

**Министерство образования и науки Российской Федерации  
Федеральное государственное автономное образовательное  
учреждение высшего профессионального образования  
«Севастопольский государственный университет»**

**ИССЛЕДОВАНИЕ ПРИНЦИПОВ РАБОТЫ  
ПЛАТФОРМЫ QT  
И СРЕДЫ РАЗРАБОТКИ QT CREATOR**

**Методические указания**  
к лабораторной работе по дисциплине  
**«Кроссплатформенное программирование»**  
для студентов, обучающихся по направлению  
**09.03.02 “Информационные системы и технологии”**  
очной и заочной форм обучения

**Севастополь  
2018**

УДК 004.415.2

**Исследование принципов работы платформы Qt и среды разработки Qt Creator.** Методические указания/Сост. Строганов В.А. – Севастополь: Изд-во СевГУ, 2018.–19 с.

Методические указания предназначены для оказания помощи студентам при выполнении лабораторных работ по дисциплине «Кроссплатформенное программирование».

Методические указания составлены в соответствии с требованиями программы дисциплины «Кроссплатформенное программирование» для студентов направления 09.03.02 и утверждены на заседании кафедры «Информационные системы»,  
протокол № от « » 2018 г.

## Содержание

1. Цель работы	4
2. Общие положения	4
3. Основные теоретические положения	4
3.1. Фреймворк Qt	4
3.2. Состав библиотеки Qt	5
4. Описание лабораторной установки	5
4.1. Создание проекта Qt	6
5. Порядок выполнения лабораторной работы	14
5. Содержание отчета	15
6. Контрольные вопросы	15
Библиографический список	15
Приложение	16

## 1. ЦЕЛЬ РАБОТЫ

Исследование технологии подготовки и выполнения программ в интегрированной среде Qt Creator. Получение базовых навыков работы с фреймворком Qt.

## 2. ОБЩИЕ ПОЛОЖЕНИЯ

Целью лабораторных работ является приобретение навыков проектирования и разработки кросс-платформенных приложений на основе фреймворка Qt. В ходе выполнения работ студенты знакомятся с основными механизмами, обеспечивающими функционирование Qt-приложений. В качестве лабораторной установки используется персональный компьютер и интегрированная среда разработки Qt Creator. На работы №№1 — 6 выделяется по 5 академических часов, работа №7 выполняется за 6 часов. В ходе самостоятельной подготовки студенты изучают основные теоретические положения, связанные с выполняемой работой. В ходе аудиторного лабораторного занятия разрабатывается программа по варианту задания. По результатам выполнения лабораторной работы оформляется отчет. Защита лабораторной работы проводится в виде устного опроса.

## 3. ОСНОВНЫЕ ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

### 3.1. Фреймворк Qt

Qt представляет собой фреймворк для разработки кроссплатформенных приложений. Основным языком программирования, применяемым для создания Qt-приложений, является C++. Также имеются реализации для Ruby – QtRuby, для Python – PyQt, PHP – PHP-Qt и других языков программирования.

Qt представляет собой полностью объектно-ориентированный, кроссплатформенный фреймворк. Дает возможность разрабатывать платформонезависимое ПО, написанный код можно компилировать для Linux, Windows, Mac OS X и других операционных систем. Qt включает в себя множество классов для работы с сетью, базами данных и проч., а также для создания графического интерфейса пользователя.

Qt использует МОС (Meta Object Compiler) для предварительной компиляции программ. Исходный текст программы обрабатывается МОС, который ищет в классах программы макрос Q\_ОБЪЕКТ и переводит исходный код в мета-объектный код, после чего мета-объектный код компилируется компилятором C++. МОС расширяет функциональность фреймворка, благодаря ему добавляются такие понятия, как слоты и сигналы.

В Qt имеется огромный набор виджетов (Widget), таких как: кнопки, прогресс бары, переключатели, checkbox, и другие – они обеспечивают стандартную функциональность GUI (графический интерфейс пользователя).

Позволяет использовать весь функционал пользовательского интерфейса – меню, контекстные меню, drag&drop.

Это позволяет создавать пользовательский интерфейс, который имеет одинаковый внешний вид в различных операционных системах.

Для разработки в Qt существует среда разработки Qt Creator. Qt Creator предоставляет кроссплатформенную, полностью интегрированную среду разработки (IDE) для создания приложений для множества настольных и мобильных платформ. Он доступен для операционных систем Linux, Mac OS X и Windows. Он включает в себя Qt Designer, с помощью которого можно создавать графический интерфейс. Визуальное создание интерфейса позволяет легко и просто создавать интерфейс, перетаскивая различные виджеты(выпадающие списки, кнопки, переключатели) на форму.

Qt поставляется вместе с Qt Assistant – интерактивным справочником, содержащим в себе информацию по работе с Qt. В состав Qt также входит Qt Linguist, позволяющий локализировать приложение для разных языков.

### 3.2. Состав библиотеки Qt

Библиотека Qt состоит из различных модулей, которые подключаются при помощи директивы `#include`. В состав входят:

- QtCore – классы ядра библиотеки Qt, они используются другими модулями.
- QtGui – модуль содержит компоненты графического интерфейса.
- QtNetwork – модуль содержит классы для работы с сетью. В него входят классы для работы с протоколами FTP, HTTP, IP и другими.
- QtOpenGL – модуль содержит классы для работы с OpenGL
- QSql – содержит классы для работы с различными базами данных с использованием языка SQL.
- QtSvg – содержит классы, позволяющие работать с данными Scalable Vector Graphics (SVG)
- QDom – классы для работы с XML
- QtScript – классы для работы с Qt Scripts

Имеются и другие модули.

## 4. ОПИСАНИЕ ЛАБОРАТОРНОЙ УСТАНОВКИ

В качестве лабораторной установки для выполнения работ используется компьютер с установленным ПО – интегрированной средой разработки Qt Creator.

Процесс установки Qt Creator подробно рассмотрен в приложении. После запуска попадаем в главное меню.

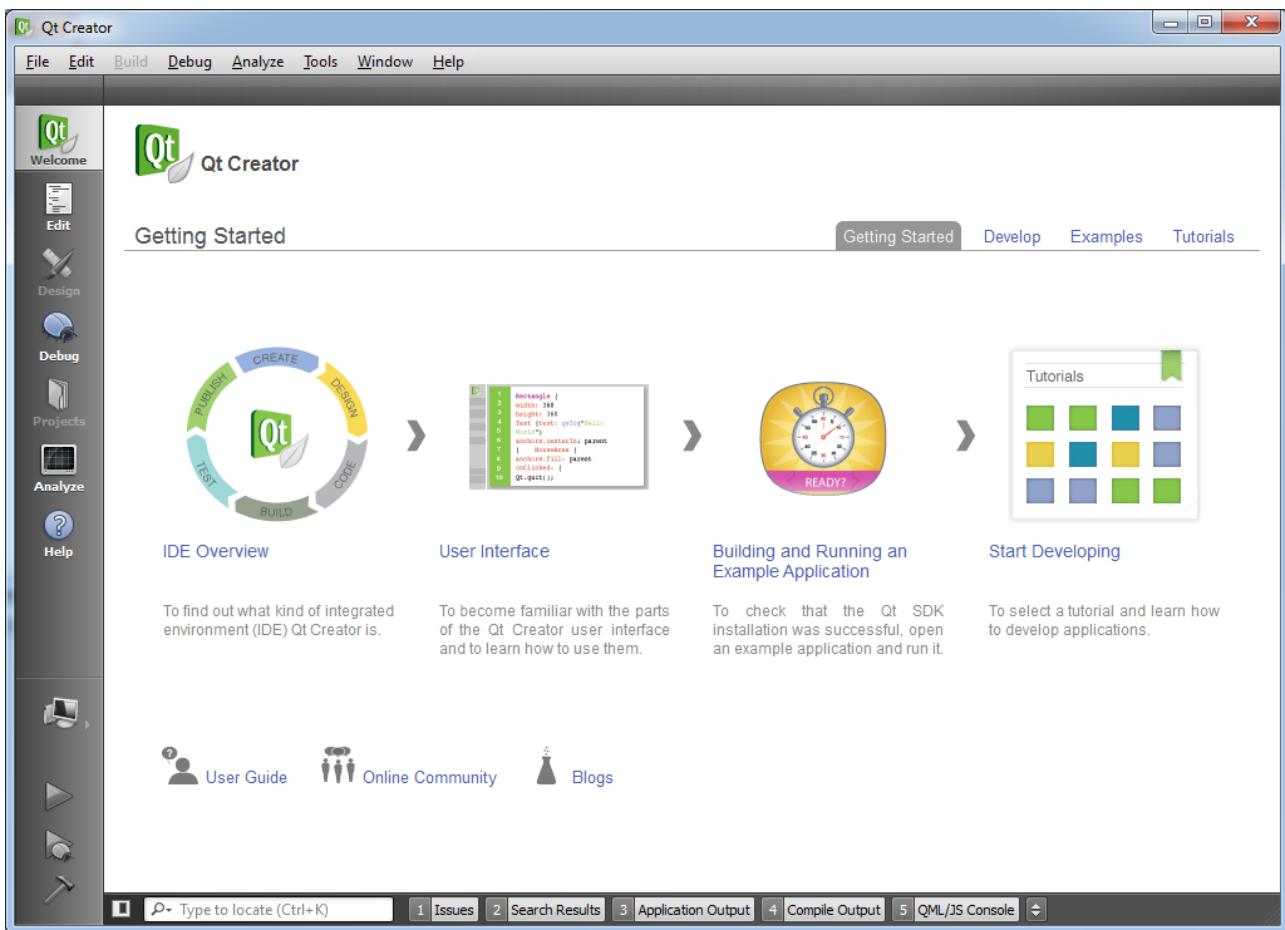


Рисунок 4.1 – Главное меню среды Qt Creator

В меню «File» находятся основные функции для работы с файлами и проектами (Создание, загрузка, сохранение).

В меню «Edit» находятся функции редактирования.

В меню «Build» находятся функции компиляции, запуска и очистки приложения (оно станет активным после создания проекта).

В меню «Debug» находятся основные функции отладки приложения.

В меню «Analyze» находятся функции профилирования приложения.

В меню «Tools» находятся настройки среды и проекта.

В меню «Window» находятся функции управления окном.

В меню «Help» находится документация фреймворка Qt и среды Qt Creator.

Боковое меню будет подробно рассмотрено далее.

Для начала работы с фреймворком необходимо создать проект.

## 4.1. Создание проекта Qt

Для создания проекта необходимо перейти в меню «File» и выбрать пункт «Create new file or project» (Ctrl+N), после этого появится окно создания проекта.

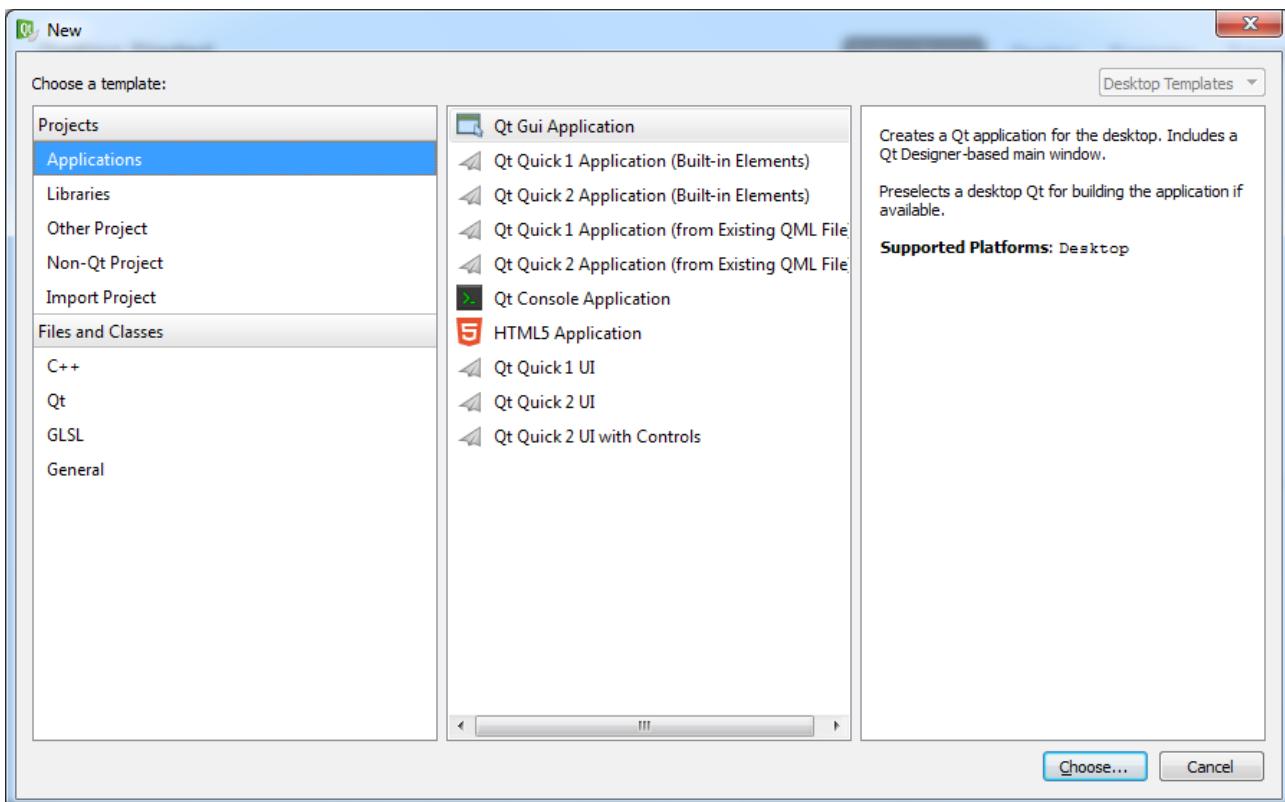


Рисунок 4.2 – Окно создания проекта

Выбираем 1 пункт (Qt Gui Application) и нажимаем Choose.

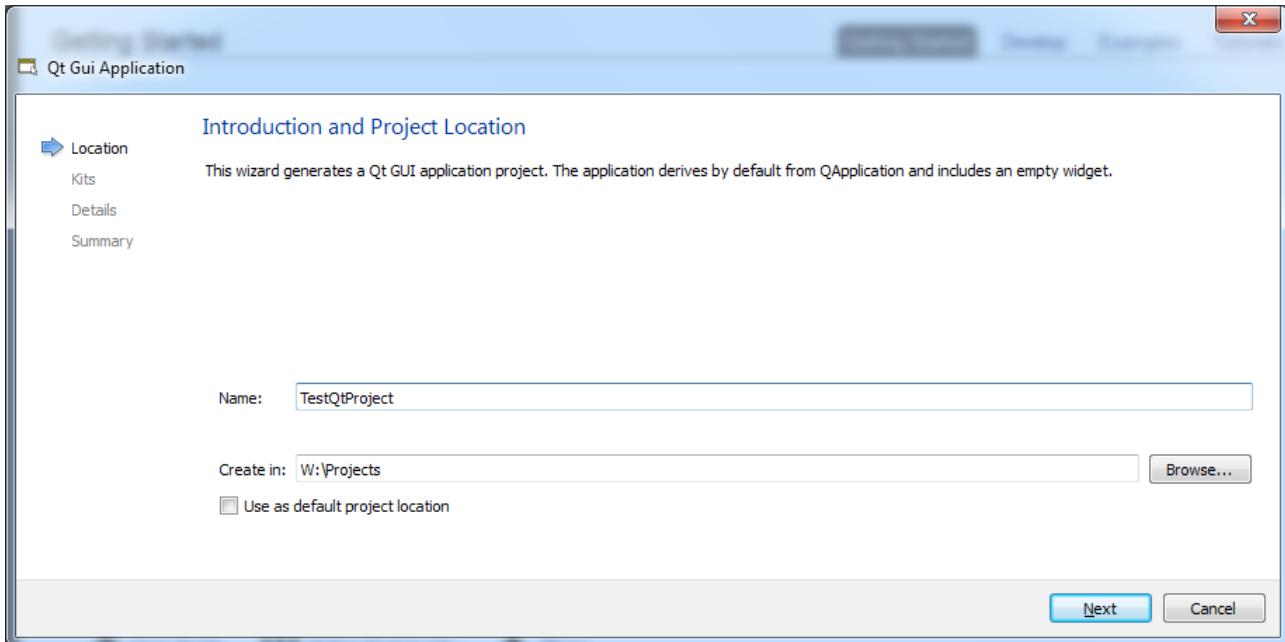


Рисунок 4.3 – Настройки проекта

Выбираем название и путь к проекту и нажимаем Next.

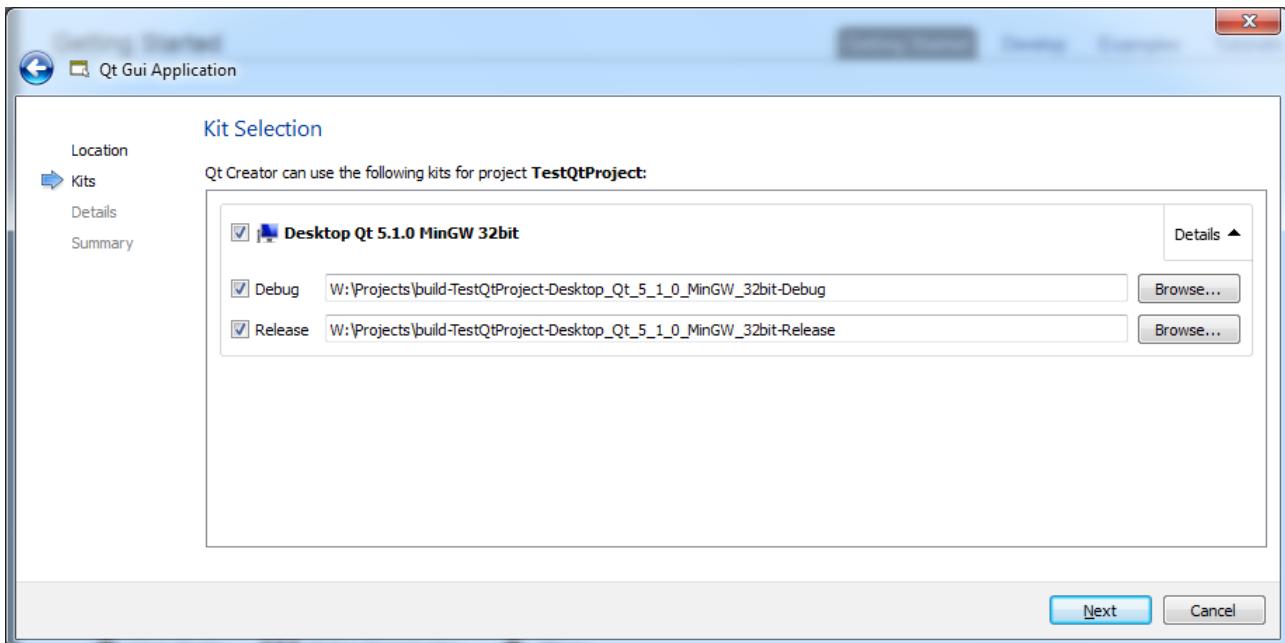


Рисунок 4.4 – Выбор параметров компиляции

Оставляем параметры компиляции по умолчанию и нажимаем Next.

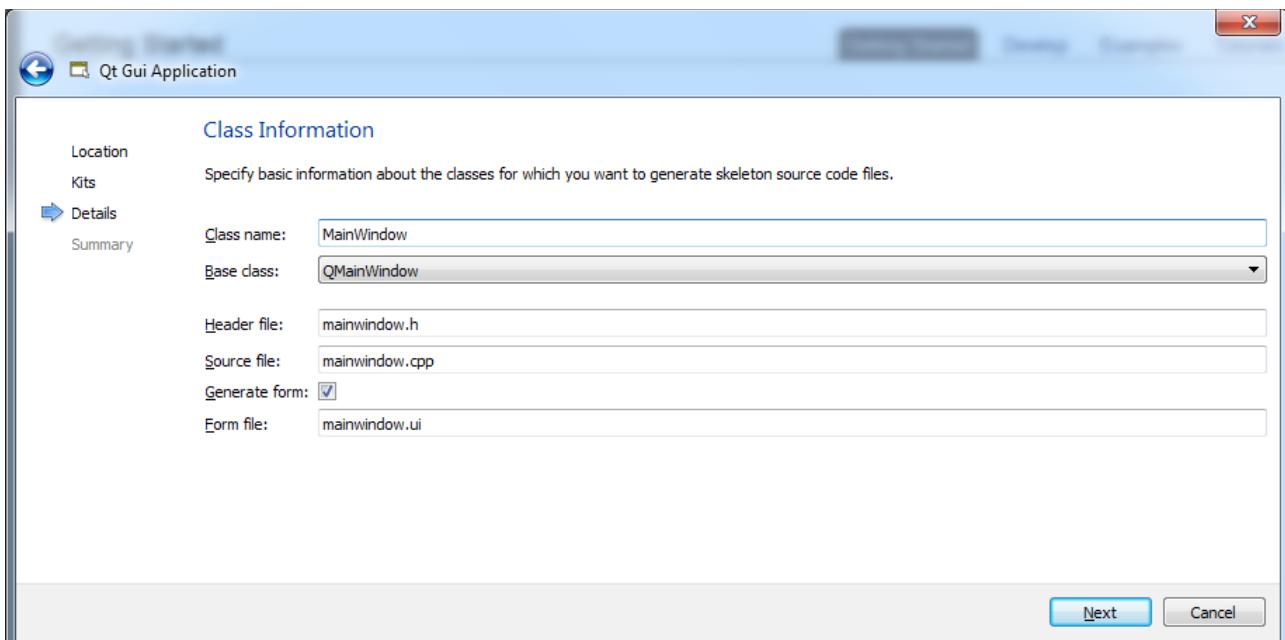


Рисунок 4.5 – Настройка параметров основного класса

В этом окне выбираются параметры класса, представляющего основное окно приложения (его имя, название заголовочного файла и файла исходного кода, а также графического файла формы). Оставляем все по умолчанию и нажимаем Next.

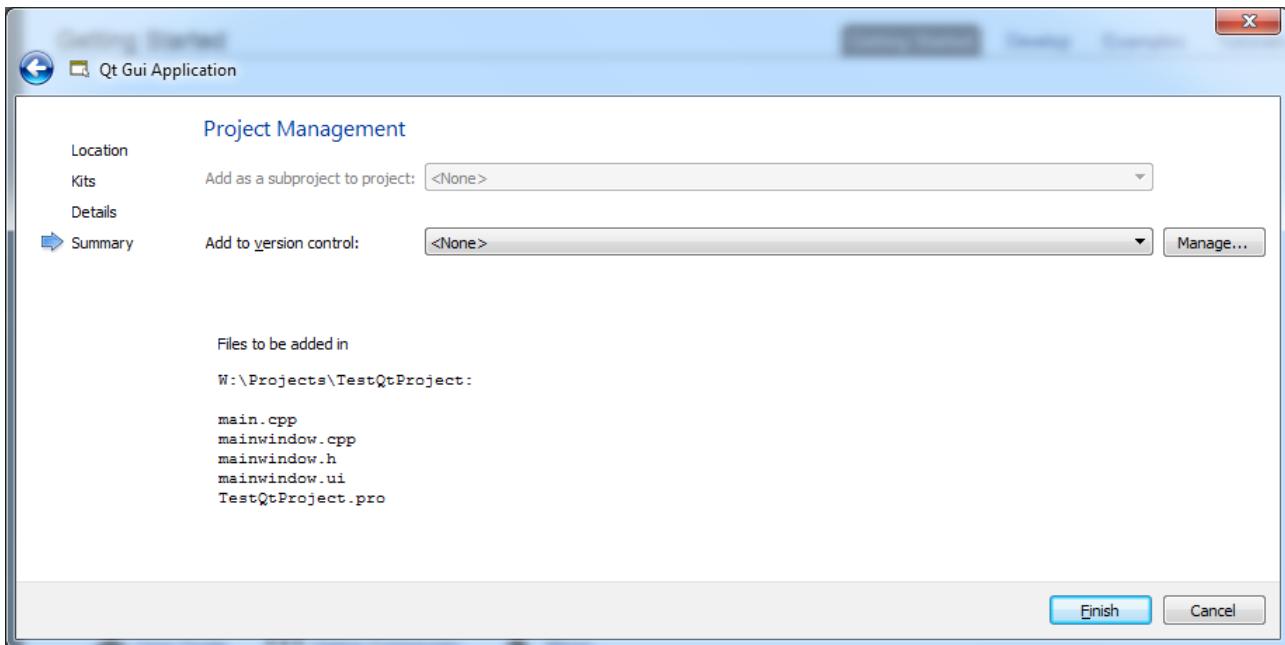


Рисунок 4.6 – Завершающее окно создания проекта

**Оставляем все по умолчанию и нажимаем «Finish».**

Проект создан и активировалась вкладка «Edit» в боковом меню.

### 3.2.2. Основы работы с проектом Qt

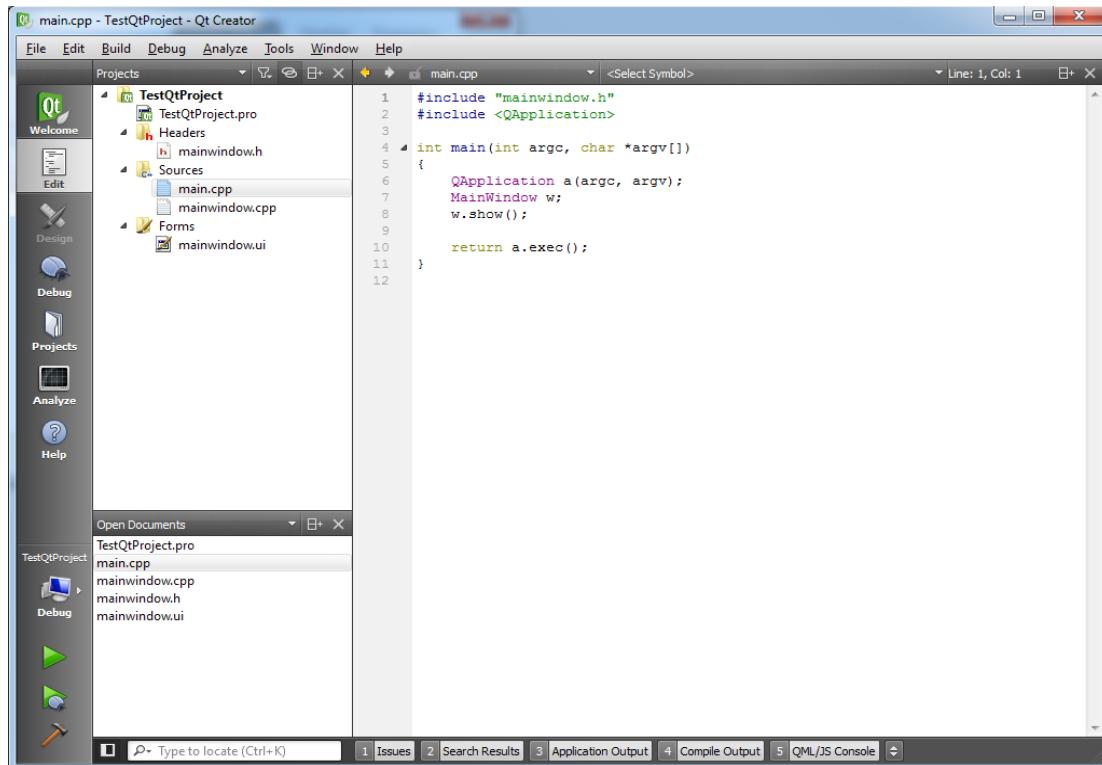


Рисунок 4.7 – Окно редактирования исходных файлов проекта

Вкладка «Edit» позволяет **редактировать исходный код проекта**. Стандартный проект содержит 5 файлов:  
- <Project name>.pro – файл проекта. Содержит основные настройки

проекта и список входящих в него файлов.

- Main.cpp содержит базовый код для запуска Qt приложения и отображения основной формы.

- Mainwindow.h – заголовочный файл класса Mainwindow (основное окно приложения).

- Mainwindow.cpp содержит исходный код этого класса.

- Mainwindow.ui позволяет создать внешний вид формы при помощи Qt Designer.

**Запустим проект.** Для этого необходимо в меню «Build» выбрать пункт «Run» (Ctrl+R) или нажать на зеленый треугольник в левой нижней части окна.

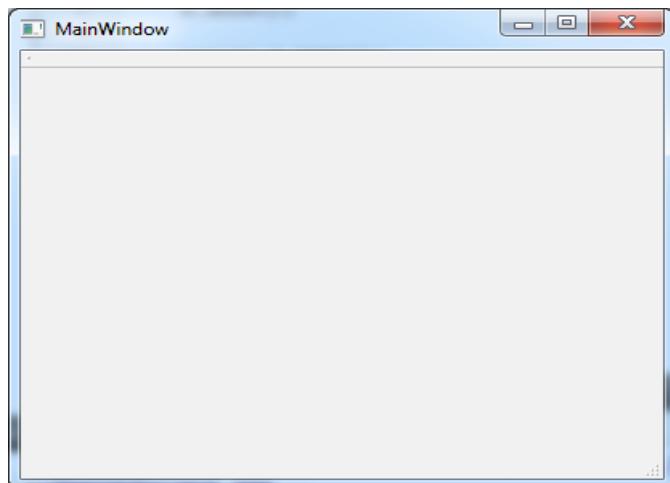


Рисунок 4.8 – Приложение после запуска

В данный момент у нас есть просто пустая форма. Для изменения внешнего вида формы и добавления элементов управления необходимо перейти на файл Mainwindow.ui, после чего в боковом меню откроется вкладка «Design». На этой вкладке находится окно приложения, слева находится список Qt виджетов, которые мы можем добавить на форму, справа вверху находится список всех уже добавленных на форму виджетов, справа внизу находится список свойств выбранного виджета. Найдем и добавим виджет «Label» на форму.

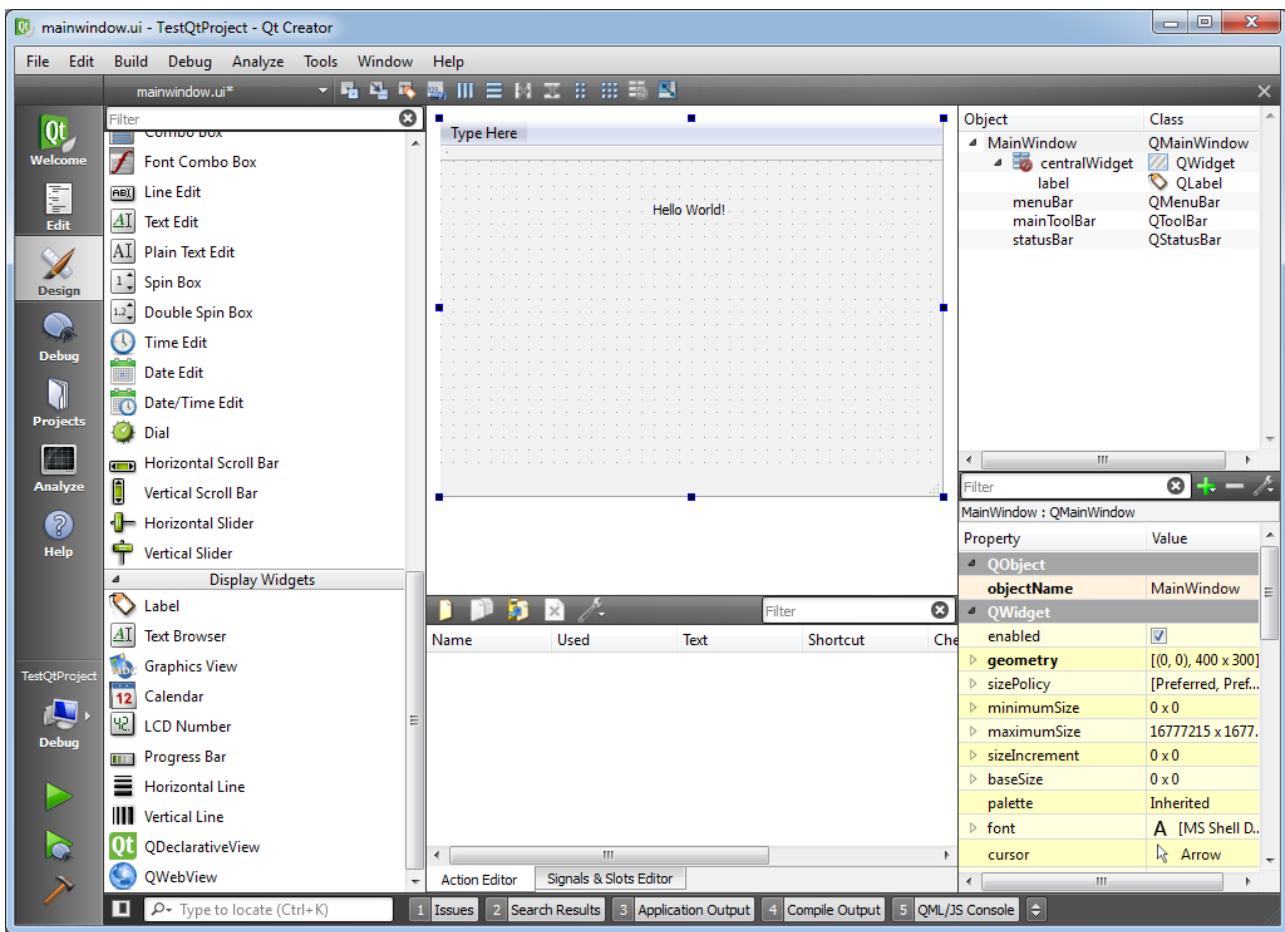


Рисунок 4.9 – Вкладка «Design»

После запуска приложения видим, что появился текст, который мы ввели для виджета Label.

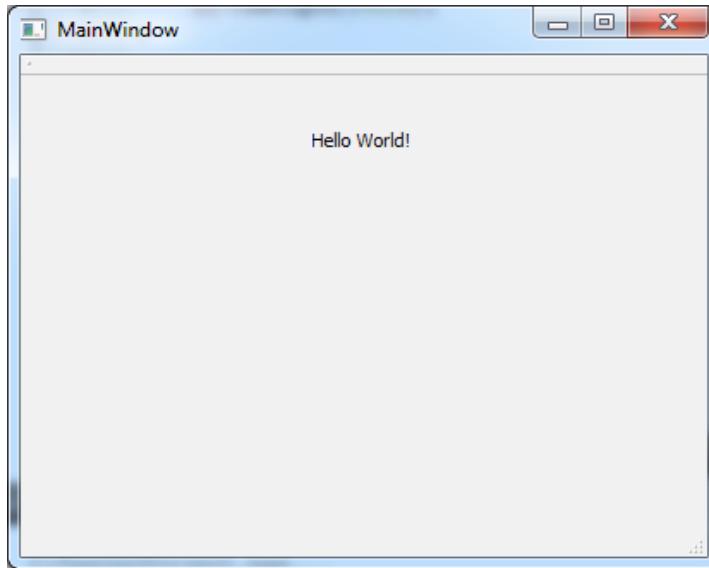


Рисунок 4.10 – Приложения после добавления виджета

Если необходимо добавить несколько виджетов на форму, то нужно позаботиться об их расположении, для этого необходимо задать схему расположения элементов на форме (Layout). Щелкаем правой кнопкой мыши по

форме на вкладке «Design», выбираем меню Layout и выбираем пункт Layout Horizontally.

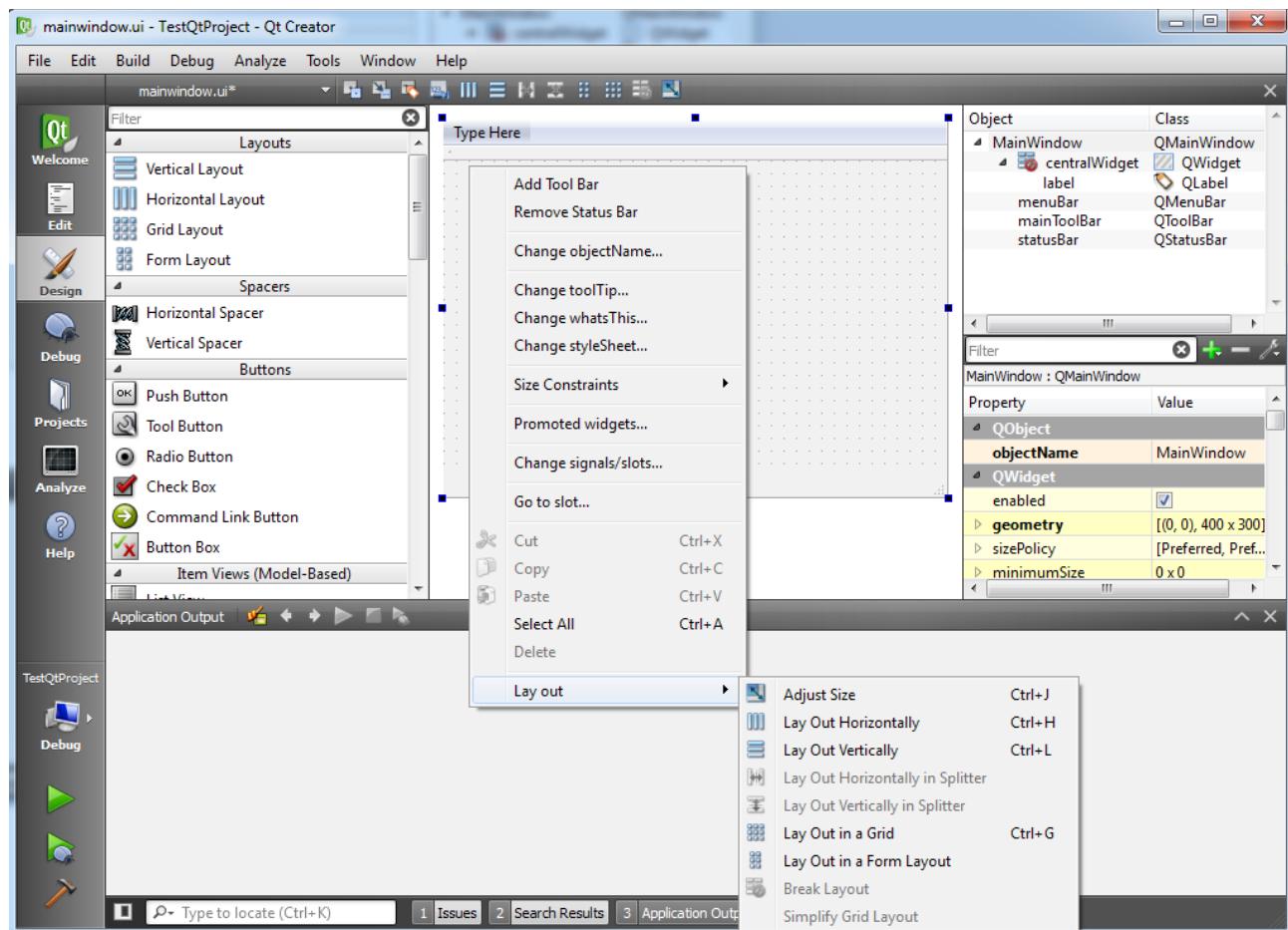


Рисунок 4.11 – Выбор схемы размещения элементов

Также можно выбрать виджет Layout из списка и делать их вложенными.

Добавим еще несколько виджетов на форму. Они будут располагаться горизонтально, согласно схеме размещения.

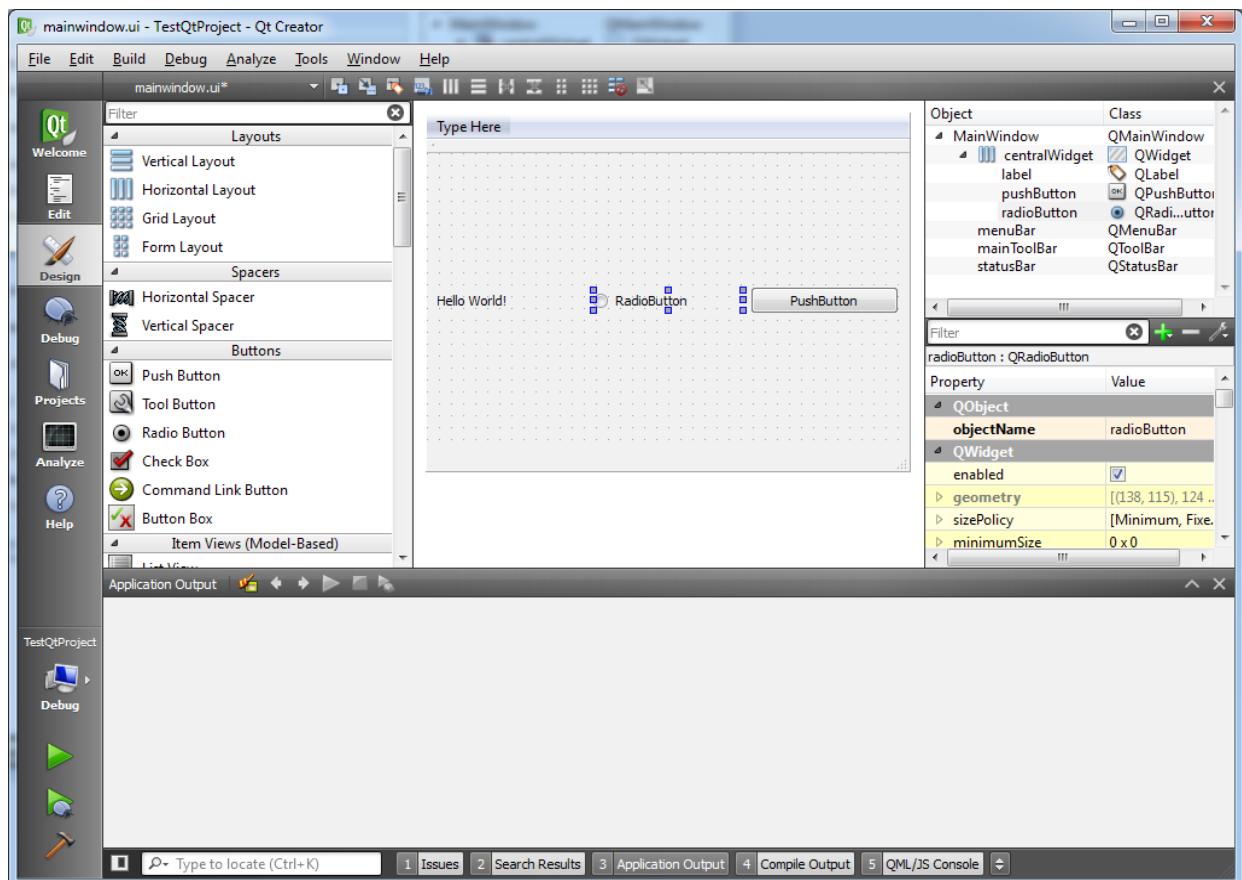


Рисунок 4.12 – Форма после добавления виджетов со схемой размещения Horizontal

Запускаем приложение и видим, что при растягивании формы происходит пропорциональное смещение виджетов, благодаря схеме размещения.

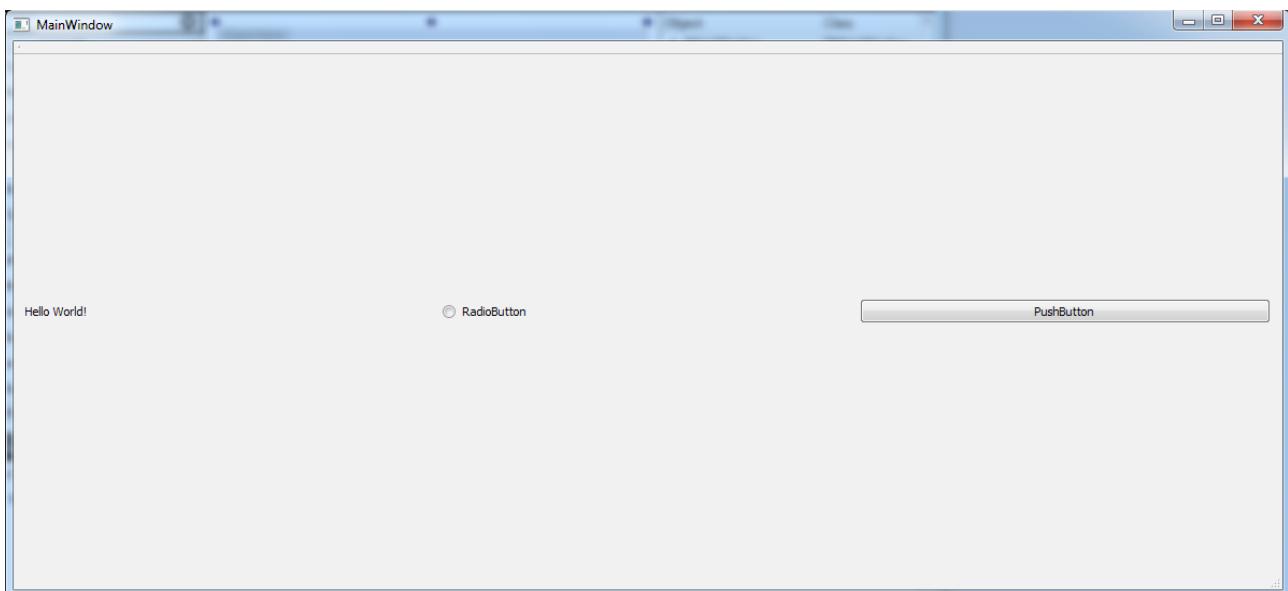


Рисунок 4.13 – Запущенное приложение со схемой размещения

После того, как форма создана, можно приступить к её программированию. Для этого необходимо получить программный доступ к виджетам. Имя виджета можно узнать в его меню свойств на вкладке «Design»

(objectName), после этого переходим во вкладку «Edit», открываем файл Mainwindow.cpp и в конструкторе после строки:

```
ui->setupUi(this);
```

получаем доступ к виджету label при помощи объекта ui и изменяем его текст:

```
ui->label->setText("Hello again");
```

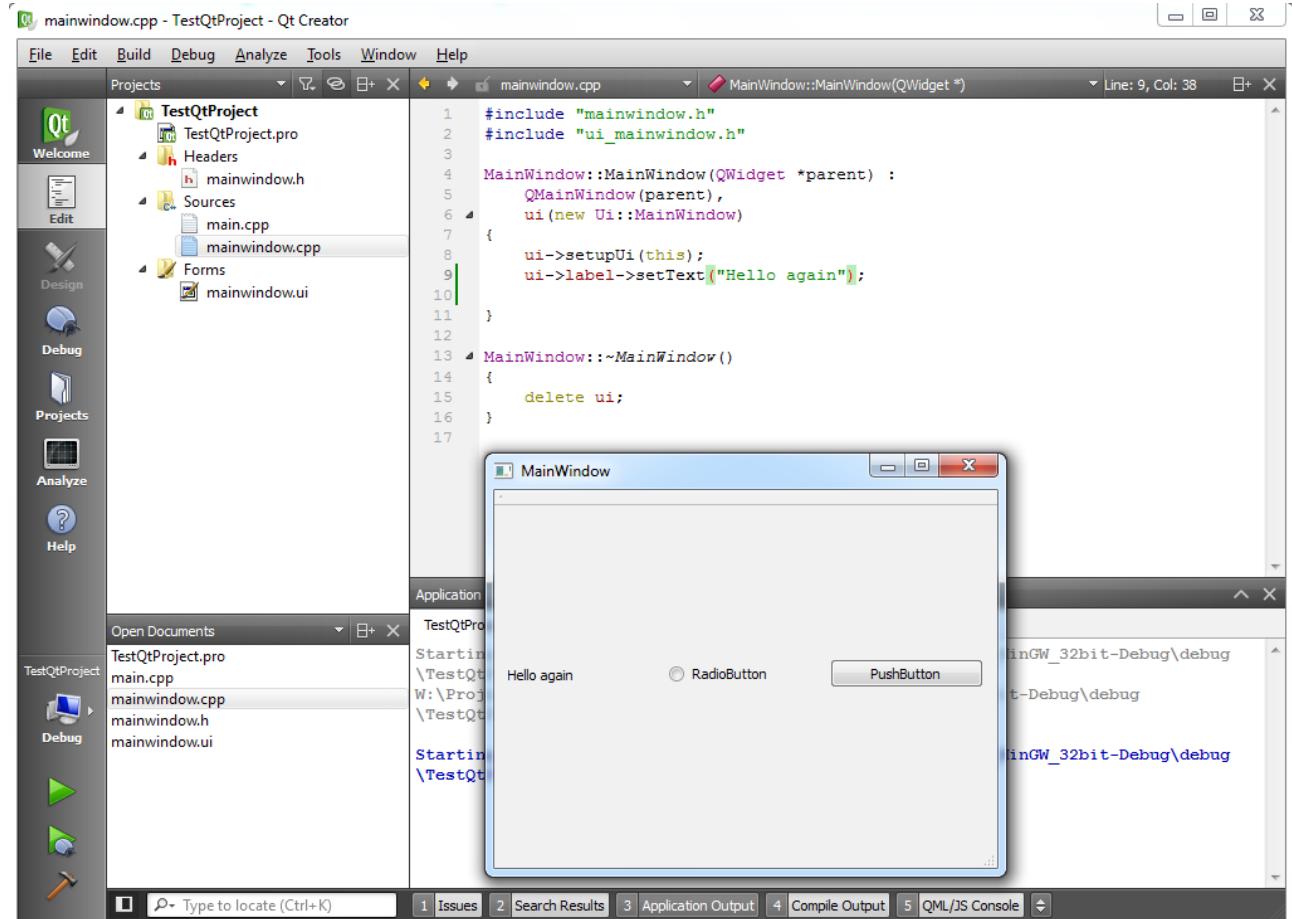


Рисунок 4.14 – Приложение после программного изменения текста виджета Label

Каждый виджет содержит множество свойств подвергаемых изменению, которые можно увидеть на вкладке «Design» или в документации в меню «Help».

## 4. ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ

4.1. Изучить основные возможности фреймворка Qt (выполняется в ходе домашней подготовки к лабораторной работе).

4.2. Установить фреймворк Qt и среду разработки Qt Creator (выполняется в ходе домашней подготовки к лабораторной работе).

4.3. Создать проект Qt Gui Application.

4.4. Выбрать вертикальную схему размещения и разместить на форме виджеты Plain Text Edit и Label

4.5. В списке свойств виджета Plain Text Edit найти свойство, отвечающее за текст, который будет отображаться там по умолчанию и ввести туда ФИО

4.6. В программном коде установить текст для виджета Label, указав группу.

4.7. Повторить этапы 4.4 — 4.6, используя горизонтальную схему размещения виджетов.

4.8. Исследовать поведение интерфейса пользователя при изменении размеров графического окна приложения, в зависимости от использованной схемы размещения виджетов.

## **5. СОДЕРЖАНИЕ ОТЧЕТА**

5.1. Цель работы.

5.2. Постановка задачи.

5.3. Текст программы.

5.4. Внешний вид окна приложения.

5.5. Выводы.

## **6. КОНТРОЛЬНЫЕ ВОПРОСЫ**

6.1. Порядок установки фреймворка Qt.

6.2. Порядок создания проекта Qi GUI Application.

6.3. Структура проекта Qt, назначение основных файлов.

6.4. Что понимается под виджетом?

6.5. Каким образом упорядочивается размещение виджетов на форме Qt-приложения?

6.6. Какие существуют схемы размещения виджетов в Qt?

6.7. Приведите пример программного изменения свойств виджета.

6.8. Для чего предназначен фреймворк Qt?

6.9. Для каких платформ возможна компиляция Qt-проектов?

6.10. Перечислите основные модули Qt и их назначение.

6.11. Какие языки программирования могут быть использованы для создания Qt-приложений?

## **БИБЛИОГРАФИЧЕСКИЙ СПИСОК**

1. Ж. Бланшет. Qt 3: программирование GUI на C++ / Бланшет Ж., Саммерфилд М. — М. : КУДИЦ — ОБРАЗ, 2005. - 448 с.

2. Е.Р. Алексеев. Программирование на языке C++ в среде Qt Creator /Е. Р. Алексеев, Г. Г. Злобин, Д. А. Костюк, О. В. Чеснокова, А. С. Чмыхало.— М.:Альт Линукс, 2015. — 448 с.

3. Буч, Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++/Г. Буч. — М. : БИНОМ ; СПб. : Невский диалект, 2001. - 560 с.

4. Шилдт, Г. C++: базовый курс, 3-е издание /Г. Шилдт. — М.: «Вильямс», 2012. — 624 с.

5. Шилдт, Г. Полный справочник по C++, 4-е издание /Г. Шилдт. — М.: «Вильямс», 2011. — 800 с.

## **ПРИЛОЖЕНИЕ — Порядок установки среды разработки Qt Creator**

Последнюю версию среды разработки Qt Creator можно бесплатно скачать по адресу <http://qt-project.org/downloads> (Qt 5.1.0 for Windows 32(64)-bit (MinGW 4.8, OpenGL) ).

Рассмотрим подробно процесс установки Qt Creator. После запуска установочного файла, появится начальное окно установки.

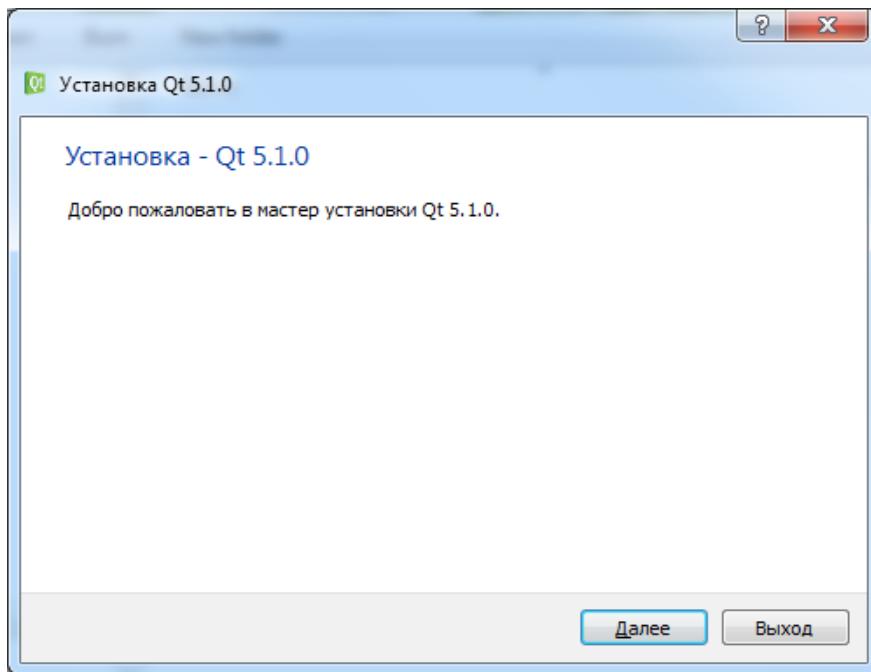


Рисунок 1 – Начальное окно установки Qt

Нажимаем “Далее” для продолжения установки.

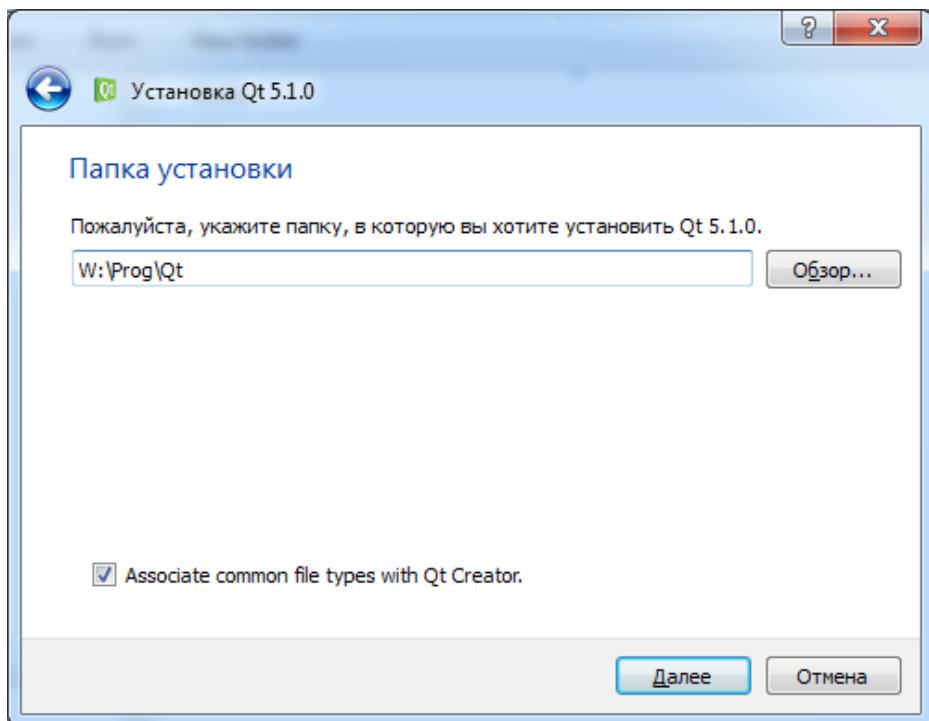


Рисунок 2 – Выбор пути установки

Выбираем путь установки и нажимаем “Далее”.

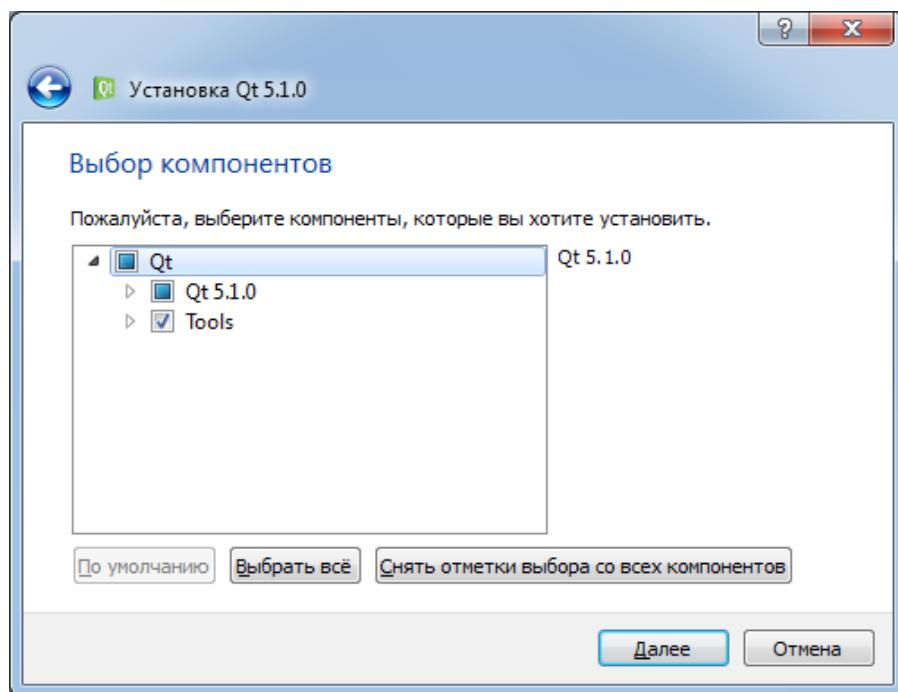


Рисунок 3 – Выбор компонентов установки

Оставляем все по умолчанию и нажимаем “Далее”.

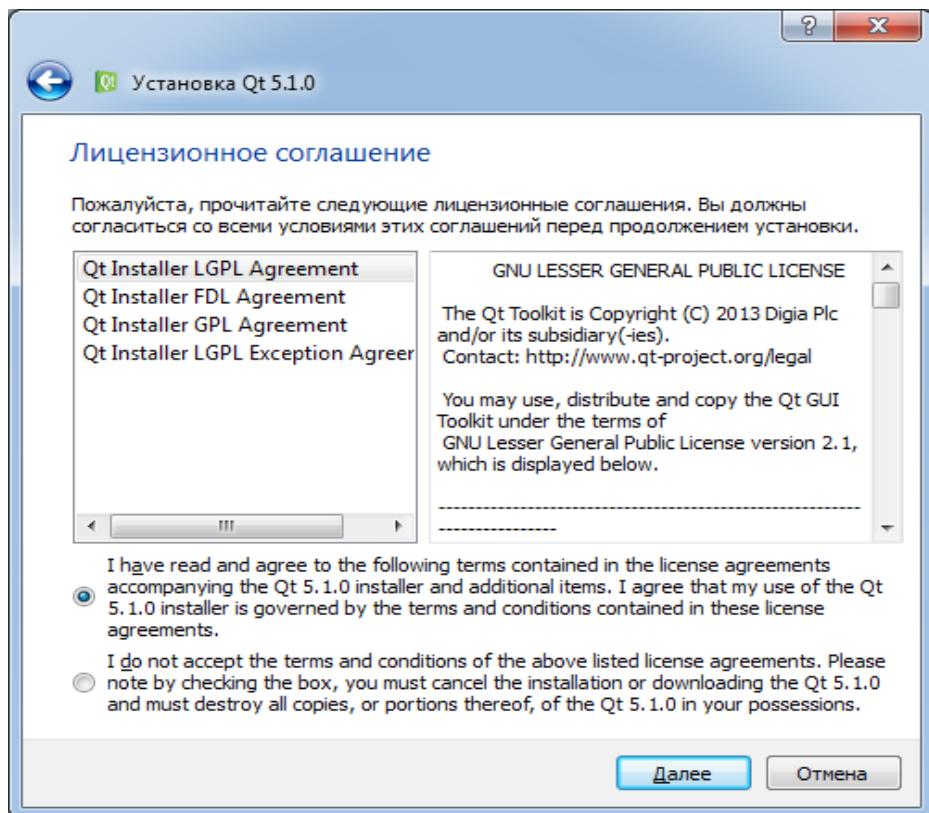


Рисунок 4 – Лицензионное соглашение

Соглашаемся с лицензионным соглашением и жмем далее.

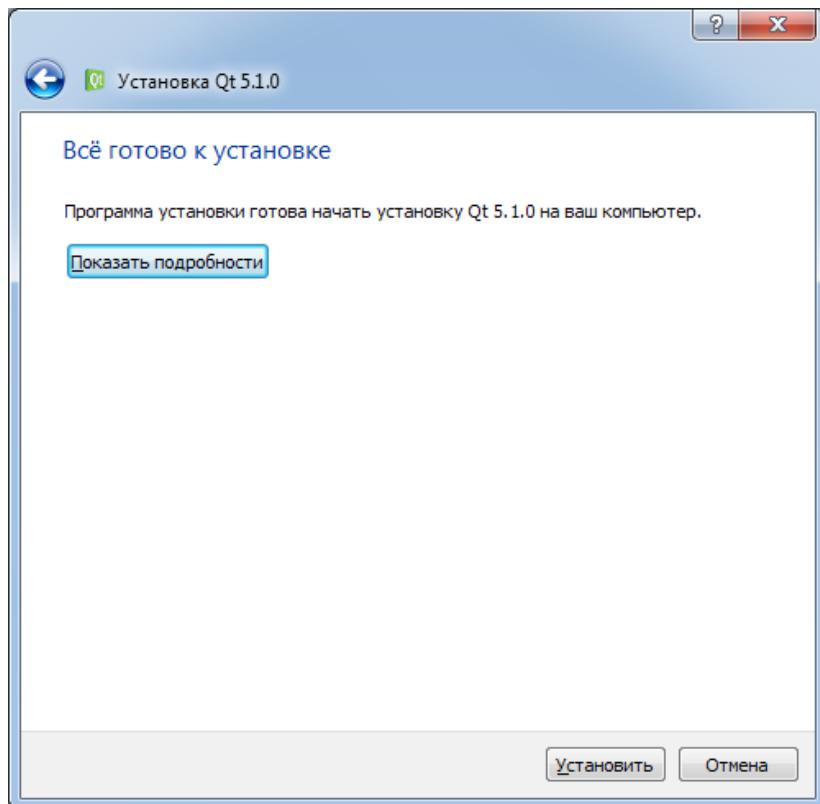


Рисунок 5 – Подтверждение установки

Нажимаем “Установить” и начинается процесс установки.

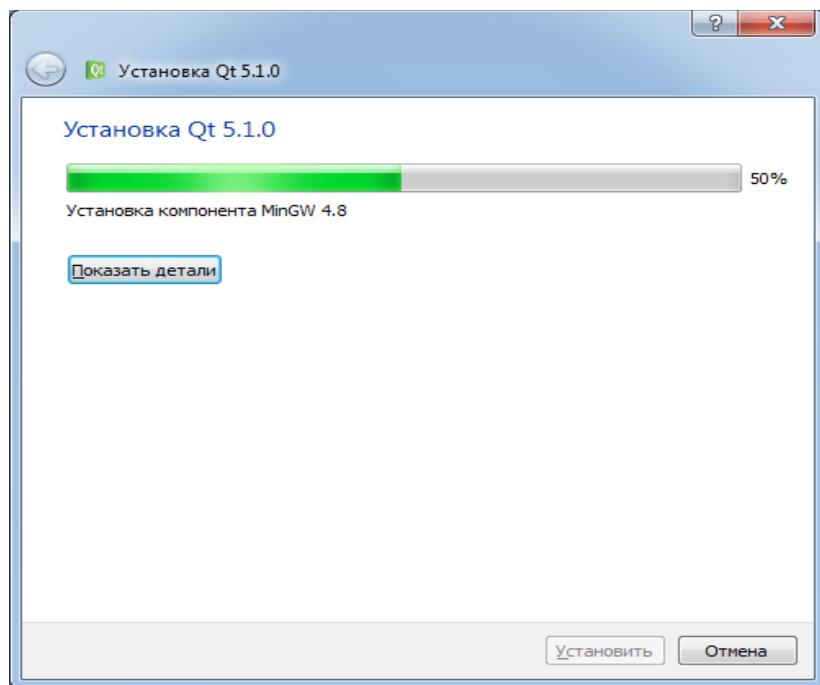


Рисунок 6 – Процесс установки

После того, как установка закончится, появляется окно завершения.

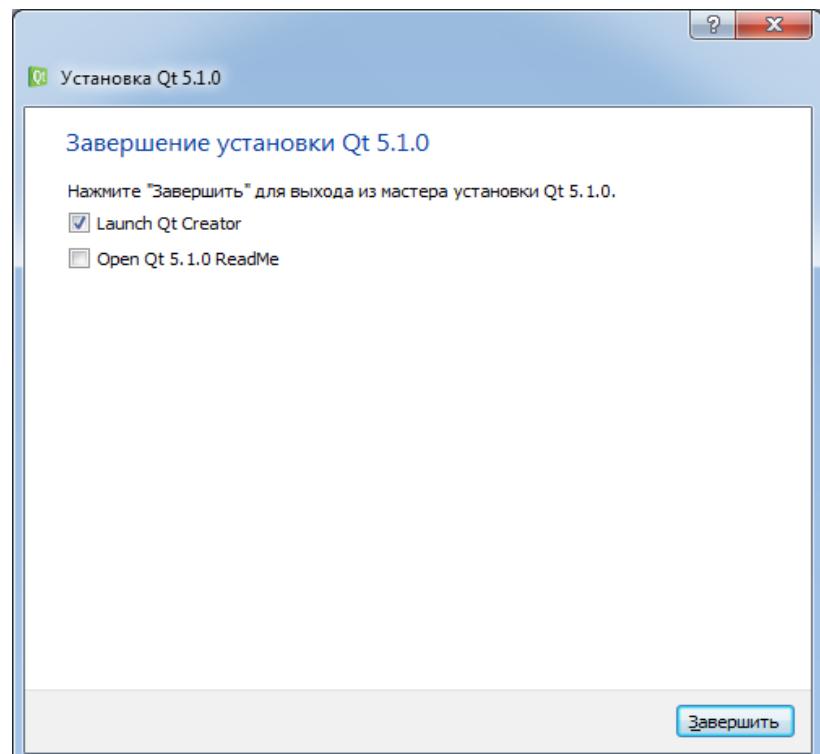


Рисунок 7 – Завершение установки

Нажимаем “Завершить” и запускаем Qt Creator.







Заказ № \_\_\_\_\_ от «\_\_\_\_\_» \_\_\_\_\_ 2018 г. Тираж \_\_\_\_\_ экз.