниц, типографики, логотипов, sharp-edged artistic иллюстраций (например, мультипликация, clip art, сложные геометрические шаблоны), технических иллюстраций, создания диаграмм и составления блок-схем.

2. Растровые редакторы больше подходят для обработки и ретуширования фотографий, создания фотореалистичных иллюстраций, коллажей, и создания рисунков от руки с помощью графического планшета.

Векторные графические редакторы позволяют пользователю создавать и редактировать векторные изображения непосредственно на экране компьютера, а также сохранять их в различных векторных форматах, например, CDR, AI, EPS, WMF или SVG.

Векторные редакторы часто противопоставляют растровым редакторам. В действительности их возможности часто дополняют друг друга:

Последние версии растровых редакторов (таких, как GIMP или Photoshop) предоставляют пользователю и векторные инструменты (например, изменяемые кривые), а векторные редакторы (CorelDRAW, Adobe Illustrator, Xara Designer, Adobe Fireworks, Inkscape, SK1 и другие) реализуют и растровые эффекты (например, заливку), хотя иногда и несколько ограниченные по сравнению с растровыми редакторами

Основные инструменты векторных редакторов

1. Кривые Безье — позволяют создавать прямые, ломаные и гладкие кривые, проходящие через узловые точки, с определёнными касательными в этих точках;
2. Заливка — позволяет закрашивать ограниченные области определённым цветом или градиентом;
3. Текст создаётся с помощью соответствующего инструмента, а потом часто преобразуется в кривые, чтобы обеспечить независимость изображения от шрифтов, имеющихся (или отсутствующих) на компьютере, используемом для просмотра;
4. Набор геометрических примитивов;
5. Карандаш — позволяет создавать линии «от руки». При создании таких линий возникает большое количество узловых точек, от которых в дальнейшем можно избавиться с помощью «упрощения кривой».

Векторный графический редактор является наиболее удачным средством управления цельными фигурами и объектами особенно в целях написания рукописных заметок в компьютерной среде. На данный момент всё чаще встаёт вопрос о переходе на преимущественно дистанционное обучение в учебных заведениях. В связи с этим требуется приложение с функционалом, которое было бы заточено под подобные цели осуществления приемлемого редактирования и оформления конспектов, чертежей, иллюстраций с учётом возможности добавления и редактирования различных форм записи, фигур, опции рисования, стирания и т.д. Мир меняет свой поток событий, в соответствии с этим способ обучения тоже не стоит на месте: рукописные схемы, графики, рисунки, чертежи и, собственно, конспекты теперь изменяют свой формат с недолговечных бумажных листов в легкодоступный цифровой формат. Теперь не нужно будет покупать тонны расходного материала в виде ручек, паст, ластиков, корректоров, стикеров и макулатуры, что к тому же благоприятно скажется на экологии нашей планеты.

2 Обзор программного обеспечения

2.1 Qt Creator

Qt Creator (ранее известная под кодовым названием Greenhouse) — кроссплатформенная свободная IDE для разработки на C, C++, javascript и QML. Разработана Trolltech (Digia) для работы с фреймворком Qt. Включает в себя графический интерфейс отладчика и визуальные средства разработки интерфейса как с использованием qtwidgets, так и QML. Поддерживаемые компиляторы: GCC, Clang, mingw, MSVC, Linux ICC, GCCE, RVCT, WINSCW

Особенности:

Основная задача Qt Creator — упростить разработку приложения с помощью фреймворка Qt на разных платформах. Поэтому среди возможностей, присущих любой среде разработки, есть и специфичные, такие как отладка приложений на QML и отображение в отладчике данных из контейнеров Qt, встроенный дизайнер интерфейсов: как на QML, так и на qtwidgets.

Работа с проектами:

Qt Creator поддерживает системы сборки qmake, cmake, autotools, с версии 2.7 qbs. Для проектов, созданных под другими системами, может использоваться в качестве редактора исходных кодов. Есть возможность редактирования этапов сборки проекта.

Также IDE нативно поддерживает системы контроля версии, такие как Subversion, Mercurial, Git, CVS, Bazaar, Perforce. Начиная с версии 2.5, в поле комментария к правке поддерживается автодополнение.

Редактирование кода:

В Qt Creator реализовано автодополнение, в том числе ключевых слов, введённых в стандарте C++11 (начиная с версии 2.5), подсветка кода (её определение аналогично таковому в Kate, что позволяет создавать свои виды

подсветок или использовать уже готовые). Также, начиная с версии 2.4, есть возможность задания стиля выравнивания, отступов и постановки скобок.

2.2 Фреймворк Qt

2.2.1 Общая информация о фреймворке

Qt (произносится [ˈkjuːt] (кьют) как «cute» или неофициально Q-T (кью-ти, ку-тэ)) — фреймворк для разработки кроссплатформенного программного обеспечения на языке программирования C++. Для многих языков программирования существуют наборы библиотеки, позволяющие использовать преимущества Qt: Python — PyQt, PySide; Ruby — QtRuby; Java — Qt Jambi; PHP — PHP-Qt и другие.

Со времени своего появления в 1996 году библиотека легла в основу многих программных проектов. Кроме того, Qt является фундаментом популярной рабочей среды KDE, входящей в состав многих дистрибутивов Linux.

Функции и состав:

Qt позволяет запускать написанное с его помощью программное обеспечение в большинстве современных операционных систем путём простой компиляции программы для каждой системы без изменения исходного кода. Включает в себя все основные классы, которые могут потребоваться при разработке прикладного программного обеспечения, начиная от элементов графического интерфейса и заканчивая классами для работы с сетью, базами данных и XML. Является полностью объектно-ориентированным, расширяемым и поддерживающим технику компонентного программирования.

Отличительная особенность — использование метаобъектного компилятора — предварительной системы обработки исходного кода. Расширение возможностей обеспечивается системой плагинов, которые возможно размещать непосредственно в панели визуального редактора.

Также существует возможность расширения привычной функциональности виджетов, связанной с размещением их на экране, отображением, перерисовкой при изменении размеров окна.

2.2.2 Описание используемых классов фреймворка

2.2.2.1 Класс QGraphicsScene

Класс служит контейнером для QGraphicsItems. Он используется вместе с QGraphicsView для визуализации графических элементов, таких как линии, прямоугольники, текст или даже пользовательские элементы, на 2D-поверхности. QGraphicsScene является частью платформы графического представления.

QGraphicsScene также предоставляет функциональные возможности, которые позволяют эффективно определять как местоположение элементов, так и определять, какие элементы видны в произвольной области сцены. С помощью виджета QGraphicsView вы можете либо визуализировать всю сцену, либо увеличить масштаб и просмотреть только ее части.

2.2.2.2 Класс QColor

Цвет обычно указывается в терминах компонентов RGB (красный, зеленый и синий), но его также можно указать в терминах компонентов HSV (оттенок, насыщенность и значение) и CMYK (голубой, пурпурный, желтый и черный). Кроме того, цвет можно указать с помощью названия цвета. Имя цвета может быть любым из названий цветов SVG 1.0.

Конструктор QColor создает цвет на основе значений RGB. Чтобы создать QColor на основе значений HSV или CMYK, используются функции toHsv() и toCmyk() соответственно. Эти функции возвращают копию цвета в нужном формате. Кроме того, статические функции fromRgb(), fromHsv() и fromcmyk() создают цвета из указанных значений. В качестве альтернативы, цвет может быть преобразован в любой из трех форматов с помощью функции ConvertTo() (возвращающей копию цвета в желаемом формате) или

любой из функций setRGB (), setHsv() и setCmyk(), изменяющих формат этого цвета. Функция spec() сообщает, как был указан цвет.

Цвет можно задать, передав строку RGB (например, "#112233") или строку ARGB (например, "#ff112233") или имя цвета (например, "синий") в функцию setNamedColor(). Названия цветов взяты из названий цветов SVG 1.0. Функция name() возвращает название цвета в формате "#RRGGBB". Цвета также можно задать с помощью setRGB(), setHsv() и setCmyk(). Чтобы получить более светлый или темный цвет, используйте функции lighter() и darker() соответственно.

2.2.2.3 Класс QPolygon

Объект QPolygon является объектом QVector<QPoint>. Самый простой способ добавить точки в QPolygon - использовать оператор потоковой передачи векторов.

В дополнение к функциям, предоставляемым QVector, QPolygon предоставляет некоторые функции, базирующиеся на точках.

Каждую точку в полигоне можно получить, передав ее индекс функции point(). Для заполнения полигона QPolygon, предоставляется функция setPoint(), которая устанавливает точки с заданным индексом, а также функция setPoints() для установки всех точек в полигоне (изменяется его размер до заданного количества точек) и функция putPoints(), которая копирует количество заданных точек в полигон из указанного индекса (при необходимости изменяет размер полигона).

Полигон предоставляет функции boundingRect() и translate() для функций геометрии.

2.2.2.4 Класс QLine

QLine описывает линию конечной длины (или отрезок) на двумерной поверхности. Начальная и конечная точки линии задаются с использованием целочисленной координаты.

Положения начальной и конечной точек линии могут быть получены с помощью функций p1(), x1(), y1(), p2(), x2() и y2(). Функции dx() и dy() возвращают горизонтальную и вертикальную составляющие линии. Используя функцию isNull() можно определить, представляет ли линия допустимую линию или вырожденная линию.

2.2.2.5 Класс QInputDialog

Входным значением может быть строка, число или элемент из списка. Должна быть установлена метка, указывающая пользователю, что он должен ввести.

Предусмотрены пять статических функций: getText(), getMultiLineText(), getInt(), getDouble() и getItem().

2.3 Язык программирования C++

C++ — компилируемый, статически типизированный язык программирования общего назначения.

Поддерживает такие парадигмы программирования, как процедурное программирование, объектно-ориентированное программирование, обобщённое программирование. Язык имеет богатую стандартную библиотеку, которая включает в себя распространённые контейнеры и алгоритмы, ввод-вывод, регулярные выражения, поддержку многопоточности и другие возможности. C++ сочетает свойства как высокоуровневых, так и низкоуровневых языков. В сравнении с его предшественником — языком C — наибольшее внимание уделено поддержке объектно-ориентированного и обобщённого программирования.

C++ широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования. Область его применения включает создание операционных систем, разнообразных прикладных программ, драйверов устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также игр. Существует

множество реализаций языка C++, как бесплатных, так и коммерческих и для различных платформ. Например, на платформе x86 это GCC, Visual C++, Intel C++ Compiler, Embarcadero (Borland) C++ Builder и другие. C++ оказал огромное влияние на другие языки программирования, в первую очередь на Java и C#.

Синтаксис C++ унаследован от языка C. Одним из принципов разработки было сохранение совместимости с C. Тем не менее C++ не является в строгом смысле надмножеством C; множество программ, которые могут одинаково успешно транслироваться как компиляторами C, так и компиляторами C++, довольно велико, но не включает все возможные программы на C.

2.4 GitHub

GitHub — крупнейший веб-сервис для хостинга IT-проектов и их совместной разработки.

Веб-сервис основан на системе контроля версий Git и разработан на Ruby on Rails и Erlang компанией GitHub, Inc. Сервис бесплатен для проектов с открытым исходным кодом и – с 2019 года – небольших частных проектов, предоставляя им все возможности, а для крупных корпоративных проектов предлагаются различные платные тарифные планы.

3 Обзор нормативной документации

Отчёт выполнен в соответствии с требованиями ОС ТУСУР-2013[12].

4 Описание структуры разрабатываемой программы

4.1 Функциональность приложения

Функциональность данного графического редактора предоставляется двух различных панелях: в главном меню и в панели инструментов.

Главное меню подразделяется на такие разделы, как:

- Файл;
- Вид;
- Сведения.

Раздел «Файл» подразумевает под собой возможность осуществления таких базовых операций над файлами, как:

- Создание файла – Очищает холст от всех объектов, если они были на нём, тем самым подготавливая творческую среду;
- Открытие существующего файла – вызывает диалоговое окно для выбора желаемого файла;
- Сохранение файла – вызывает диалоговое окно для выбора пути сохранения файла. Требуется указать полное название файла вместе с расширением графического файла, иначе изображение не будет сохранено.

Раздел «Вид» предоставляет возможность выставления:

- Заднего фона в зависимости от выбранного цвета кисти (по умолчанию устанавливается градиентная заливка);
- Разлиновки в зависимости от выставленного размера кисти (чем больше размер кисти, тем шире расстояние между линиями разлиновки).

Также предусмотрены информационные блоки, расположенные в разделе «Сведения», такие как:

- Руководство пользователя (оно же - Мануал);
- Обратная связь с разработчиками

Панель инструментов подразделяется на такие типы, как:

- Функции-свойства. Они способны менять свойства самой кисти;
- Инструменты отрисовки. Одна кисть может отрисовывать только определённый вид фигуры, выбранный из данной панели.

В функции-свойства кисти входят:

- Различные стили штриха: «Прямая линия», «Пунктирная линия», «Точечная линия», «Линия тире-точка», «Линия тире-точка-точка», «Широкий пунктир». Выбранный стиль штриха распространяется как на разлиновку холста, отрисовку кривой линии, так и на контур отображаемых фигур (если выбрана опция заливки фигуры), либо на стиль линии у фигур (в противном случае);
- Выбор размера кисти. С помощью счётчика (SpinBox) выставляется целочисленное значение, обозначающее ширину обрисовываемой кисти. Размер учитывается в случае использования инструмента разлиновки, как было сказано выше; а также в стандартных инструментах отрисовки кривой и фигур. Важным моментом является то, что при выставлении инструмента вставки текста, размер шрифта выставляется в соответствии с выбранным размером кисти;
- Палитра. Палитра задаёт цвет кисти. Выбранный цвет распространяется на способ разлиновки холста, отрисовки кривой и фигур, а также цвет текста, при выборе инструмента вставки текста на холст;
- Градиент. Данная опция выставляет в качестве цвета кисти градиентную заливку. Свойство распространяется также на цвет заливки фигур, если оное выбрано пользователем;
- Заполнение. Заливка подразумевает под собой заполнение выбранным цветом (из палитры, либо градиентом) полых фигур;

- Дублирование. Свойство позволяет отрисовывать множество фигур, размер которых определяется стартовой точкой, куда пользователь изначально нажал левой кнопкой мыши и текущей точкой, где в данный момент находится курсор мыши.

Среди инструментов отрисовки можно выбрать:

- Кисть. Базовый инструмент, позволяющий рисовать кривые линии. На неё распространяются все функции-свойства, перечисленные выше.
- Ластик. Также базовый инструмент, на который распространяются свойства «Размер кисти», «Стиль штриха». При изменении инструмента на любой другой, в качестве цвета выставляется предыдущий цвет (если был выбран градиент, то снова выставляется градиент);
- Вставка текста. После выбора данного инструмента пользователю требуется нажать кнопкой мыши (неважно правой, или левой) в желаемом месте холста. Далее будет вызвано диалоговое окно, в котором можно написать некоторый текст. По нажатию кнопки «Ок», на холсте будет напечатано введённое сообщение. Есть возможность написания длинных текстов, имеющих символы переноса строки. Отображение в ширину будет таким же, какое имеется в диалоговом окне. Размер шрифта зависит от выставленного размера кисти. Цвет текста также можно выставить с помощью функции-свойства «Палитра», однако с помощью «Градиента» задать цвет текста не имеется возможным (цвет будет оставлен тем же, что и был выставлен до этого). Таким образом, текст будет отрисован в области, левый верхний угол которой начинается в точке куда нажал пользователь.
- Различные фигуры. В качестве выбора пользователю предоставлены такие фигуры как: «Отрезок», «Прямоугольник»,

«Эллипс», «Равнобедренный треугольник», «Прямоугольный треугольник», «Ромб», «Равнобедренная трапеция», «Пятиугольник». На данные фигуры распространяются такие свойства, как: «Размер кисти» - задаёт ширину отрисовываемой линии у фигуры, «Цвет кисти» - выставляется «Палитрой», либо «Градиентом»; «Заливка» - заполняет полые фигуры (все, кроме «Отрезка») выбранным для кисти цветом (в том числе Градиентом), при этом окантовка фигур задаётся таким же цветом, который был выбран ранее; «Стиль штриха» - свойство, задаваемое для стиля окантовки фигур, вне зависимости от заполнения фигуры, «Дублирование» - свойство отрисовки, позволяющая дублировать множество однотипных фигур, выбранных пользователем. При совместном применении инструмента фигуры и свойств «Дублирование», «Заполнение» и «Градиент» пользователь может получить весьма любопытные результаты, аналогичные 3D объектам. Внимание: данная комбинация ресурсозатратна при выборе достаточно большого размера кисти и/или дублировании больших фигур «Пятиугольник».

Ещё базовые функции:

- Удалить всё. Очищает холст, от всех объектом на нём.
- Выход. Останавливает работу приложения и удаляет приложение из реестра активных задач.

Функциональная схема[25] XPainter’а представлена в приложениях В, Г и Д.

4.2 Описание класса MainWindow

Класс MainWindow наследуется от QMainWindow и является связующим звеном между классами Canvas, Figure, Manual и Feedback. В

конструкторе данного класса установлены все существующие кнопки приложения. При нажатии пользователем на ту или иную кнопку, в конструкторе класса активируются сигналы, которые перенаправляют действие приложения на соответствующие слоты. Далее происходит обработка функций, делегирующая своё выполнение в требуемый класс.

4.3 Описание класса Canvas

Класс Canvas наследуется от класса QGraphicsScene и является классом, предоставляющим возможность взаимодействия пользователя и непосредственно холста графического редактора. Объект «Холст» инициализируется изначально в конструкторе класса MainWindow, и является основным звеном, уничтожаясь только по завершению цикла работы приложения.

При первом обращении курсора мыши к холсту ничего не происходит, так как инструмент для отрисовки не был задан. Только при выборе инструмента, из класса MainWindow поступает сигнал об активации того или иного слота. Слот устанавливает значение в переменную-переключатель для инструмента и теперь, при обращении курсора мыши к холсту, будет отрисована «Линия», «Ластик», «Фигура», либо «Текст».

В зависимости от инструмента, поведение обработчиков событий различно, поэтому имеет смысл описать каждый инструмент по-отдельности.

4.3.1 Отрисовка «Кривой»

При обработке события «Нажатие мыши» в области холста, происходит установка булевой переменной, отвечающей за инструмент «Ластик» в состояние False. Далее, для корректной обработки сценария, заключающегося в том, что предыдущим инструментом был «Ластик», следующим этапом является установка в качестве текущей кисти конфигурации предыдущей кисти. После оных манипуляций, настройка кисти завершается и в вектор точек добавляется текущая точка.

Обработка события «Передвижение мыши» подразумевает под собой два различных сценария – с дублированием и без дублирования.

Сценарий без дублирования переопределяет текущих последний отрезок между предпоследней точкой, составленной по расположению курсора мыши пользователя на предыдущей итерации и последней, то есть текущей точкой, составленной также по расположению курсора мыши на текущей обработке события «Передвижение мыши». Далее в вектор точек заносится новая точка, и таким образом последней точкой теперь является текущая, обеспечивая тем самым непрерывное обновление положения стартовой позиции для линии. После, в вектор линий добавляется составленная линия и на завершающем этапе на холст добавляется текущая линия.

Сценарий без дублирования отличается лишь тем, что при отрисовке новой линии старая точка не обновляется, и каждый раз новая линия составляется из стартовой точки, то есть инициализированной при обработке события «Нажатие мыши» и текущей точки, составленной из координат расположения мыши пользователя на текущий момент.

Обработка события «Отпускание мыши» добавляет заполненный вектор линий в полилинию, тем самым сохраняя контроль над объектом даже после завершения отрисовки линии. Это может пригодиться в дальнейшем на этапе модификации приложения и добавлении нового функционала, по типу реализации «Отмены», «Лассо» и т.д. После занесения вектора линий в контейнер линий, текущий вектор очищается.

4.3.2 Отрисовка «Фигуры»

Обработка события «Нажатие мыши» в области холста, как и случае отрисовки «Линии», устанавливает булеву переменную инструмента «Ластик» в состояние False. Далее также настраивается текущая кисть свойствами предыдущей кисти. Отличительной особенностью отрисовки

фигуры является то, что для корректного построения фигуры, требуется лишь запомнить начальное положение отрисовки и далее, инициализировать объект фигуры конструктором по умолчанию, который задаёт фиктивный элемент. Всё это потребуется в последующем при обработке события «Передвижение мыши».

Обработка события «Передвижение мыши» также подразделяется на два сценария, но отличие не разительное и состоит в том, удалять ли предыдущую отрисованную фигуру, или нет. Если выбрано свойство «Дублирование», то блок с условием не выполняется, а значит отрисованная на предыдущей итерации фигура не удаляется, обеспечивая тем самым дублирование фигур, различающиеся только лишь диаметром, то есть расстоянием от стартовой точки до текущей. После блока с условием идёт непосредственное запоминание текущей точки абсциссы и ординаты. Далее созданный классом Figure объект сохраняется в переменной и фигура отрисовывается на холсте.

Обработка события «Отпускание мыши» устанавливает указатель переменной фигуры в nullptr.

4.3.3 Отрисовка «Ластика»

Обработка события «Нажатие мыши» в области холста сначала проверяет, не был ли выбран в качестве предыдущего инструмента «Ластик», так как далее происходит обновление переменной, отвечающая за текущий цвет, последним значимым цветом. Если предыдущим инструментом не был «Ластик», то выполняется оная манипуляция и далее устанавливается белый цвет кисти. Булева переменная обновляется до True. В вектор точек заносится стартовая точка на холсте. Далее инструмент ластик будет отрисовываться аналогично инструменту «Кривая», с той лишь разницей, что цвет кисти является белым, а последний запомненный цвет установлен в значении последнего значимого для отрисовки фигур цвета.

4.3.4 Отрисовка «Текста»

Обработка события «Нажатие мыши» в области холста инициализирует начальную точку координатами абсциссы и ординат. Далее инициализируется булева переменная, отвечающая за взаимодействие кнопки «Ок». Следующим этапом является вызов диалогового окна посредством статического метода getMultiLineText класса QInputDialog, который принимает в качестве параметров текущий виджет, «Заголовочный текст», который будет отображён в шапке окна, «Заголовок», отображаемый над текстовым полем, «Текст по умолчанию», представляющий из себя примерный текст для вставки и булева переменная «Ok», выставленная заранее по-умолчанию True. Следующим черёдом настраивается шрифт, который будет применён к отображаемому тексту с помощью класса QFont. Шрифту устанавливается: размер равный ширине междустрочных линий (если они были отрисованы, в противном случае остаётся минимальная ширина равна 1) минус коэффициент 5; стиль шрифта устанавливается в QFont::SansSerif. Далее, методом addSimpleText, с атрибутами QString строки text, инициализированной диалоговым окном, и QFont стиля font, модифицированной методами настройки размера и стиля шрифта, на холст добавляется многострочный текст и из метода обратно возвращается объект класса QGraphicsSimpleTextItem, который инициализирует объект textItem. Данному тексту применяется метод установки кисти, который устанавливает в качестве текущего цвета цвет кисти. Завершающим этапом является перемещение отрисованного текста на позицию по координатам текущей точки нажатия на холст. События «Перемещение мыши» и «Отпускания мыши» не происходят, так как диалоговое окно вызывается незамедлительно после нажатия мыши на холст.

4.3.5 Сеттеры и модификаторы

Среди типов сеттеров данного класса различают такие типы, как:

- Сеттеры инструмента: «Кривая», «Фигура», «Ластик», «Текст». В соответствии с выбранным инструментом вызывается слот, который устанавливает переменной-переключателю, отвечающей за инструмент требуемое значение;
- Сеттеры фигуры: «Отрезок», «Прямоугольник», «Эллипс», «Равнобедренный треугольник», «Прямоугольный треугольник», «Ромб», «Равнобедренная трапеция», «Пятиугольник». В соответствии с выбранной фигурой вызывается слот, который устанавливает переменной-переключателю, отвечающей за фигуру требуемое значение;
- Сеттер дублирования. Устанавливает булевой переменной, отвечающей за состояние дублирования значение противоположное от текущего состояния этой же переменной;
- Сеттер заполнения полых фигур. Устанавливает булевой переменной, отвечающей за состояние заливки значение противоположное от текущего состояния этой же переменной;
- Сеттер градиента. Запоминает текущий цвет кисти, если предыдущим инструментом не был выбран «Ластик». Устанавливает булевой переменной, отвечающей за состояние градиента значение противоположное от текущего состояния этой же переменной. Значение булевой переменной, отвечающей за состояние «Ластика», устанавливается значением False.

Среди типов модификаторов данного класса различают такие типы, как:

- Модификаторы стиля отрисовки линии: «Прямая линия» - Qt::SolidLine, «Пунктирная линия» - Qt::DashLine, «Точечная линия» - Qt::DotLine, «Линия точка-тире» - Qt::DashDotLine, «Линия точка-точка-тире» - Qt::DashDotDotLine, «Широкий пунктир». «Широкий пунктир» заслуживает отдельного

внимания, так как данный стиль составлен вручную: изначально объявляется вектор типа qreal; инициализируется qreal переменная значением, равному размеру кисти, умноженному на коэффициент 10. Далее в вектор линий заносится 1 пиксель и только что проинициализированная переменная, отвечающая за длину отступа между чередующимися линиями. На завершающем этапе кисти устанавливается шаблонных стиль методом setDashPattern();

- Модификатор размера кисти. Данный слот вызывается каждый раз, когда в MainWindow поступает сигнал от том, что значение SpinBox было изменено. В соответствии с переданным значением нового размера кисть модифицируем свой размер.

Помимо прочего, класс Canvas содержит методы «Установки заднего фона», «Отрисовки разлиновки», «Удалить всё», «Загрузка изображения», «Сохранение изображения».

«Установка заднего фона» сначала очищает холст от всех объектов, расположенных на нём. Далее вызывается метод, устанавливающий фигуру «Прямоугольник» для переменной-переключателя. Используя класс Figure, инициализируем переменную background конструктором с переданными параметрами: 0, 0, ширина холста, высота холста, что, в целом, будут означать левую верхнюю и правую нижнюю точки; выбранная фигура (прямоугольник), кисть, True (gradient) и True (fillPath). Возвращенный прямоугольник отрисовывается на холсте.

«Отрисовка разлиновки» сначала инициализирует новую кисть объектом текущей кисти. Междустрочная ширина разлиновки устанавливается значением ширины текущей кисти плюс 15. Далее ширина текущей кисти переопределяется значением 1. Запускается цикл от 0 до высоты холста с шагом равным междустрочной ширине разлиновки. На каждой итерации отрисовывается линия с начальной точкой, имеющей

координату абсцисс, равной 0, и ординат, равной индексу текущей строки по высоте; и конечной точкой, имеющей координату абсцисс, равной ширине холста, и ординат, равной такому же индексу строки. В качестве пятого параметра передаётся текущая кисть с размером 1. После завершения цикла отрисовки, конфигурация кисти, а именно размер, восстанавливается значением инициализированной изначально переменной.

Метод «Удалить всё» удаляет все объекты на холсте, тем самым очищая холст.

Реализация методов «Загрузка изображения» и «Сохранение изображения» была делегирована Фадругим разработчиком.

4.4 Описание класса Figure

Класс Figure наследуется от классов QObject и QGraphicsItem и является классом, предоставляющим готовые фигуры, в зависимости от переданных координат двух точек на холсте. Обязательным фактором является активная область для создаваемой фигуры, которая зависит не только от координат левой верхней и правой нижней точек, но и от размера кисти. Также, в качестве свойства фигуры, в класс Figure принимается булевы параметры градиента и заполнения фигуры, которые учитываются при непосредственной отрисовке фигуры. Причём градиентный объект, хранящийся в переменной класса QLinearGradient, инициализируется непосредственно в функции отрисовки фигур.

Так как класс является наследником класса QGraphicsItem, то в будущем можно будет добавить возможность перемещения и изменения объектов, находящихся на холсте.

4.5 Описание класса Manual

Класс Manual наследуется от класса QWidget и является информационным окном, предоставляющим сведения о функционале графического редактора. Данный класс содержит в себе обозреватель текста,

который отображает HTML-документ, представляющий из себя руководство пользователя, а также контактные данные разработчиков.

4.6 Описание класса Feedback

Класс Feedback наследуется от класса QWidget и является диалоговым окном, предоставляющим возможность взаимодействия с разработчиками посредством отправки сообщения на электронный адрес разработчикам в целях улучшения и обновления данного продукта. В качестве требуемых данных выступают: «Имя», «Email» и «Сообщение». Поле «Сообщение» допускает написание многострочного текста. На данном этапе разработки данные «Обратной связи» записываются в текстовый файл, который можно просмотреть в директории, расположенной в пути установки приложения.

5 Описание маркетинговых исследований

Проанализировав и исследовав различные графические редакторы, можно выяснить, что существуют различные программы и сервисы для работы с векторной, растровой и 3D-графикой на разных платформах.

Самые продвинутые графические редакторы вроде инструментов компании Adobe стоят немалых денег. Но есть бесплатные и вполне достойные альтернативы, возможностей которых хватит большинству пользователей.

Среди таких можно выделить бесплатные векторные, растровые редакторы.

5.1 Бесплатные векторные редакторы

Предназначены для создания и редактирования логотипов, интерфейсов и прочей масштабируемой графики.

5.1.1 Gravit Designer

- Платформы: веб, Windows, macOS, Linux.

Gravit Designer[13], ранее известный как Gravit, — это полнофункциональный векторный редактор. Он подходит для любых задач — от дизайна интерфейсов и иконок до работы с презентациями, иллюстрациями и анимацией.

Аккуратный интуитивный интерфейс Gravit Designer можно настраивать под себя. Графический редактор содержит массу инструментов для создания прекрасных детализированных векторных изображений. Среди них — неразрушающие (их действие можно отменять) функции для булевых операций, инструменты «Нож» и «Граф путей», множество режимов заливки и смешивания, а также мощный текстовый движок.

Если понадобится получить доступ к работе на ходу, облачный сервис Gravit Cloud позволит вернуться к проекту на любом устройстве.

5.1.2 Vectr

- Платформы: веб, Windows, macOS, Linux.

Vectr[14] предлагает все функции, которые только могут понадобиться для создания векторной графики, а также множество опций для использования фильтров, теней и шрифтов. Этого хватит, чтобы справиться с большинством повседневных дизайнерских задач. Особенно полезны возможности совместного редактирования и синхронизации, благодаря которым можно работать когда и где угодно в тандеме с другими людьми.

5.1.3 BoxySVG

- Платформа: веб.

Простой инструмент для создания масштабируемой векторной графики, который пригодится как начинающим веб-дизайнерам и разработчикам, так и профессионалам. BoxySVG[15] запускается прямо в браузере и справляется со своими задачами не хуже настольных редакторов.

Приложение отличается интуитивным интерфейсом и высокой скоростью работы. Арсенал настроек и функций не так велик, как у профессионального софта, но включает все необходимые инструменты, среди которых карандаши, кисти, текст, клонирование, формы и многое другое. BoxySVG поддерживает горячие клавиши и все популярные форматы файлов для экспорта готовых проектов.

5.2 Бесплатные растровые редакторы

Предназначены для создания и редактирования любых не масштабируемых рисунков и фотографий.

5.2.1 GIMP

- Платформы: Windows, macOS, Linux.

Бесплатный графический редактор с открытым исходным кодом. GIMP[16] укомплектован богатым набором функций для рисования,

цветокоррекции, клонирования, выделения, улучшения и других действий. Интерфейсом GIMP отличается от популярнейшего Photoshop, но долго искать нужные инструменты вам не придётся.

Команда GIMP позаботилась о совместимости, так что вы сможете без проблем работать со всеми популярными форматами изображений. Кроме того, здесь встроен файловый менеджер, похожий на Bridge из программ от компании Adobe.

5.2.2 Photo Pos Pro

- Платформа: Windows.

Если вы работаете на Windows и не нуждаетесь в таком количестве инструментов, как у GIMP, вашим идеальным редактором может стать Photo Pos Pro[17]. Последний создан с прицелом на операции с изображениями и отлично справляется с типичными задачами вроде регулировки контрастности, освещения и насыщенности. Но Photo Pos Pro подходит и для более сложных манипуляций.

Эта программа может похвастать очень дружелюбным интерфейсом и детальной справкой, которая помогает разобраться новичкам. Если вы хотите сделать Photo Pos Pro ещё функциональнее, к вашим услугам множество расширений и плагинов.

5.2.3 Paint.NET

- Платформа: Windows.

Paint.NET[18] является альтернативой программе Paint, встроенной во все версии Windows. Но пусть схожесть названий не сбивает вас с толку: это гораздо более продвинутый и полезный редактор.

Команда разработки делает упор на простоту использования и совершенствует в Paint.NET скорее функции для редактирования снимков, чем возможности дизайна графики. Тем не менее Paint.NET позволяет

управлять перспективой, манипулировать пикселями на холсте, клонировать выделенные зоны и так далее.

Благодаря поддержке слоёв, широкому выбору инструментов для выделения и настроек вроде яркости, контрастности и кривых, Paint.NET можно рассматривать как достойную замену Photoshop.

5.3 Бесплатные редакторы ведения конспектов

5.3.1 Nebo

Nebo[19] позволяет вести заметки более эффективно, интеллектуально и удобно в любой сфере, от совещаний до аудиторий, от планирования проектов до записей в дневнике..

Редактируйте и форматируйте с помощью Surface Pen – функции рукописного ввода позволяют писать, добавлять и удалять написанное, применять форматирование и стили, используя только стилус.

Разнообразное содержимое: создавайте интерактивные диаграммы с элементами, которые можно редактировать, удалять и перемещать. Пишите от руки редактируемые формулы, вычисляйте результат и преобразуйте

Рисуйте от руки эскизы и подписывайте изображения.

5.3.2 OneNote

OneNote[20] позволяет создавать неограниченное количество записных книжек, которые, в свою очередь, можно разбивать на страницы и разделы — всё это помогает выстраивать многоуровневую структуру данных и грамотно организовывать информацию. Поддерживаются функции рукописного ввода и рисования, можно вставлять видео из Интернета и снимки экрана, записывать звуковые заметки, а также добавлять документы Word, таблицы Excel и презентации PowerPoint как в виде вложенных файлов, так и в формате виртуальных распечаток (изображений). Важные и актуальные записи можно помечать тегами для последующего быстрого доступа к ним, а для записных книжек с конфиденциальной информацией предусмотрена

защита паролем. Также OneNote имеет развитые средства поиска и позволяет делиться заметками с сотрудниками, друзьями и близкими.

5.3.3 MetaMoJi Note Lite

MetaMoJi Note[21] – это кроссплатформенное приложение для создания заметок и примечаний к PDF-документам, а также цифровой альбом для ваших зарисовок. Фиксируйте свои идеи в любое время с помощью улучшенного распознавания ручного письма и конвертации рукописи в набранный текст или создания примечаний в импортированных PDF-документах. MetaMoJi Note – это виртуальная маркерная доска для создания зарисовок, примечаний, цифрового гибрида письменных заметок, а также для занятия скрапбукингом.

MetaMoJi Note ¬– это единственное приложение для создания заметок, доступное для всех основных мобильных платформ. Ему присвоено множество наград: Tabby Award в категории «Лучшее приложение для повышения персональной производительности» - Silver Stevie® Award в категории «Международный бизнес» - Финалист конкурса Appy Award в категории «Высокая производительность» - Приложение для повышения производительности №1 в Японии

5.4 Место нашего продукта на рынке

Лицензия на программное обеспечение[22] — это правовой инструмент, определяющий использование и распространение программного обеспечения, защищённого авторским правом. Обычно лицензия на программное обеспечение разрешает получателю использовать одну или несколько копий программы, причём без лицензии такое использование рассматривалось бы в рамках закона как нарушение авторских прав издателя. По сути, лицензия выступает гарантией того, что издатель ПО, которому принадлежат исключительные права на программу, не подаст в суд на того, кто ею пользуется.

Лицензии на программное обеспечение в целом делятся на две большие группы: несвободные (собственнические, они же проприетарные; и полусвободные) и лицензии свободного и открытого ПО. Их различия сильно влияют на права конечного пользователя в отношении использования программы.

Свободное программное обеспечение[23], свободный софт — программное обеспечение, пользователи которого имеют права («свободы») на его неограниченную установку, запуск, свободное использование, изучение, распространение и изменение (совершенствование), а также распространение копий и результатов изменения. Если на программное обеспечение есть исключительные права, то свободы объявляются при помощи свободных лицензий.

Наше приложение является приложением общественного достояния.

5.4.1 Приложение общественного достояния

Вообще говоря, произведению, находящемуся в общественном достоянии (public domain), не требуется каких-либо лицензий. Любая лицензия — это ограничение, а общественное достояние предполагает отсутствие каких-либо ограничений на использование произведения. Но все же одно ограничение имеется: такое произведение никто не имеет права присваивать себе в исключительную собственность, то есть ограничивать права других на использование этого произведения.

Правовые системы современных стран зачастую «по умолчанию» считают владельцем имущественных авторских прав на созданное произведение то лицо, творческим трудом которого произведение было создано. Поэтому для корректного освобождения своего произведения — для перевода его в общественное достояние — необходимы формальные инструменты, с помощью которых можно «уведомить» правовые системы

стран о своем отказе от авторского права и прочих смежных правах на произведение в пользу общества.

Перечислим существующие инструменты перевода приложения в общественное достояние[24].

5.4.1.1 CC0

Популярным и самым проработанным инструментом для отказа от своих авторских прав и передачи произведения в общественное достояние является инструмент CC0 (CC Zero), разработанный организацией Creative Commons. Это универсальный, действующий во всем мире инструмент. Воспользоваться им может только владелец авторских прав. Для его использования достаточно указать на связь своего произведения с CC0. Вы также можете разместить специальную метку на веб-странице произведения, информирующую пользователей и поисковые машины о том, что ваше произведение является общественным достоянием.

Фонд свободного программного обеспечения рекомендует для перевода программ в общественное достояние пользоваться инструментом CC0.

5.4.1.2 Unlicense

Unlicense — еще один популярный инструмент для перевода произведения в общественное достояние. Для его использования совместно со своим произведением вместо обычных файлов COPYING или LICENSE, нужно разместить файл UNLICENSE, содержащий специальный текст «нелицензии», информирующий о передаче произведения в общественное достояние и об отказе от прав и ответственности

5.4.1.3 Public Domain Mark

Дополнительно стоит сказать о Public Domain Mark. Public Domain Mark — это специальная метка, которая ставится на чужие произведения,

уже находящиеся в общественном достоянии. Ее нельзя использовать для освобождения собственных произведений, но ей можно помечать произведения других авторов, о которых достоверно известно, что они находятся в общественном достоянии. Благодаря этой метке такие произведения можно легко находить через специальные поисковые системы, например, расширенный поиск Google.

5.4.1.4 Другие способы уведомления о переводе произведения в общественное достояние

Нет ограничений в формах выражения своего решения о передачи произведения в общественное достояние. Хотя они и не всегда могут соответствовать законодательным нормам. Например, вы можете воспользоваться кратким уведомлением от Creative Commons, специальным шаблоном Wikipedia или следующим сообщением:

«Этот продукт является ОБЩЕСТВЕННЫМ ДОСТОЯНИЕМ и может быть использован КАК ЕСТЬ, со всеми достоинствами и недостатками, полностью или частично, кем угодно и в каких угодно целях БЕЗ КАКИХ-ЛИБО ОГРАНИЧЕНИЙ.»

«This product is PUBLIC DOMAIN and may be used AS IS, with all advantages and faults, in whole or in part, by anyone for any purpose, WITHOUT ANY CONDITIONS».

6 Описание команды разработчиков и их функций в проекте

Наша команда состоит из трех человек.

Куминов Павел Алексеевич, студент каф. АСУ, гр. 430-4. Занимался проектированием приложения, разработкой класса Figure, реализацией функций манипулирования с файлами.

Завзятов Владислав Сергеевич, студент каф. АСУ, гр. 430-4. Занимался разработкой классов MainWindow, Manual, Feedback.

Лузинсан Анастасия Александровна, студентка каф. АСУ, гр. 430-2. Занималась разработкой класса Canvas, а также маркетинговыми исследованиями в области графических редакторов.

Заключение

В результате данного проекта я изучила технологии создания настольных приложений для персональных компьютеров и разобралась в методологии архитектурного построения программ с последующей поддержкой, а также проанализировала и исследовала виды графических редакторов и область их применения.

В ходе выполнения проекта были получены следующие результаты:

1. Было рассмотрено устройство графических редакторов.
2. Я освежила в памяти основные аспекты языка программирования C++, а также освоила некоторые моменты, связанные с объектно-ориентированным программированием, в частности наследованием, иерархией классов и виртуальными функциями.
3. Нашла среду разработки: проанализировала, сравнила, выбрала наиболее подходящую.
4. Освоила приложение *Qt Creator*.
5. Ознакомилась с фреймворком Qt5.
6. Разработала структуру приложения.
7. Составила общий алгоритм работы приложения.
8. Реализовала свою часть проекта.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Анализ и исследование графических редакторов. [Электронный ресурс]. Режим доступа: https://www.stud24.ru/programming-computer/analiz-i-issledovanie-graficheskih-redaktorov/380651-1211331-page1.html (дата обращения 15.09.2021)

2. Графический редактор — Википедия. [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Графический_редактор (дата обращения 16.09.2021)

3. Растровый графический редактор — Википедия. [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Растровый_графический_редактор (дата обращения 16.09.2021)

4. Векторный графический редактор — Википедия. [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Векторный_графический_редактор (дата обращения.17.09.2021)

5. Графический редактор. – Информатика. [Электронный ресурс]. Режим доступа: http://psk68.ru/files/metod/uchebnik_Informatika/graf.html (дата обращения 21.09.2021)

6. C++ — Википедия. [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/C++ (дата обращения 23.09.2021)

7. Qt — Википедия. [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Qt (дата обращения 25.09.2021)

8. Qt Creator — Википедия. [Электронный ресурс]. Режим доступа: https://ru.wikipedia.org/wiki/Qt_Creator (дата обращения 26.09.2021)

9. Qt Documentation | Home. [Электронный ресурс]. Режим доступа: https://doc.qt.io/ (дата обращения 26.09 2021)

10. Уроки программирования на языке C++ — Ravesli. [Электронный ресурс]. Режим доступа: https://ravesli.com/uroki-cpp/ (дата обращения 23.09.2021)

11. Qt 5.3. Профессиональное программирование на C++. / М. Шлее – СПб.: БХВ-Петербург, 2015. – 928 с.: ил. – (в подлиннике).

12. Образовательный стандарт ВУЗа. [Электронный ресурс]. Режим доступа: https://regulations.tusur.ru/storage/58109/obrazovatelnyy_standart_vuza_os_tusur_01-2013_raboty_studencheskie_po_napravleniyam_podgotovki_i_spetsialnostyam_tehnicheskogo_profilya_obschie_trebovaniya_i_pravila_oformleniya.pdf?1490862364 (дата обращения 02.12.2021)

13. Векторный редактор — Gravit Designer [Электронный ресурс]. Режим доступа: https://www.designer.io/en/?x-clickref=1011liDZVszr (дата обращения 15.09.2021)

14. Векторный редактор — Vectr [Электронный ресурс]. Режим доступа: https://vectr.com/ (дата обращения 15.09.2021)

15. Векторный редактор — BoxySVG [Электронный ресурс]. Режим доступа: https://boxy-svg.com/ (дата обращения 15.09.2021)

16. Растровый редактор — GIMP [Электронный ресурс]. Режим доступа: https://www.gimp.org/ (дата обращения 16.09.2021)

17. Растровый редактор — Photo Pos Pro [Электронный ресурс]. Режим доступа: http://www.photopos.com/PPP3_BS/Default.aspx (дата обращения 16.09.2021)

18. Растровый редактор — Paint.NET [Электронный ресурс]. Режим доступа: https://www.getpaint.net/ (дата обращения 16.09.2021)

19. Редактор ведения заметок — Nebo [Электронный ресурс]. Режим доступа: https://www.nebo.app/ru/ (дата обращения 17.09.2021)

20. Цифровая записная книжка — Microsoft OneNote [Электронный ресурс]. Режим доступа: https://www.microsoft.com/ru-ru/microsoft-365/onenote/digital-note-taking-app?ms.url=onenotecom&rtc=1 (дата обращения 17.09.2021)

21. Редактор ведения заметок — MetaMoJi Note Lite [Электронный ресурс]. Режим доступа: https://www.microsoft.com/ru-ru/p/metamoji-note-lite/9wzdncrfjcpk?activetab=pivot:overviewtab (дата обращения 17.09.2021)

22. Википедия — Лицензия на программное обеспечение [Электронный ресурс] https://ru.wikipedia.org/wiki/Лицензия_на_программное_обеспечение (01.12.2021)

23. Википедия — Свободное программное обеспечение [Электронный ресурс] https://ru.wikipedia.org/wiki/Свободное_программное_обеспечение (дата обращения 01.12.2021)

24. Habr — Инструменты для перевода произведений в общественное достояние [Электронный ресурс] https://habr.com/ru/post/239523/ (дата обращения 01.12.2021)

25. PlanHammer [Электронный ресурс] https://planhammer.io/ (дата обращения 22.09.2021)

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А

(обязательное)

Листинг заголовочного файла класса Canvas

```
#ifndef CANVAS_H
#define CANVAS_H
#include "figure.h"
#include <QGraphicsScene>
#include <QColor>
#include <QPolygon>
#include <QLine>
#include <QString>
#include <QLinearGradient>

class Canvas : public QGraphicsScene
{
   Q_OBJECT
public:
   explicit Canvas(QObject *parent = 0);
   ~Canvas();
   QPen pen = QPen(Qt::black, 2, Qt::SolidLine);
   QColor currentColor = QColor(Qt::black);
   QPolygon *pointsOfLine = new QPolygon();
   QLine *current_line = new QLine;
   QList<QLine> *polyline = new QList<QLine>;
   QList<QList<QLine>> *polylines = new QList<QList<QLine>>;
   Figure *figure = new Figure();
   Figure *background = nullptr
```

```
    bool wasEraser = false;
    bool duplication = false;
    bool fillFigure = false;
    bool gradient = false;
public slots:
    void setSolidLine();
    void setDashLine();
    void setDotLine();
    void setDashDotLine();
    void setDashDotDotLine();
    void setCustomLine();
    void deleteAll();
    void setSize(int size);
    void setToolLine();
    void setToolDuplication();
    void setFillFigure();
    void setGradient();
    void setToolFigure();
    void setToolEraser();
    void setToolText();
    void drawGridLines();
    void setLine();//1
    void setRectangle();//2
    void setCircle();//3
    void setTriangle();//4
    void setTriangleRectangular();//5
    void setRhomb();//6
    void setTrapezoid();//7
    void setPentagon();//8
```

```
    void setBackground();//9
    void loadPicture(QString path);
    void savePicture(QString path);
private:
    void drawRectangle();
    void drawCircle();
    void drawTriangle();
    void drawTriangleRectangular();
    void drawRhomb();
    void drawTrapezoid();
    void drawPentagon();
    qreal x1=0, y1=0, x2=this->width(), y2=this->height();
    qreal widthGrid = 1;
    int chosenInstrument = 0;
    int chosenFigure = 0;
public:
    void mousePressEvent(QGraphicsSceneMouseEvent * event);
    void mouseMoveEvent(QGraphicsSceneMouseEvent *event);
    void mouseReleaseEvent(QGraphicsSceneMouseEvent *event);
};
#endif // CANVAS_H
```

ПРИЛОЖЕНИЕ Б

(обязательное)

Листинг файла с реализацией класса Canvas

```
#include "canvas.h"
#include "mainwindow.h"
#include <QInputDialog>
#include <iostream>

Canvas::Canvas(QObject *parent) : QGraphicsScene(parent)
{
   pen.setColor(Qt::black);
   pen.setStyle(Qt::SolidLine);
   pen.setCapStyle(Qt::RoundCap);
   pen.setWidth(1);
}
Canvas::~Canvas(){}

void Canvas::loadPicture(QString path)
{
   QGraphicsPixmapItem * image = new QGraphicsPixmapI-
tem(QPixmap(path));
   int imageWidth = image->pixmap().width();
   int imageHeight = image->pixmap().height();
   image->setOffset(-imageWidth / 2, -imageHeight / 2);
   image->setPos(0, 0);
   this->addItem(image);
}
void Canvas::savePicture(QString path)
```

```
{
    QString file = path;
    const qreal width = this->width();
    const qreal height = this->height();
    QImage image(width, height, QImage::Format_ARGB32);
    image.fill(Qt::white);
    QPainter _painter(&image);
    this->render(&_painter);
    image.save(file);
}

void Canvas::setSolidLine()
{ pen.setStyle(Qt::SolidLine); }
void Canvas::setDashLine()
{ pen.setStyle(Qt::DashLine); }
void Canvas::setDotLine()
{ pen.setStyle(Qt::DotLine); }
void Canvas::setDashDotLine()
{ pen.setStyle(Qt::DashDotLine); }
void Canvas::setDashDotDotLine()
{ pen.setStyle(Qt::DashDotDotLine); }
void Canvas::setCustomLine()
{
    QVector<qreal> dashes;
    qreal space = 10*pen.width();
    dashes << 1 << space;
    pen.setCosmetic(true);
    pen.setMiterLimit(5);
    pen.setDashPattern(dashes);
```

```
}
void Canvas::setToolLine()
{ chosenInstrument = 1; }
void Canvas::setToolFigure()
{  chosenInstrument = 2; }
void Canvas::setToolDuplication()
{ duplication = !duplication; }
void Canvas::setFillFigure()
{
   fillFigure = !fillFigure;
}
void Canvas::setGradient()
{
   if(!wasEraser)
       currentColor = pen.color();

   gradient = !gradient;
   wasEraser = false;
}
void Canvas::setToolText()
{ chosenInstrument = 4; }
void Canvas::setToolEraser()
{  chosenInstrument = 3; }
void Canvas::setLine()
{ chosenFigure = 1; }
void Canvas::setRectangle()
{ chosenFigure = 2; }
void Canvas::setCircle()
{ chosenFigure = 3; }
```

```
void Canvas::setTriangle()
{ chosenFigure = 4; }
void Canvas::setTriangleRectangular()
{ chosenFigure = 5; }
void Canvas::setRhomb()
{ chosenFigure = 6; }
void Canvas::setTrapezoid()
{ chosenFigure = 7; }
void Canvas::setPentagon()
{ chosenFigure = 8; }
void Canvas::setBackground()
{
   clear();
   update();
   setRectangle();
   background =  new Figure(0, 0,
                        width(), height(),
                        chosenFigure,
                        pen, true, true);
   addItem(background);
   background = nullptr;
   gradient = false;
}
void Canvas::drawGridLines()
{
   QPen background_pen = pen;
   widthGrid = pen.width()+15;
   pen.setWidth(1);
   for(int i = 0; i < this->height(); i += widthGrid)
```

```
            this->addLine(0, i, width(), i, pen);
        pen.setWidth(background_pen.width());
    }
    void Canvas::setSize(int size)
    {
        pen.setWidth(size);
    }


    void Canvas::deleteAll()
    {
        clear();
    }

    void Canvas::mousePressEvent(QGraphicsSceneMouseEvent *event)
    {
        switch(chosenInstrument)
        {
        case 1: // кисть и дублирование
            wasEraser = false;
            pen.setBrush(currentColor);
            pointsOfLine->append(event->scenePos().toPoint());    //    Сохраняем
координаты точки нажатия

            break;
        case 3: // ластик
            if(!wasEraser)
            {
                currentColor = pen.color();
```

```
            pen.setColor(Qt::white);
            wasEraser = true;
        }
        pointsOfLine->append(event->scenePos().toPoint());  // Сохраняем координаты точки нажатия
        break;
    case 2:
        wasEraser = false;
        pen.setBrush(currentColor);
        x1 = event->scenePos().x();
        y1 = event->scenePos().y();
        figure =  new Figure();
        break;
    case 4:
    {
        x1=event->scenePos().x();
        y1=event->scenePos().y();
        bool ok=true;
        QString text = QInputDialog::getMultiLineText(event->widget(),
                                                      tr("Введите текст"),
                                                      tr("Текст: "),
                                                      "Приятного времени суток\nВводите здесь свой текст ;)",
                                                      &ok);
        QFont font;
        font.setPixelSize(widthGrid-5);
        font.setStyleHint(QFont::SansSerif);
        QGraphicsSimpleTextItem *textItem = this->addSimpleText(text, font);
```

```
            textItem->setBrush(pen.brush());
            textItem->moveBy(event->scenePos().x(),event->scenePos().y());
            break;
        }
        }
    }

    void Canvas::mouseMoveEvent(QGraphicsSceneMouseEvent *event)
    {
        switch (chosenInstrument)
        {
        case 1: // Рисование
            if(duplication)// лучи
            {
                current_line = new QLine(pointsOfLine->last(), event->scenePos().toPoint());
                polyline->append(*current_line);
                addLine(*current_line, pen);
                break;
            }
        case 3: // и ластик
            current_line = new QLine(pointsOfLine->last(), event->scenePos().toPoint());
            pointsOfLine->append(event->scenePos().toPoint());
            polyline->append(*current_line);
            addLine(*current_line, pen);
            break;
        case 2: // фигура
            if(!duplication)
```

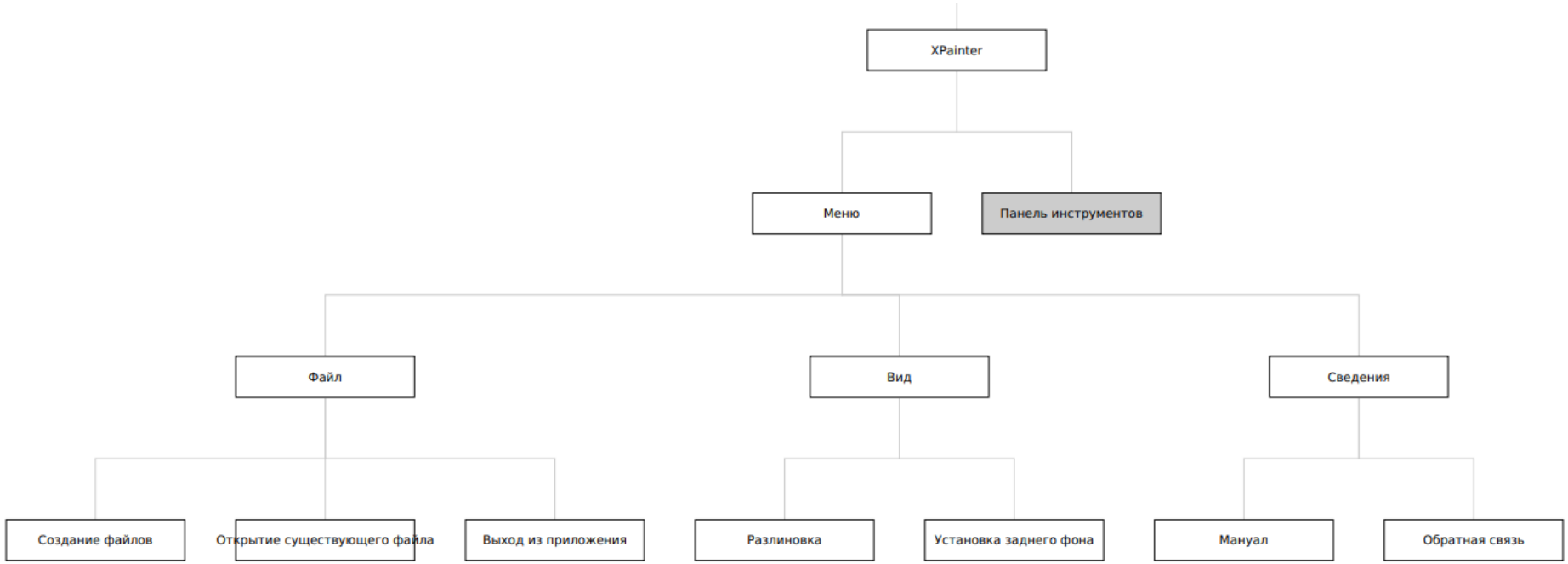
```
            figure->~Figure();
        x2 = event->scenePos().x();
        y2 = event->scenePos().y();
        figure =  new Figure(x1, y1,
                             x2, y2,
                             chosenFigure,
                             pen, gradient, fillFigure);
        addItem(figure);
    }
}


void Canvas::mouseReleaseEvent(QGraphicsSceneMouseEvent *event)
{
    Q_UNUSED(event);
    switch (chosenInstrument)
    {
    case 1: // Рисование
        polylines->append(*polyline);
        polyline->clear();
        break;
    case 2: // фигура
        figure = nullptr;
        break;
    case 3:
        polyline->clear();
    }
}
```

Приложение В

(необязательное)

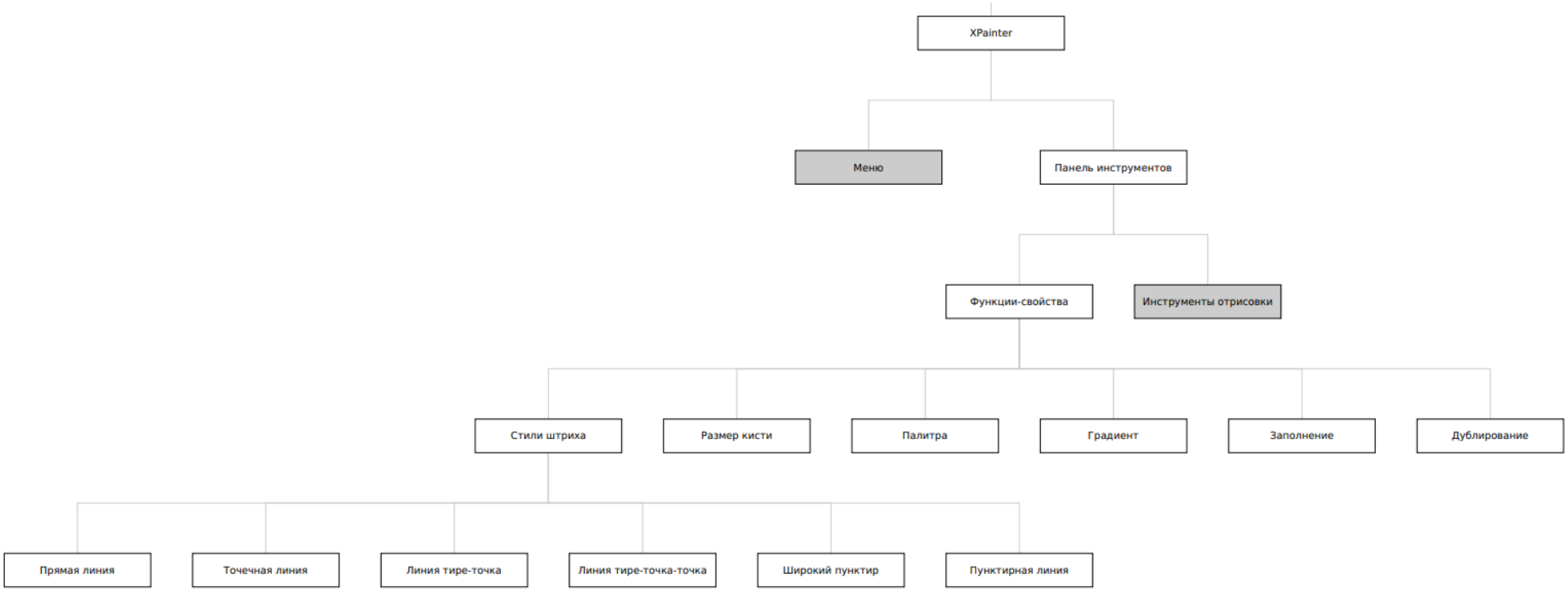
Функциональная схема меню приложения.



ПРИЛОЖЕНИЕ Г

(необязательное)

Функциональная схема функций-свойств панели инструментов приложения.



Приложение Д

(необязательное)

Функциональная схема инструментов отрисовки панели инструментов приложения.