Практическая работа №1 – Знакомство с Qt

Цель работы

Знакомство со средой разработки Qt Creator, создание простого калькулятора.

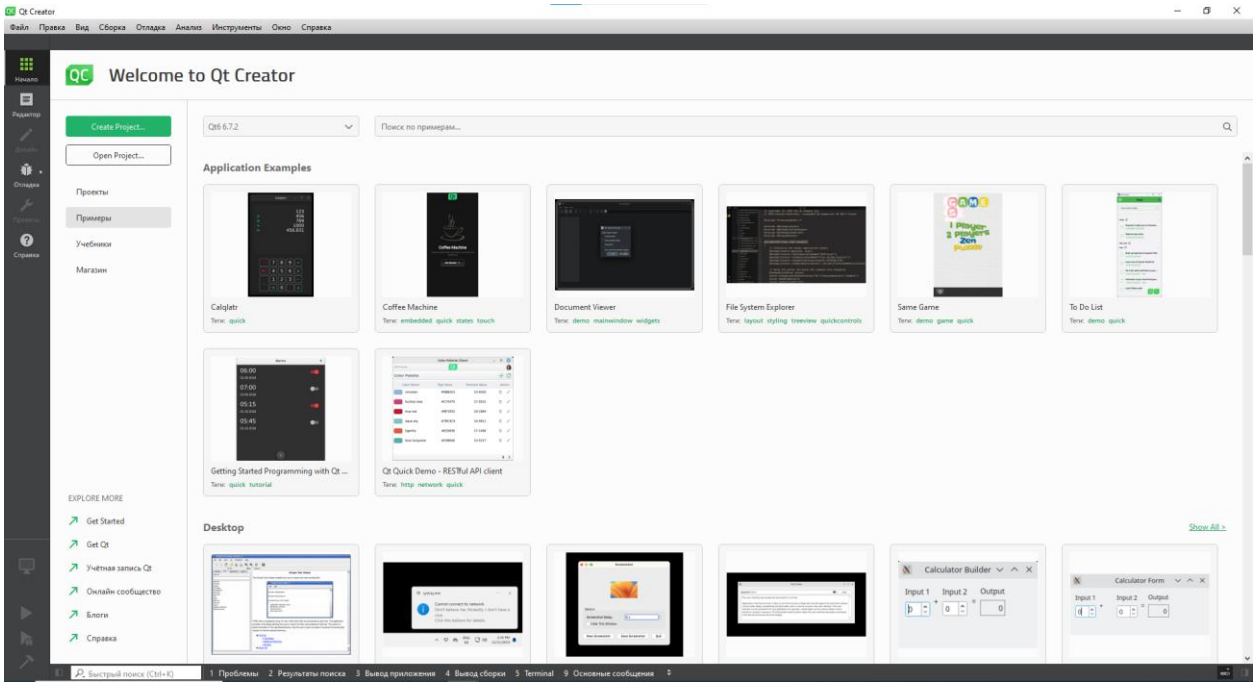
Постановка задачи

1. Установить среду разработки Qt Creator;
2. Создать новый проект;
3. Разработать интерфейс программы калькулятор со следующими операциями:
    1. Сложение;
    2. Вычитание;
    3. Умножение;
    4. Деление;
4. Реализовать логику работы программы.

Теоретическое введение

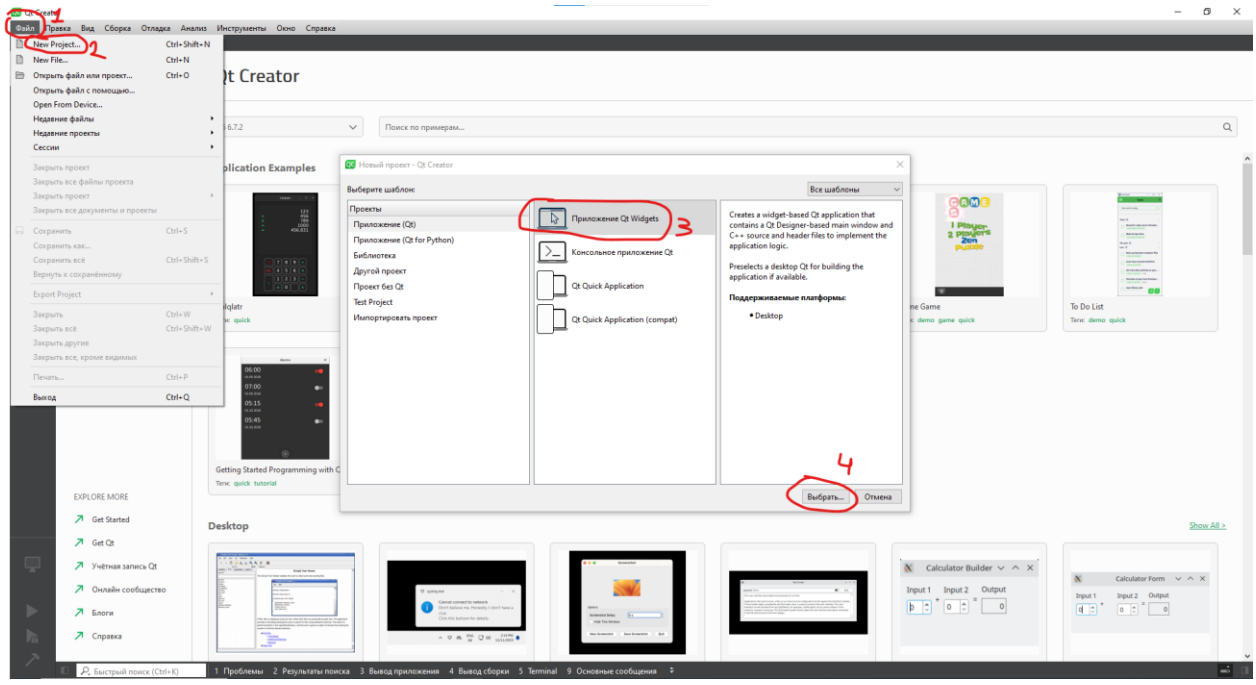
Создание нового проекта

При запуске среды разработки Qt Creator, пользователя встречает главное окно программы:



Главное окно среды разработки Qt Creator

Для создания нового проекта необходимо нажать Файл -> Создать файл или проект -> Приложение (Qt) -> Приложение Qt Widgets. После этого нажать на кнопку Choose:

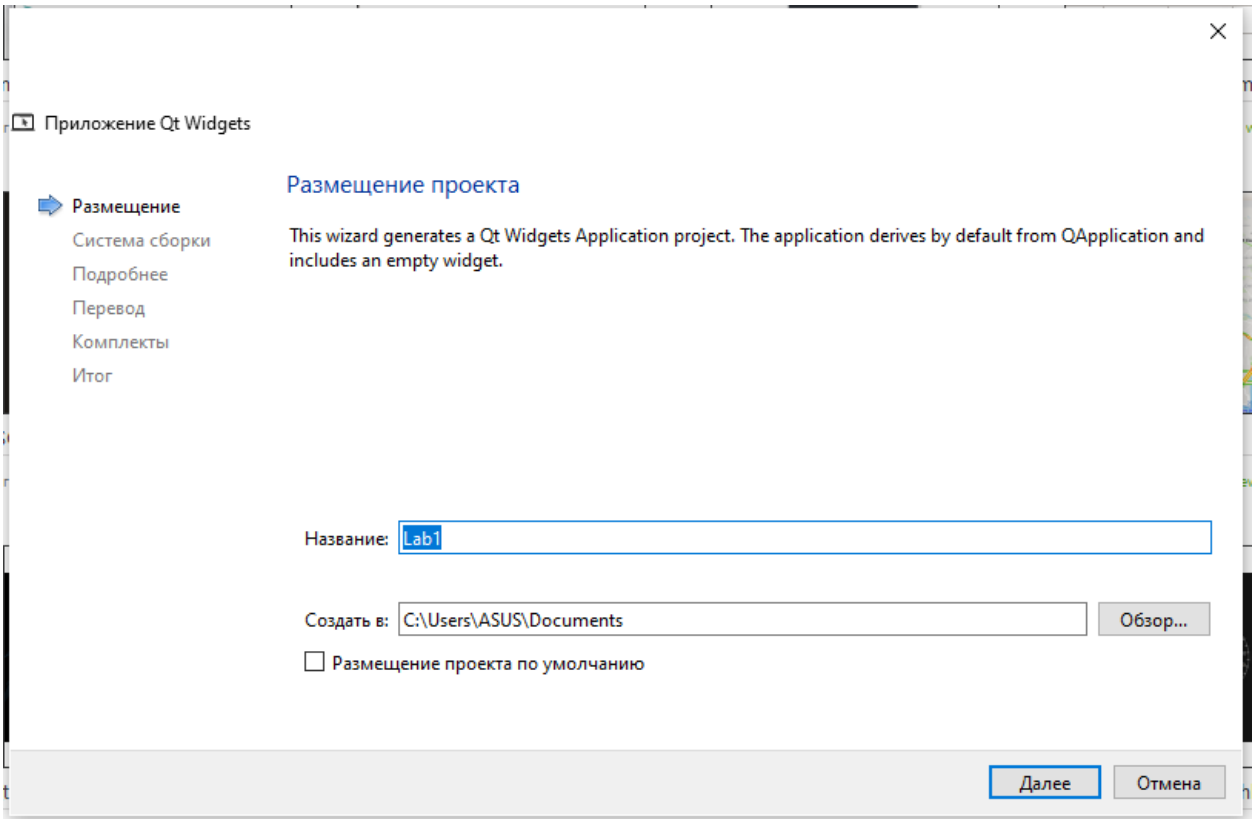


Создание проекта

Далее в открывшемся окне необходимо ввести название проекта.

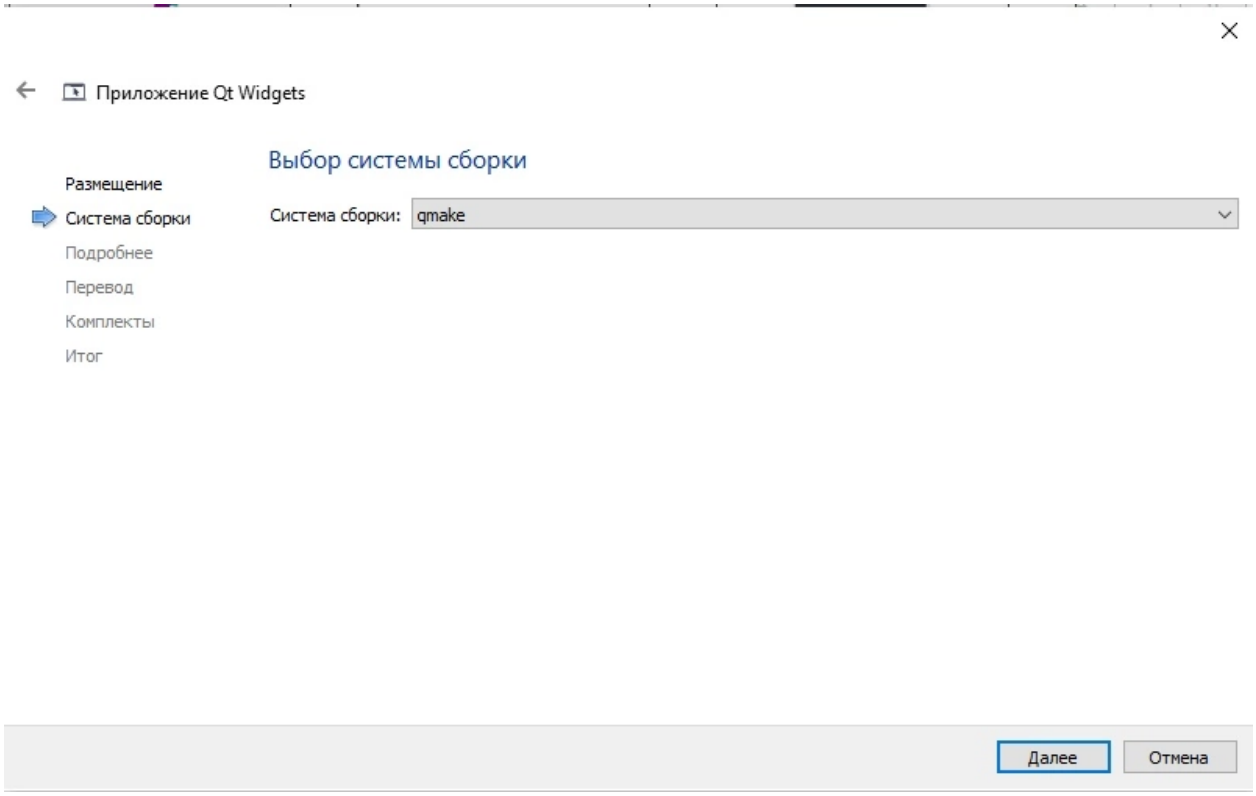
Обратите внимание на расположение папки с проектом:

Путь должен содержать только символы на латинице и цифры (если имеются), а также не содержать пробелов, в противном случае программа не будет компилироваться!

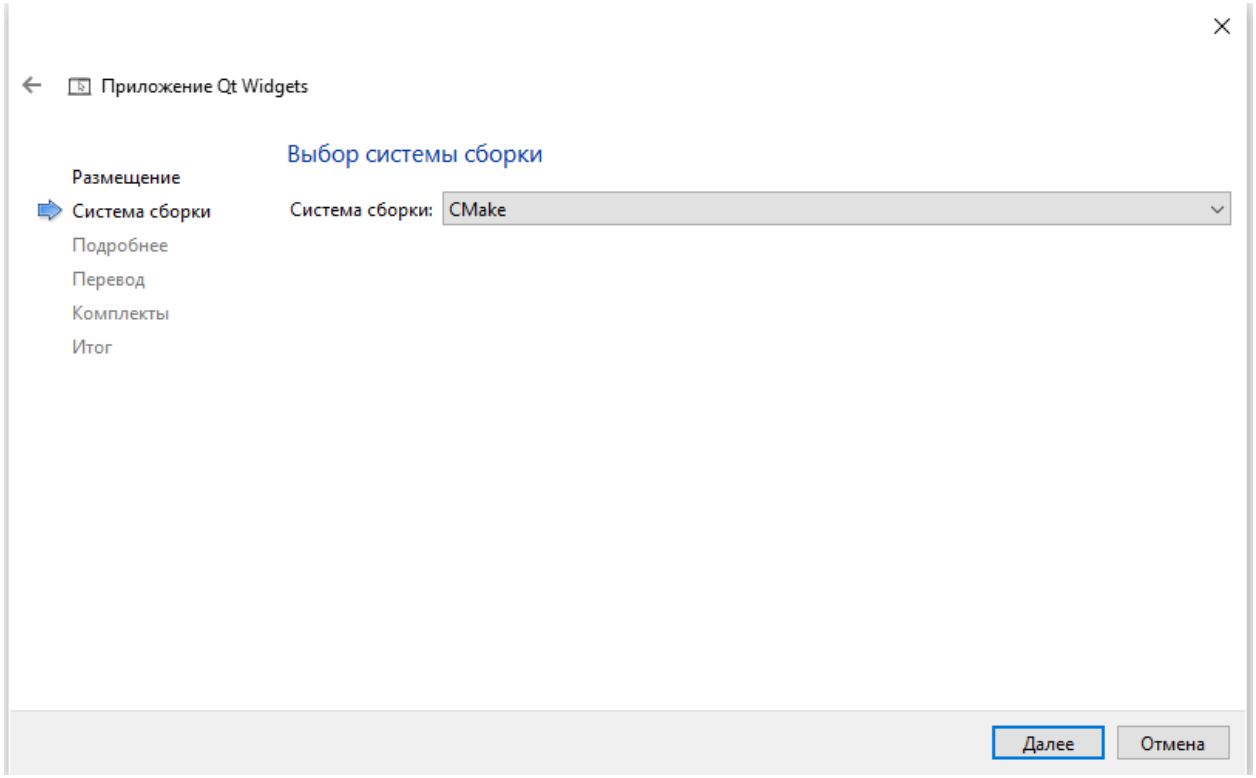


На следующем этапе необходимо выбрать систему сборки. Рекомендуется использовать ту систему сборки, которая предложена по умолчанию. Для версии Qt 5.x.x рекомендуется использовать систему сборки qmake, а для версии Qt 6.x.x — систему CMake:

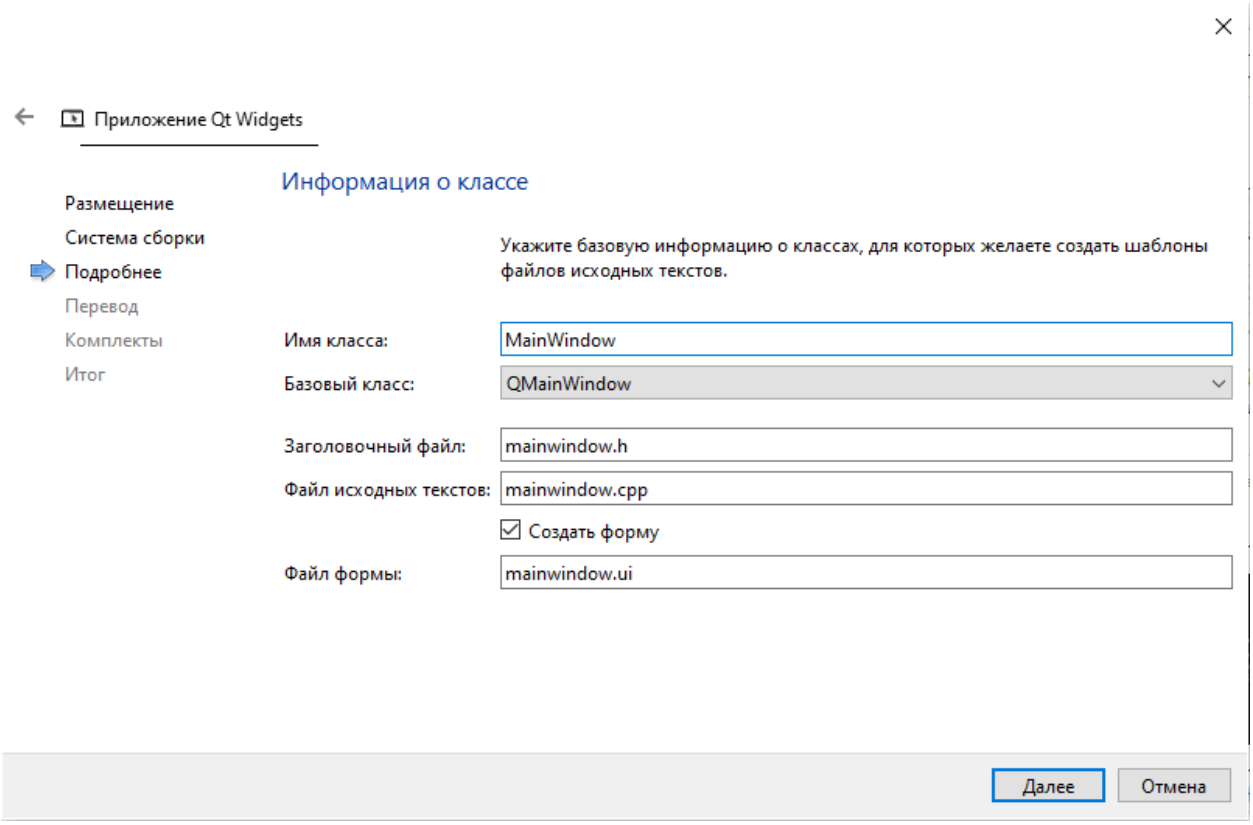
Для версии Qt 5.x.x:

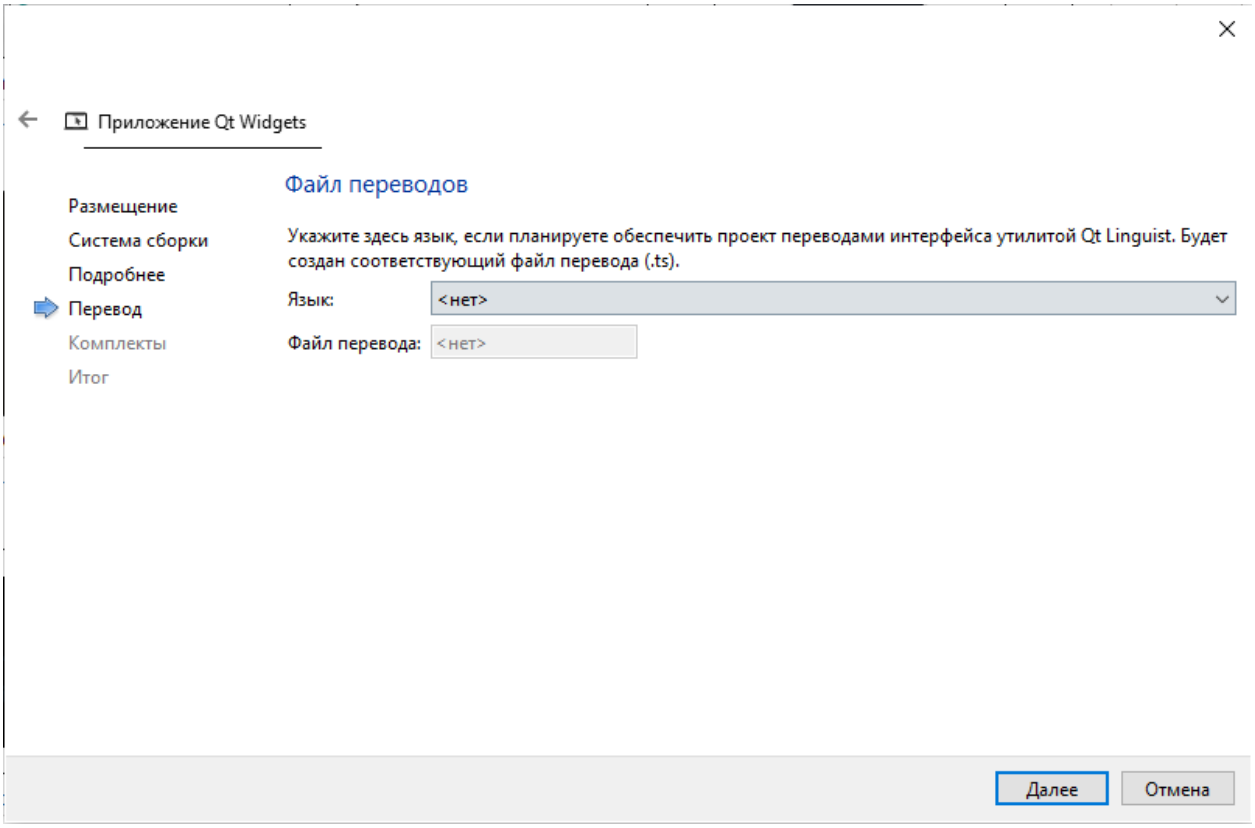


Для версии Qt 6.x.x:

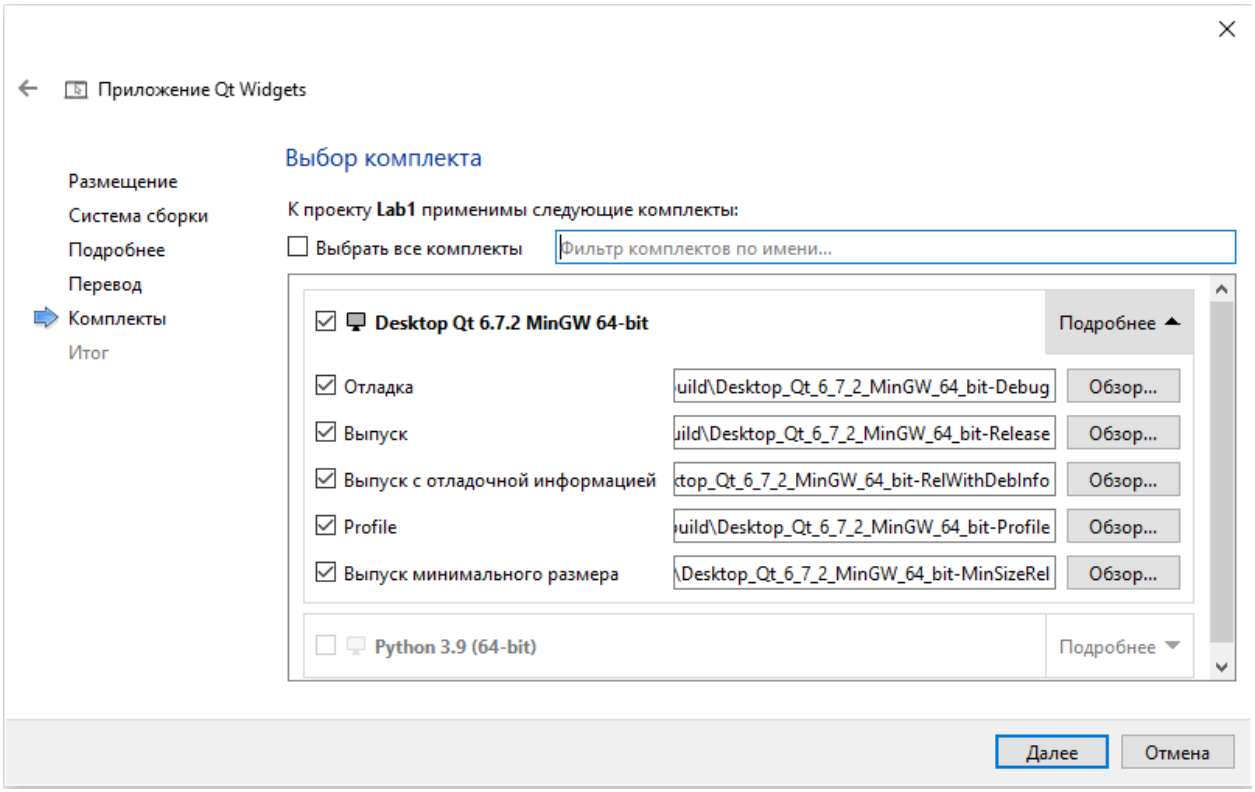


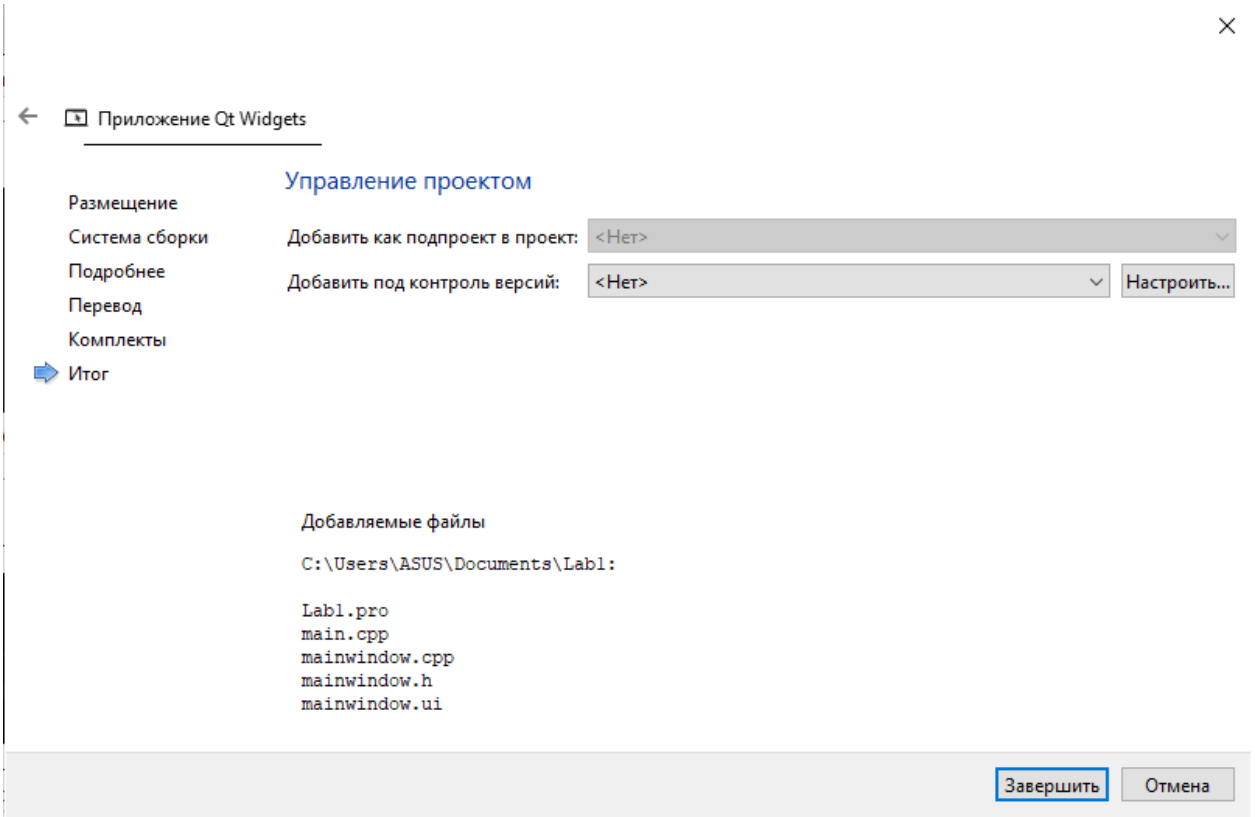
Далее везде необходимо нажать «Далее» и «Завершить». Для текущих задач изменение следующих параметров не нужно:





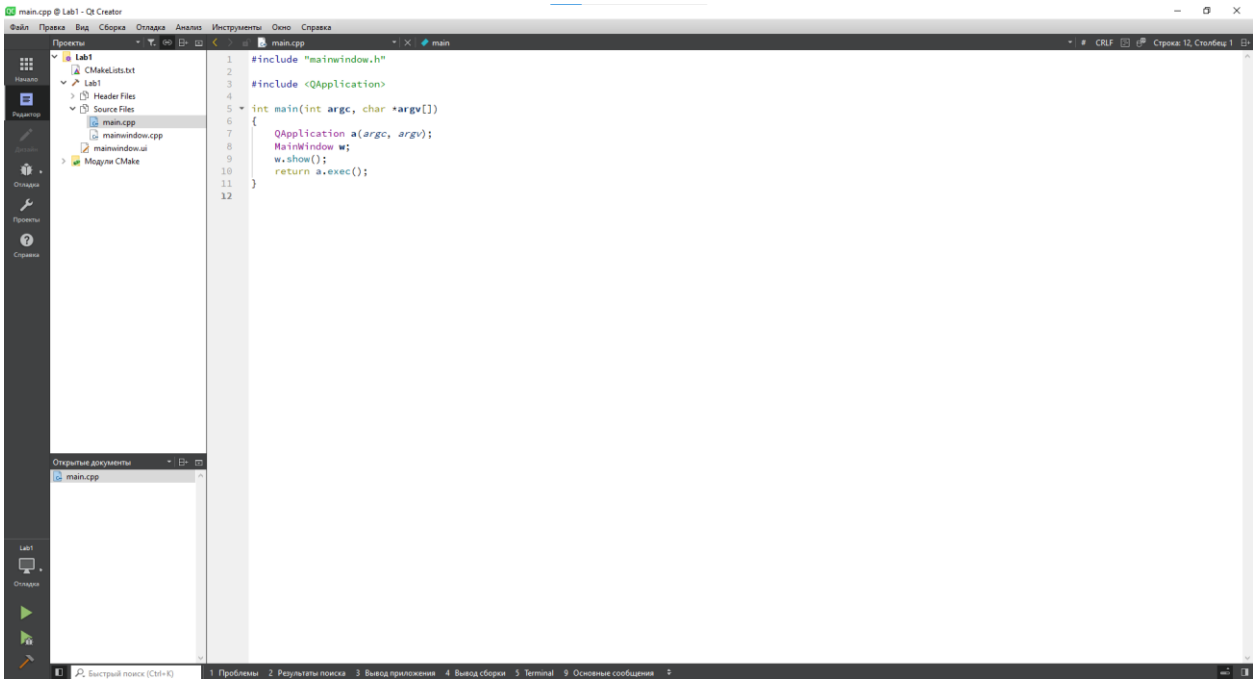
В качестве компилятора выбираем `MinGW 64-bit` (любой установленной версии):





Обзор проекта

После создания проекта открывается файл `main.cpp`. Любая программа в языке C++ начинается с запуска функции `main()`.



Помимо main.cpp, автоматически были созданы следующие файлы:

- CMakeLists.txt
- mainwindow.h
- mainwindow.cpp
- mainwindow.ui

CMakeLists.txt – файл, содержащий конфигурацию сборки проекта, с его помощью добавляются библиотеки в проект, при необходимости.

Следующие 3 файла имеют одинаковое название – mainwindow, но имеют разные расширения - *.h, *.cpp и *.ui.

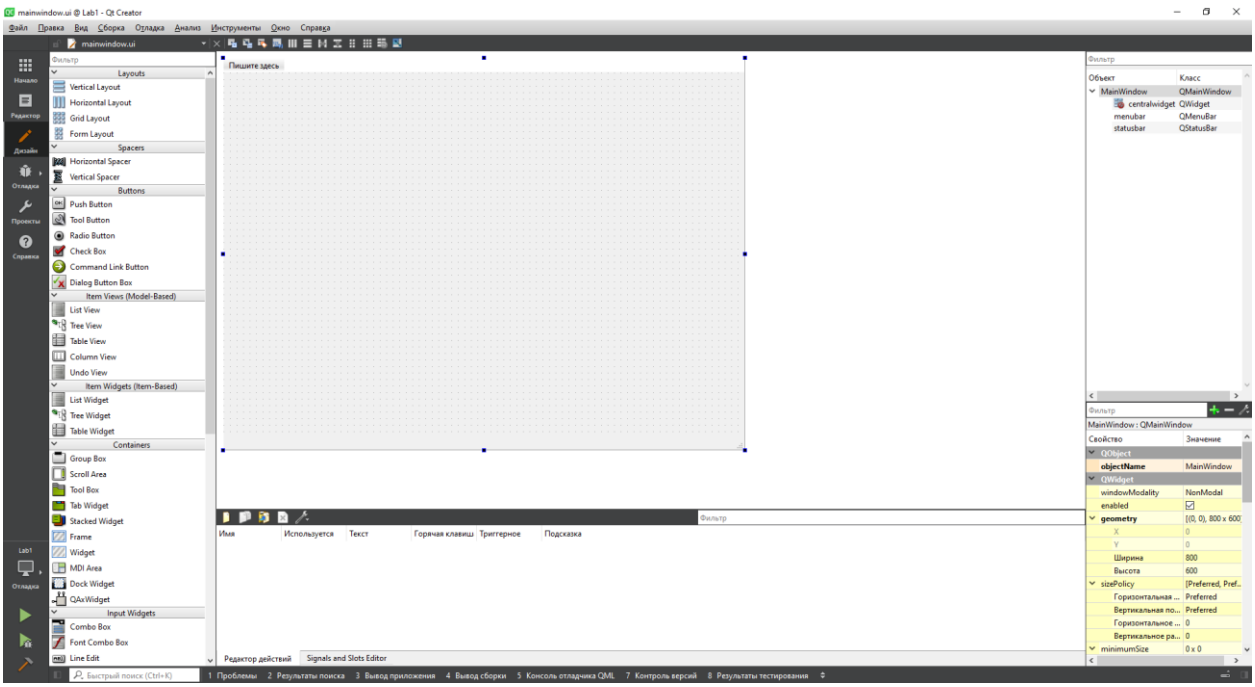
В файле mainwindow.ui находится графический интерфейс программы. Расширение *.ui является сокращением от User Interface – ui. В нём хранится вся графическая часть приложения.

В файле mainwindow.cpp находится основной код программы. Файлы *.cpp называются файлы исходных кодов. Тут будет располагаться код обработки нажатий на кнопки, ввод текста пользователем и другие варианты взаимодействия пользователя с программой.

В файле mainwindow.h находятся глобальные переменные, а также объявления функций и классов. Файлы *.h называются заголовочными файлами.

Создание графического интерфейса.

Для создания графического интерфейса необходимо перейти в файл mainwindow.ui.



Графический интерфейс программы.

В открывшемся окне в левой части располагаются элементы графического интерфейса, которые можно добавить в программу – они называются виджеты (Widgets).

Изначальное окно программы, называемое формой (Form), полностью пустое. Для добавления элемента на форму необходимо нажать левой

клавишей мышки на необходимый элемент в левом боковом меню, и, не отпуская её перенести элемент на форму.

Основные виджеты, которые будут применяться в работе – Label, LineEdit и PushButton.

Label – самый простой виджет. Он может только выводить текст на экран.

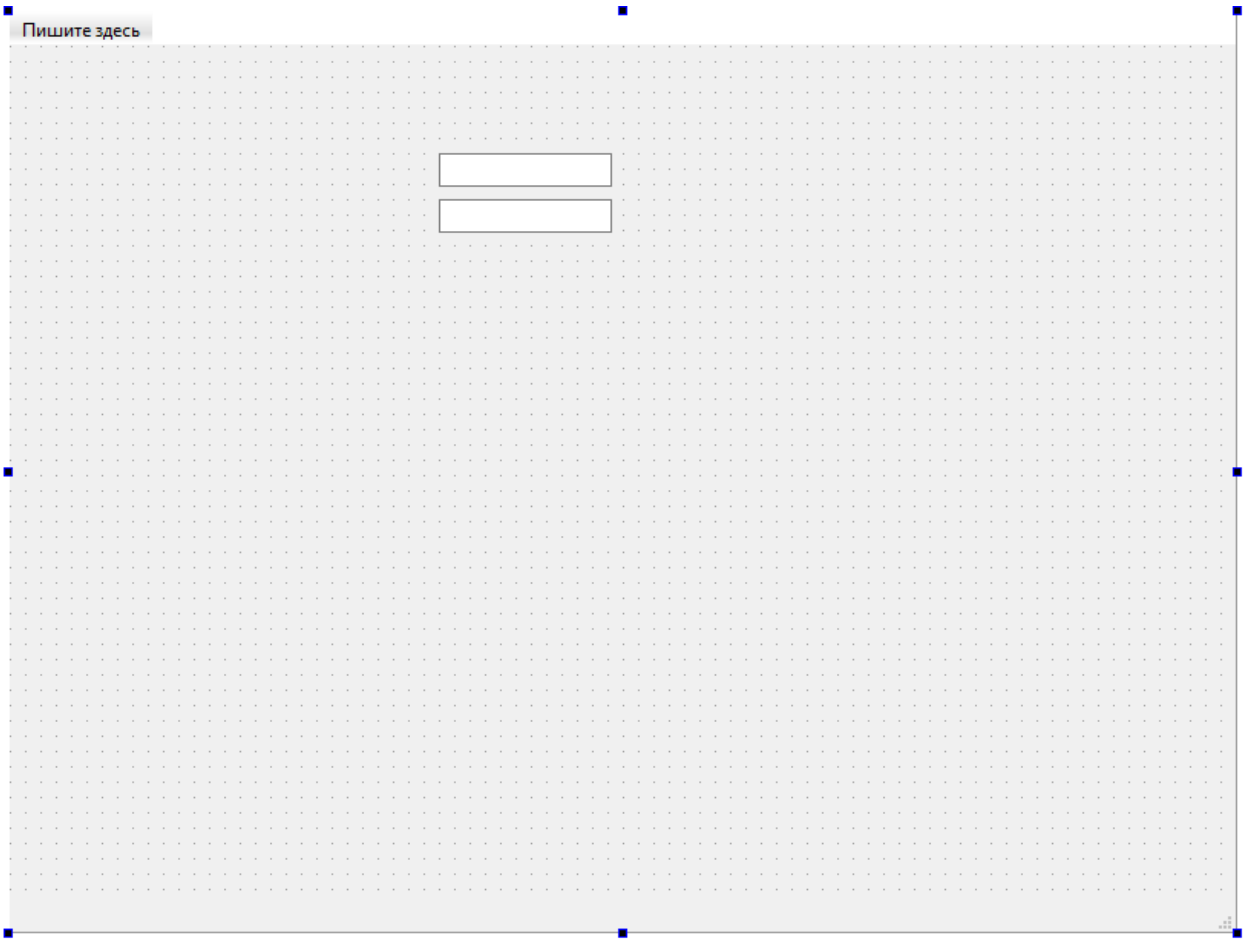
LineEdit – простая строка ввода. В него пользователь может вводить любые символы с клавиатуры.

PushButton – простая кнопка, по нажатии на которую, программа будет выполнять определённое действие.

Для выполнения поставленной задачи создадим 2 поля ввода (LineEdit):

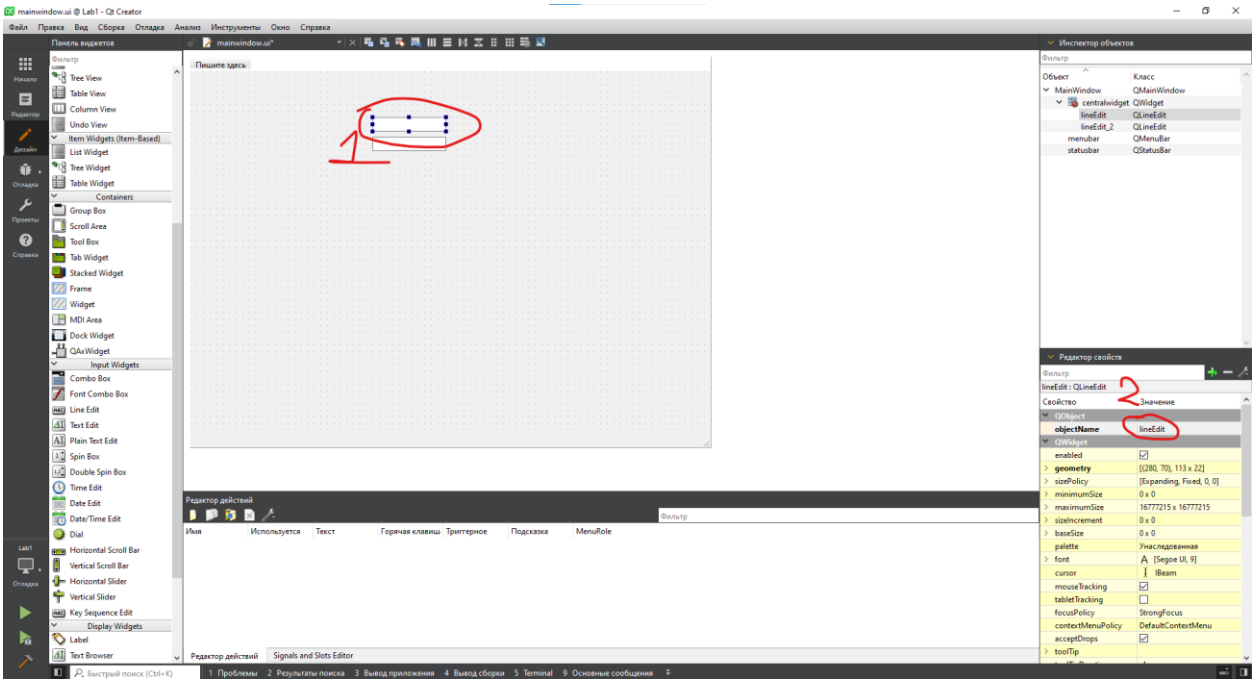
1. Поле для ввода первого числа;
2. Поле для ввода второго числа;

Результат будет следующим:



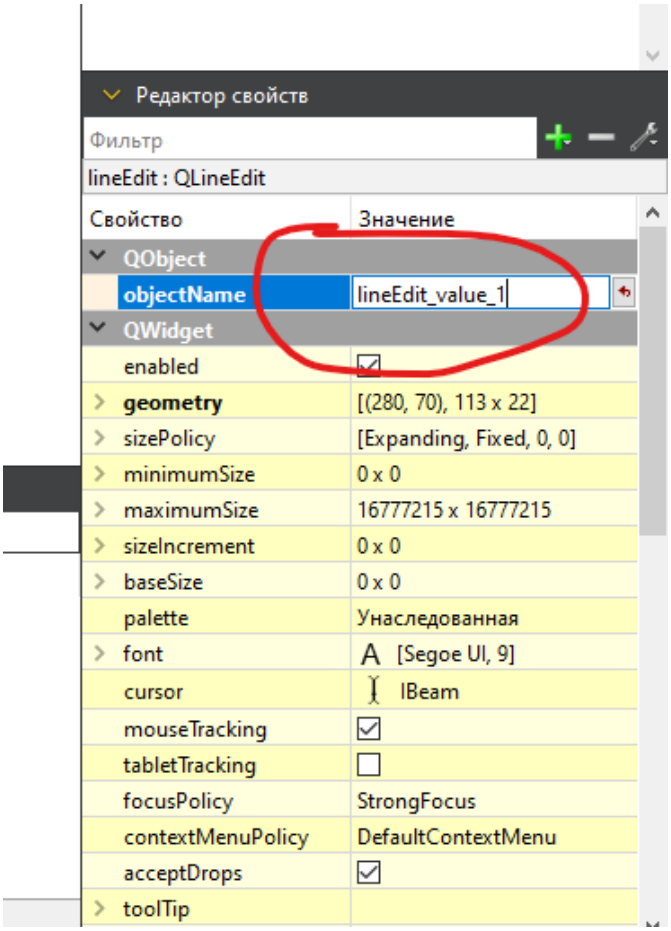
Форма с двумя LineEdit для ввода чисел

Сразу после создания элемента, ему необходимо дать понятное название, по которому мы будем обращаться к нему из кода программы. Для этого в правой части окна имеется боковое меню, разделённое на 2 блока – инспектор объектов, представляющий из себя дерево элементов формы, и редактор свойств – в котором отображаются и редактируются свойства выбранного элемента. Для изменения названия элемента, необходимо нажать на него мышкой и в меню редактора свойств сделать двойной клик на значении свойства “`objectName`”:



Изменение названия виджета

После чего на латинице без пробелов необходимо ввести название компонента, например “`lineEdit_value_1`”:

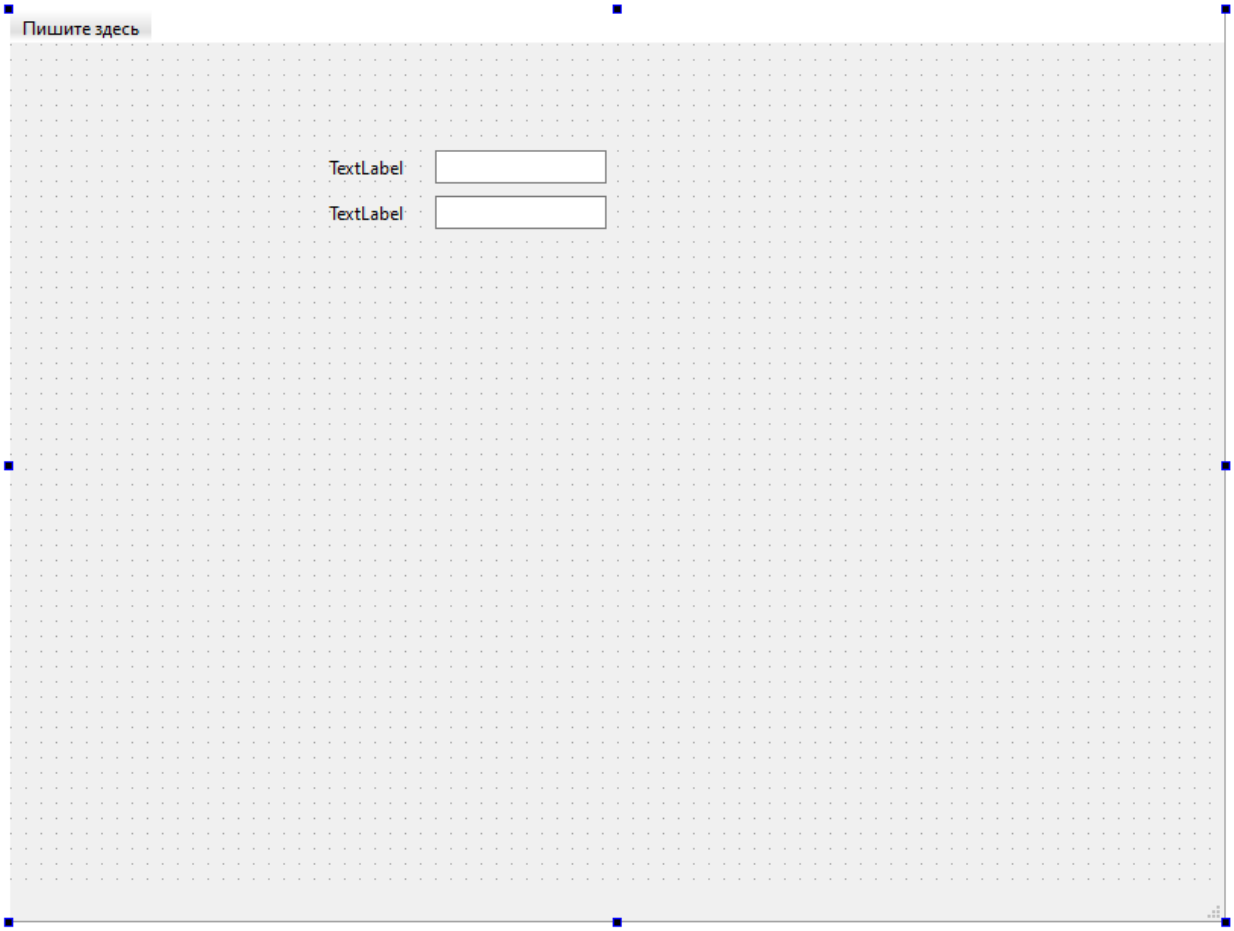


Свойство названия виджета

Префикс “`linedit_`” оставлять не обязательно, однако рекомендуется именовать компоненты в едином стиле, тем самым, оставляя префикс, в дальнейшем, при разработке программы, будет проще найти необходимый компонент в коде, используя поиск по префиксу.

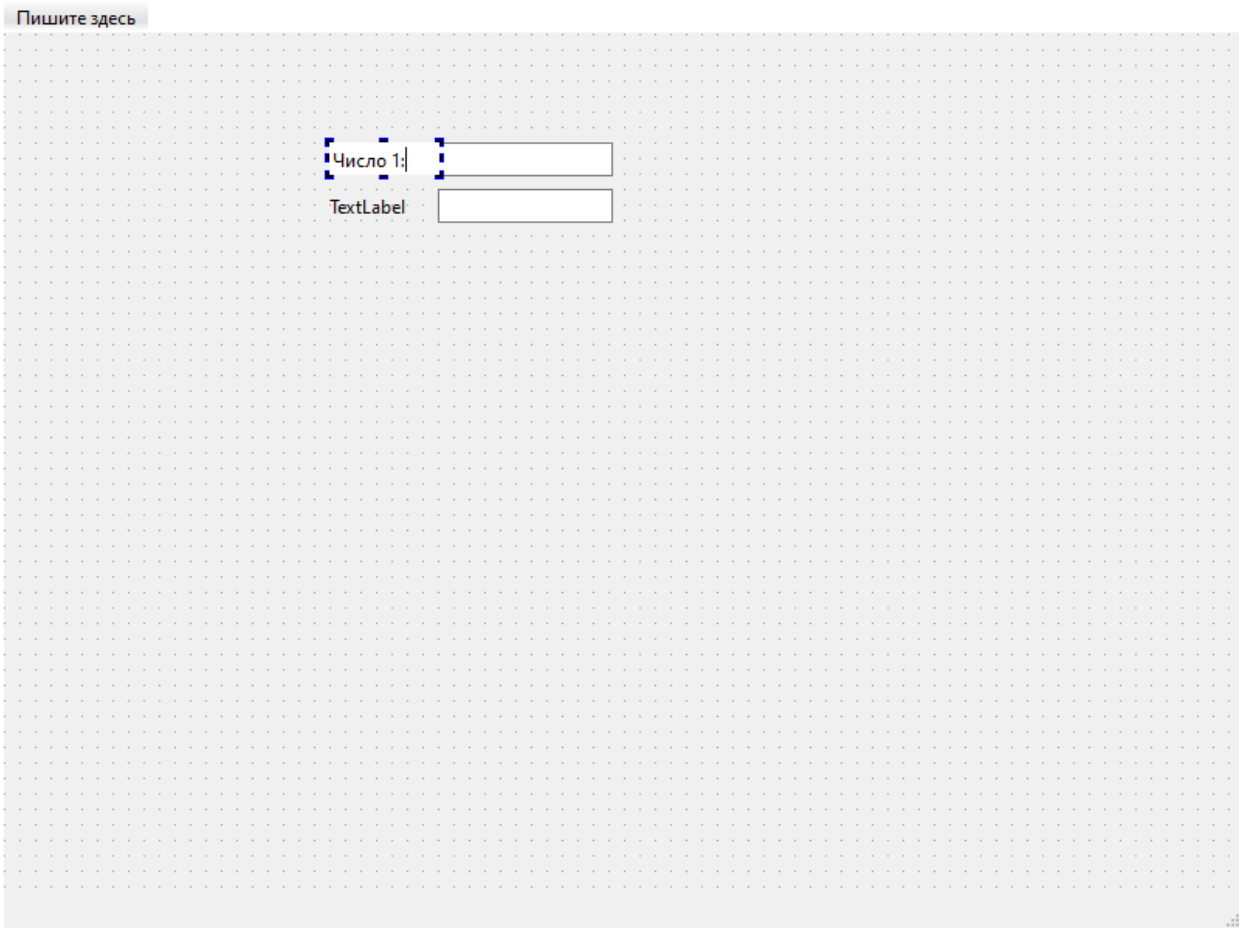
Аналогичные действия необходимо произвести со вторым lineEdit, назвав его, “`lineEdit_value_2`”.

Дальше, рядом с каждым полем добавим по одному Label, который будет указывать для пользователя, что нужно вводить в конкретную строку:



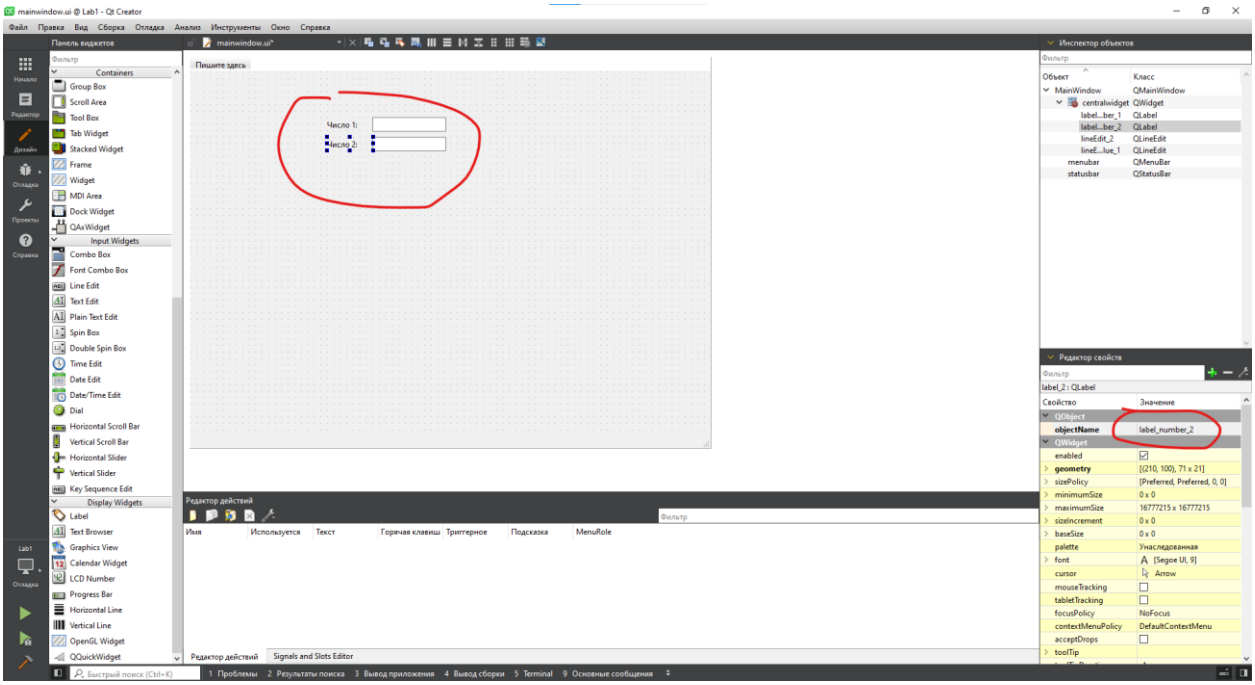
Форма с Label

Сразу после добавления элементов, необходимо задать им текст, для этого кликаем по label и вводим с клавиатуры значение:



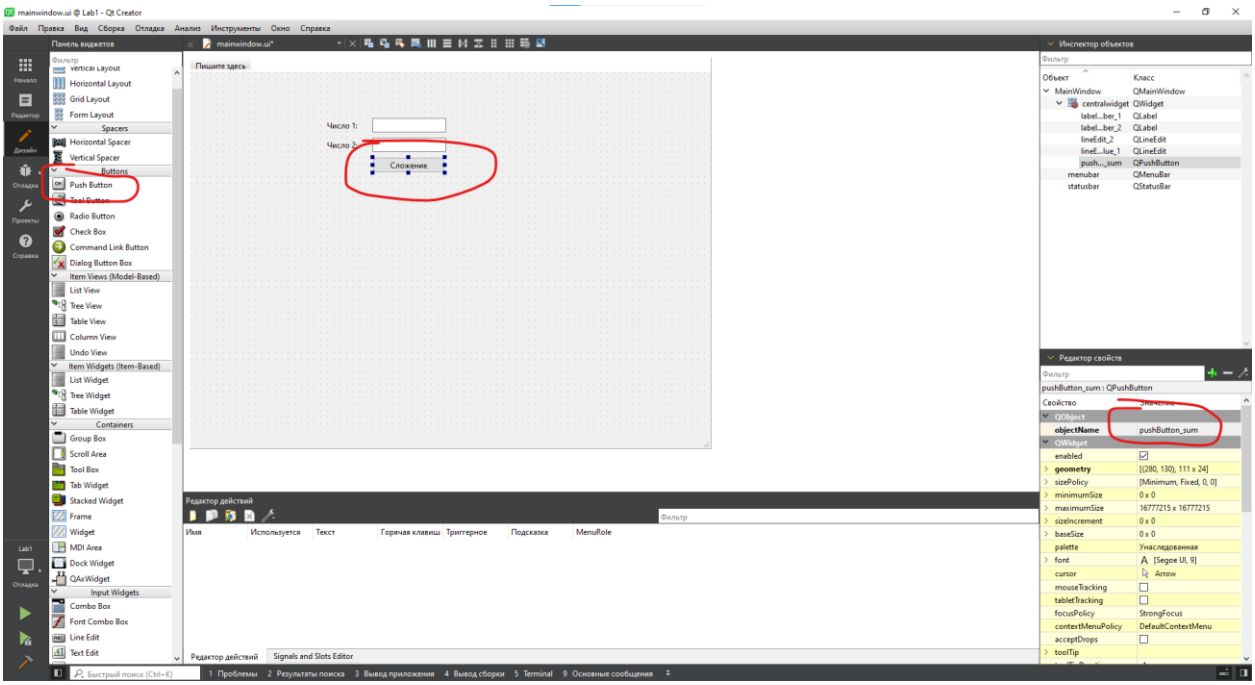
Изменение текста Label

Делаем аналогичные действия со вторым Label и задаем название элемента, как “`label_number_1`” и “`label_number_2`”, соответственно:



Изменение названия объектов Label

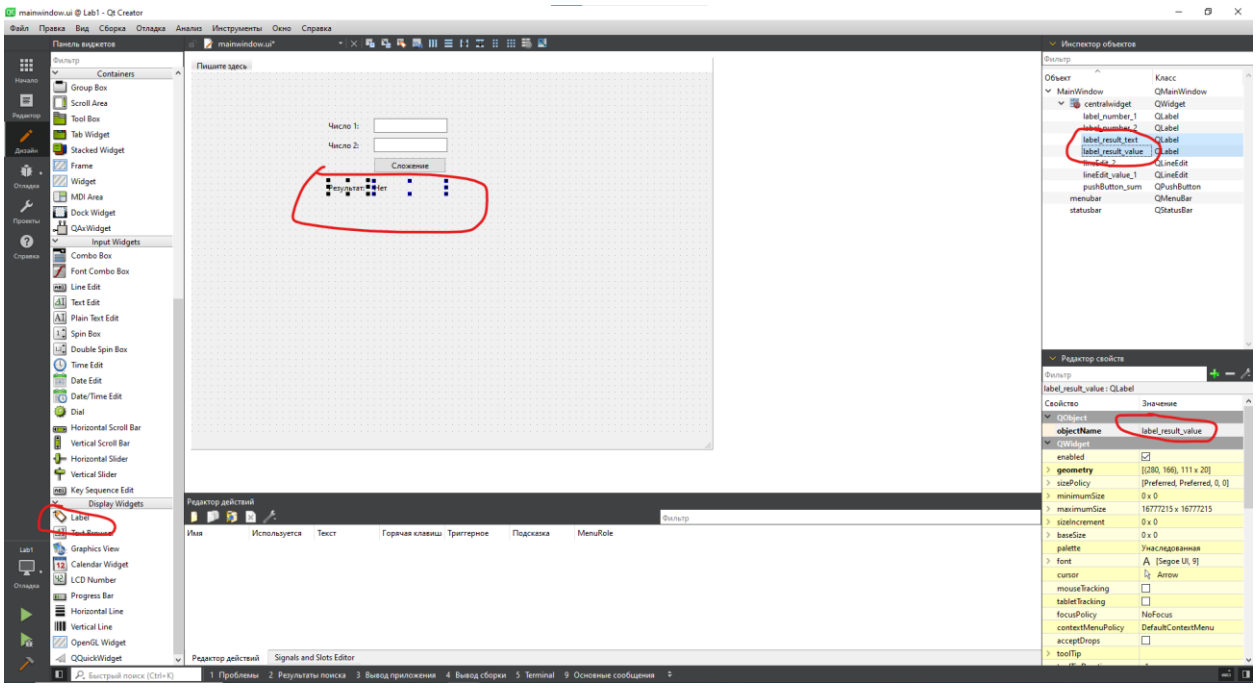
Далее, для того чтобы с введёнными числами можно было взаимодействовать, необходимо добавить кнопку действия. Добавим PushButton, сразу же переименовывая его название:



Форма программы с кнопкой суммы

Нажатие на данную кнопку должно считать сумму введённых чисел и выводить результат. Для вывода результата будем использовать label.

В данном случае нужно добавить на форму 2 label – первый для информирования пользователя, что отображаемое число является результатом действия (`label_result_text`), второй – для вывода результата (`label_result_value`):

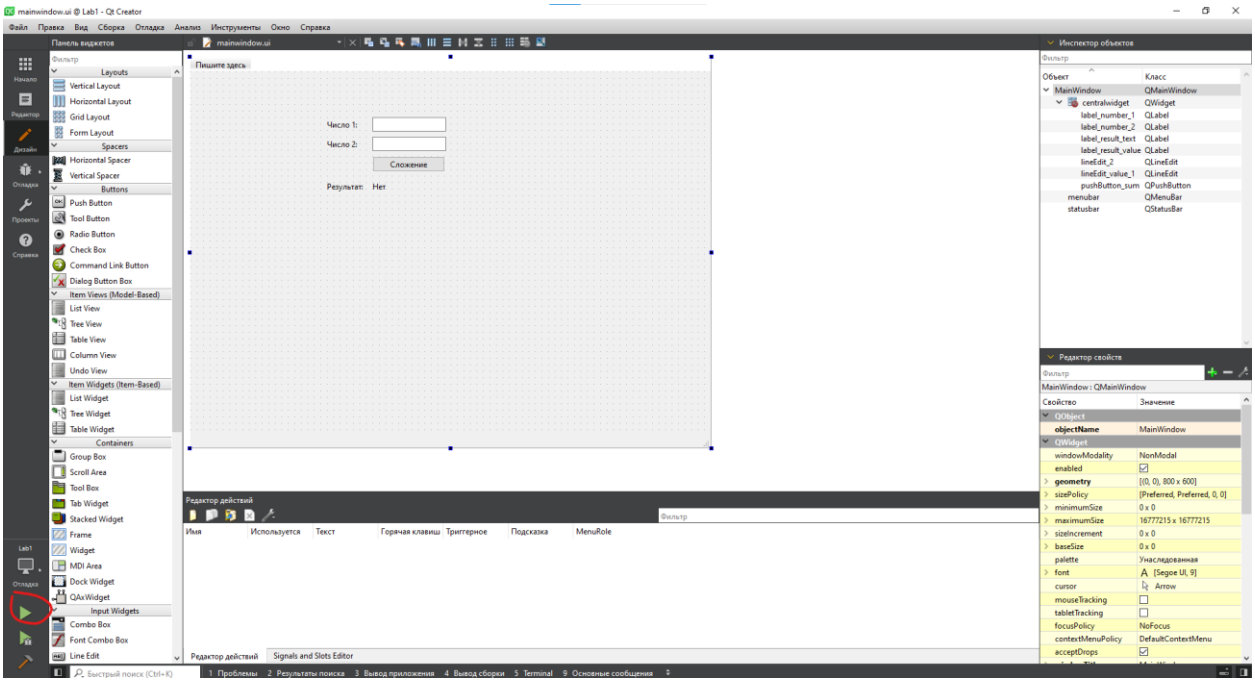


Дополнительные Label для вывода результата

Теперь, когда на форму добавлены все необходимые элементы, можно переходить к написанию программного кода, для взаимодействия с виджетами.

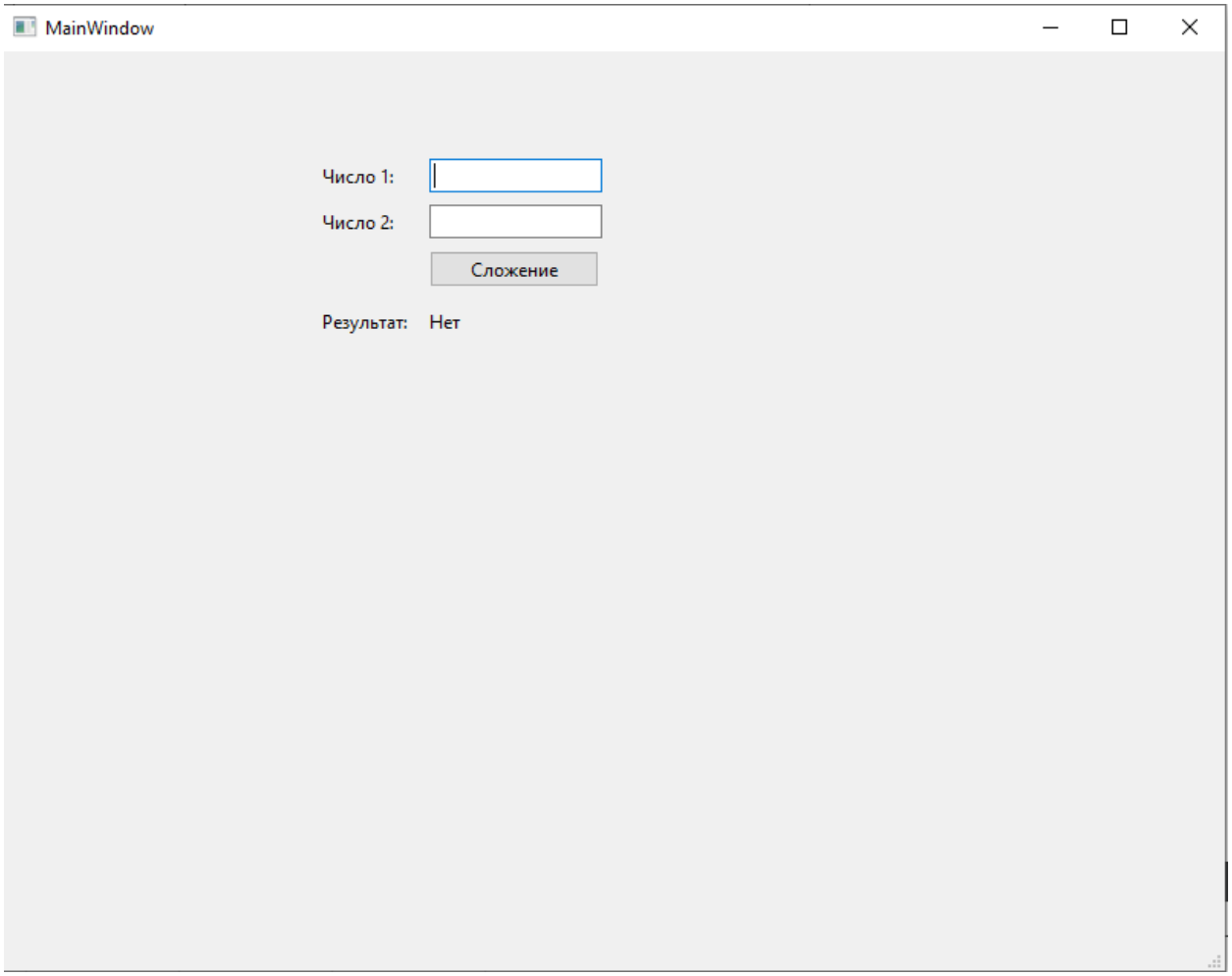
Для того, чтобы в программе появился доступ к виджетам, после их редактирования на форме, программу надо скомпилировать – при компиляции Qt преобразовывает виджеты из редактора в их реализацию в виде кода, с которым можно будет взаимодействовать.

Компиляция программы выполняется с помощью нажатия горячих клавиш “Ctrl+R” или путём нажатия на кнопку запуска программы в интерфейсе Qt:



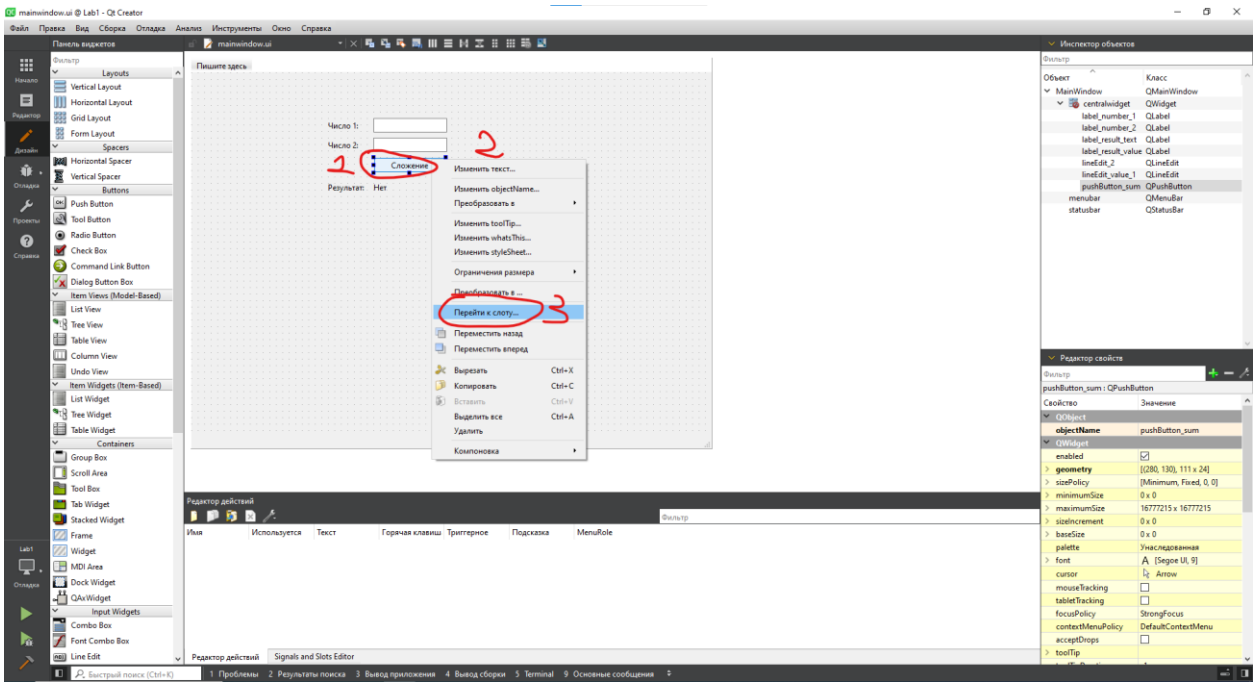
Нопка компиляции и запуска программы

В результате будет скомпилирована и запущена наша программа:



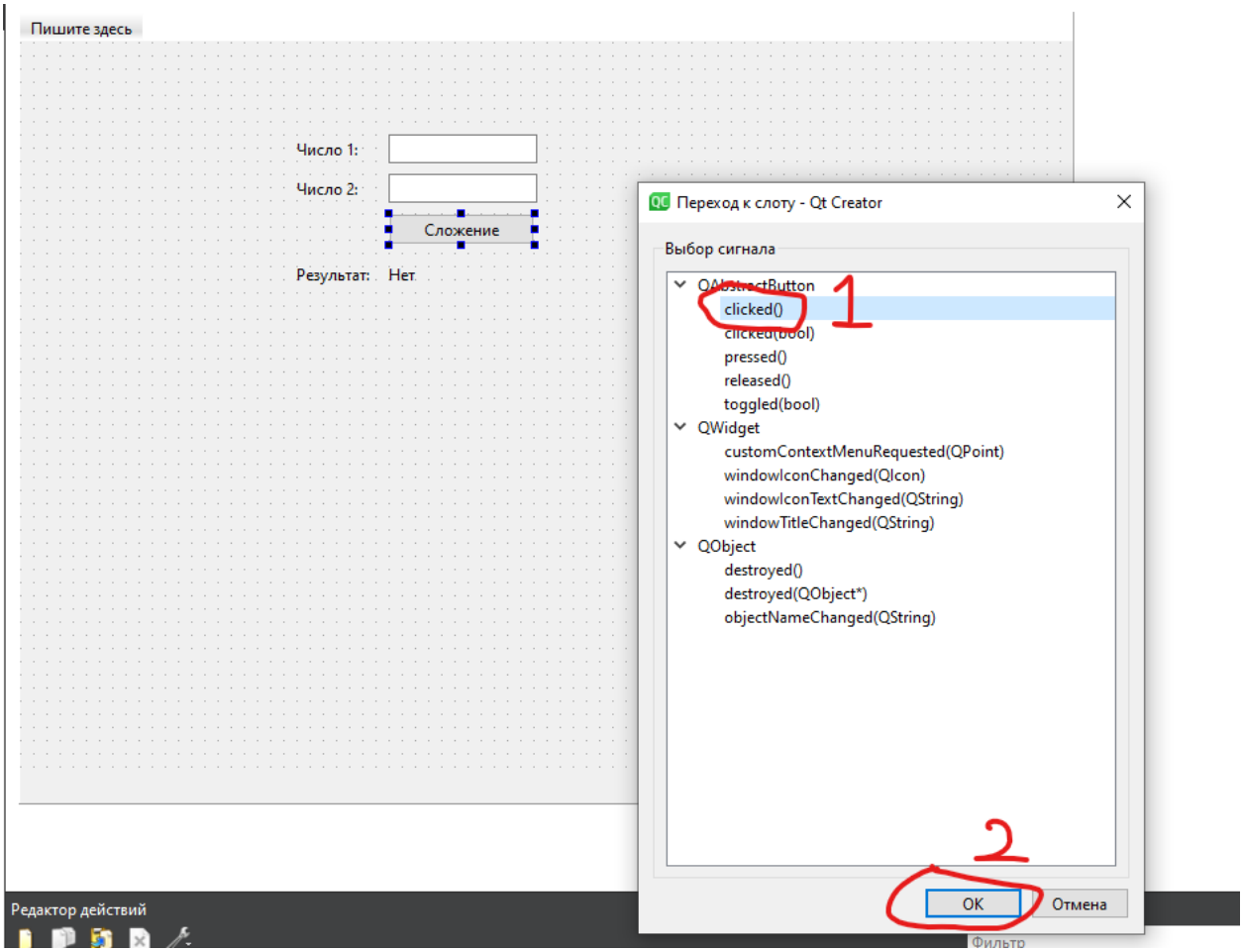
Запущенная программа

В данный момент времени нажатие на кнопку не будет давать никакого эффекта. Для того, чтобы обработать нажатие кнопки, необходимо вернуться в редактор интерфейса Qt в файл mainwindow.ui, в котором нажать левой клавишей мыши на нашу кнопку, тем самым выделив её, а затем нажать на неё правой клавишей мыши, открыв меню действий и выбрать пункт «Перейти к слоту»:



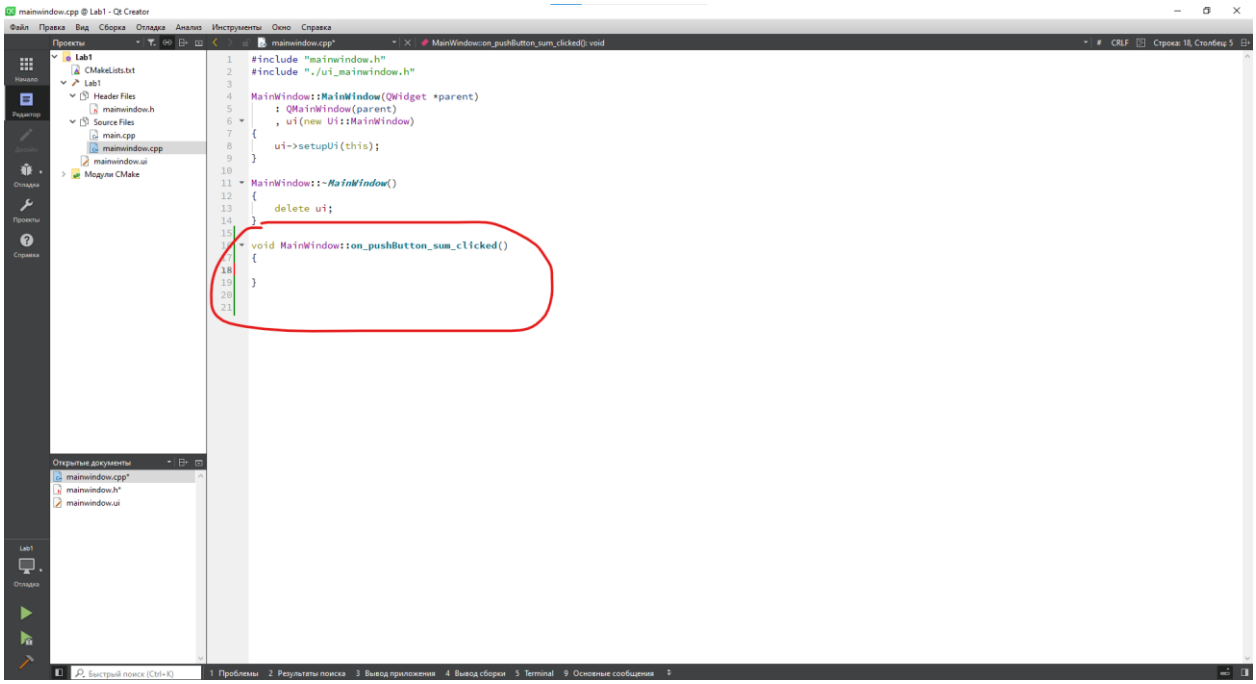
Первый шаг добавления действия на кнопку

В появившемся окне отображаются возможные обработчики, которые можно добавить для данной кнопки. В данной задаче необходимо выбрать `clicked()` и нажать на кнопку OK:



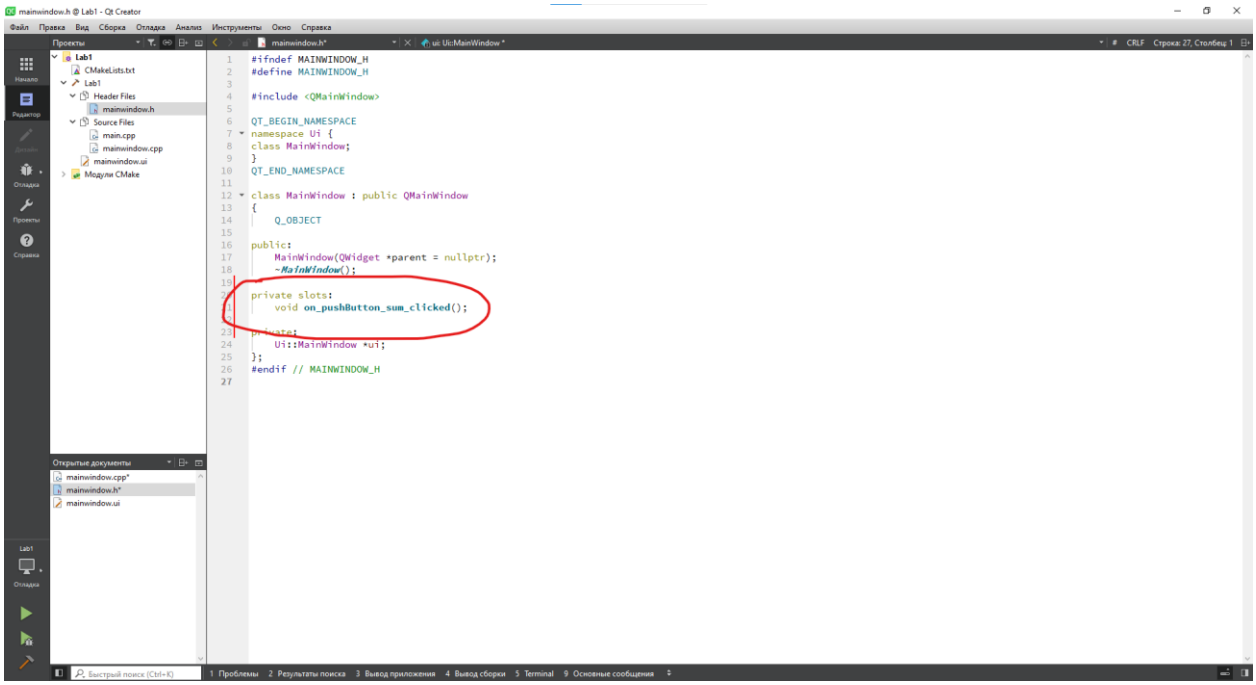
Выбор обработчика для кнопки.

После этого Qt автоматически откроет редактор кода и файл mainwindow.cpp, в котором автоматически будет создан пустой метод обработки нажатия конкретной кнопки:



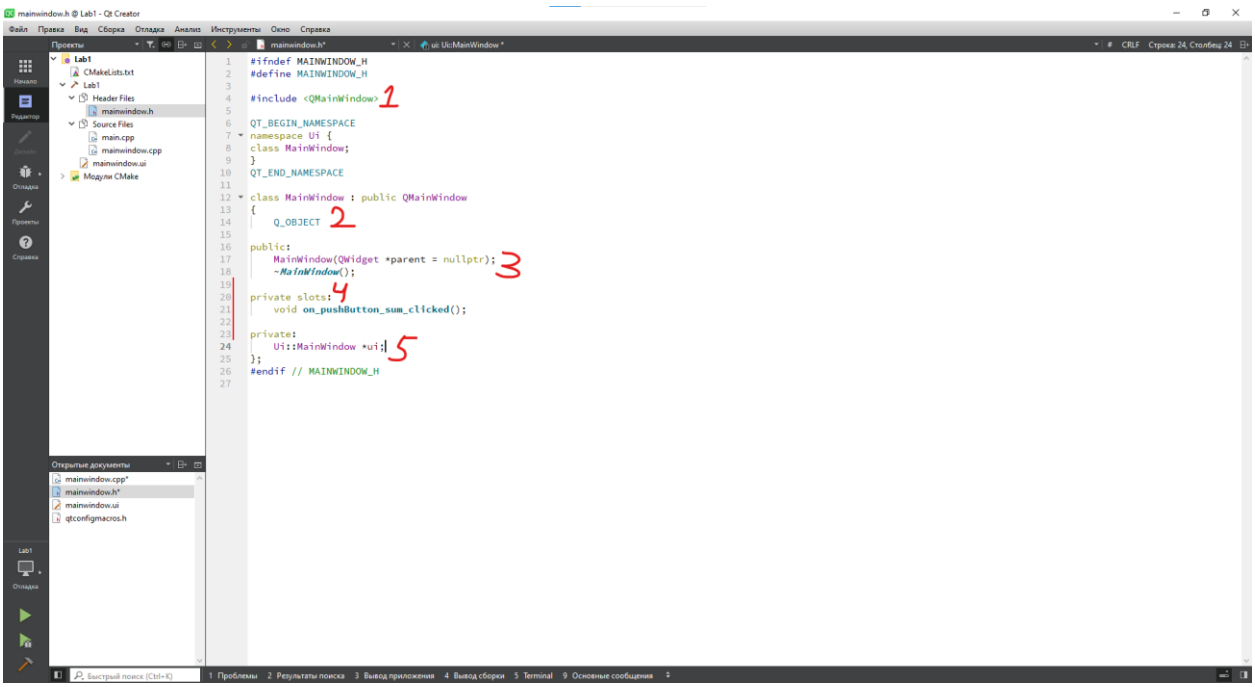
Автоматически созданный метод обработки нажатия кнопки

Также, в заголовочном файле mainwindow.h автоматически была создана строка объявления данного метода:



Объясвление метода

На данном этапе подробно разберём структуру файлов:



Структура заголовочного файла

В заголовочном файле отмечены цифрами от 1 до 5 основные элементы кода:

1 – Подключение класса QMainWindow, базовый класс Qt, на основе которого происходит разработка главного окна программы.

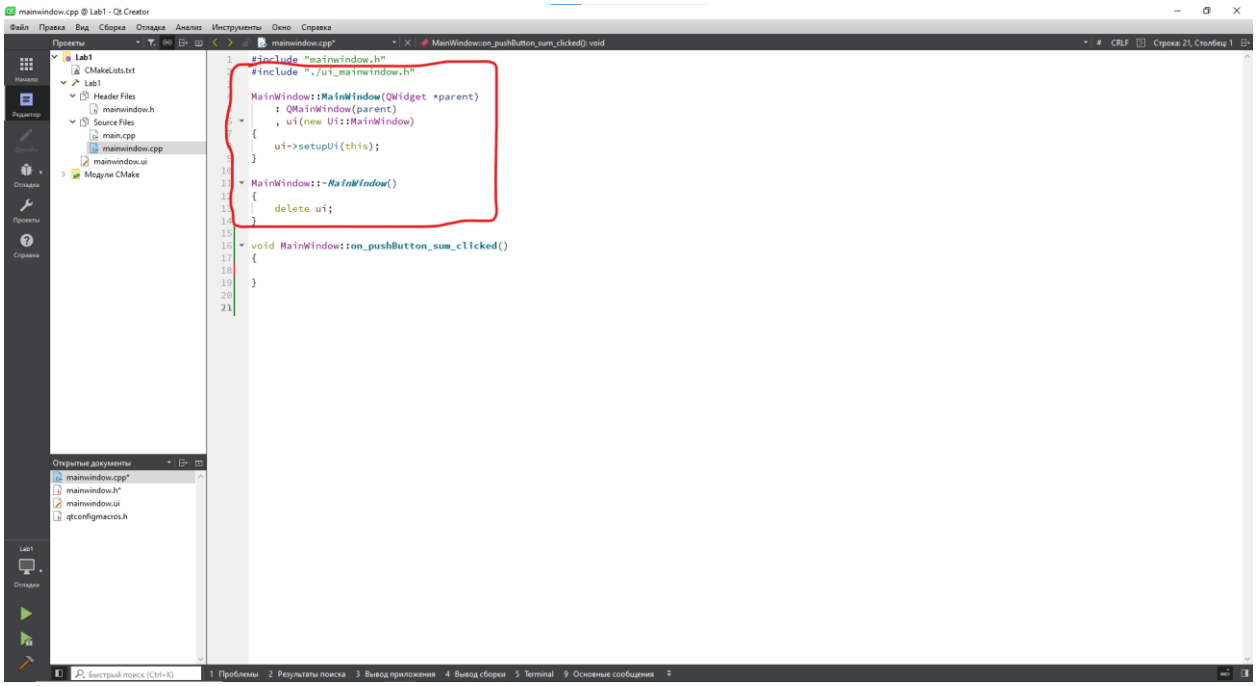
2 – Директива препроцессора Q_OBJECT, добавляет объявление базовых механизмов Qt, которые можно использовать дальше, без неё невозможна работа классов, использующих компоненты Qt.

3 – Конструктор и деструктор класса главного окна программы.

4 – Объявление слотов. Слоты в Qt это особые методы, которые можно не только вызвать напрямую из кода, но и настроить их вызов при определённом событии, например, нажатии на кнопку.

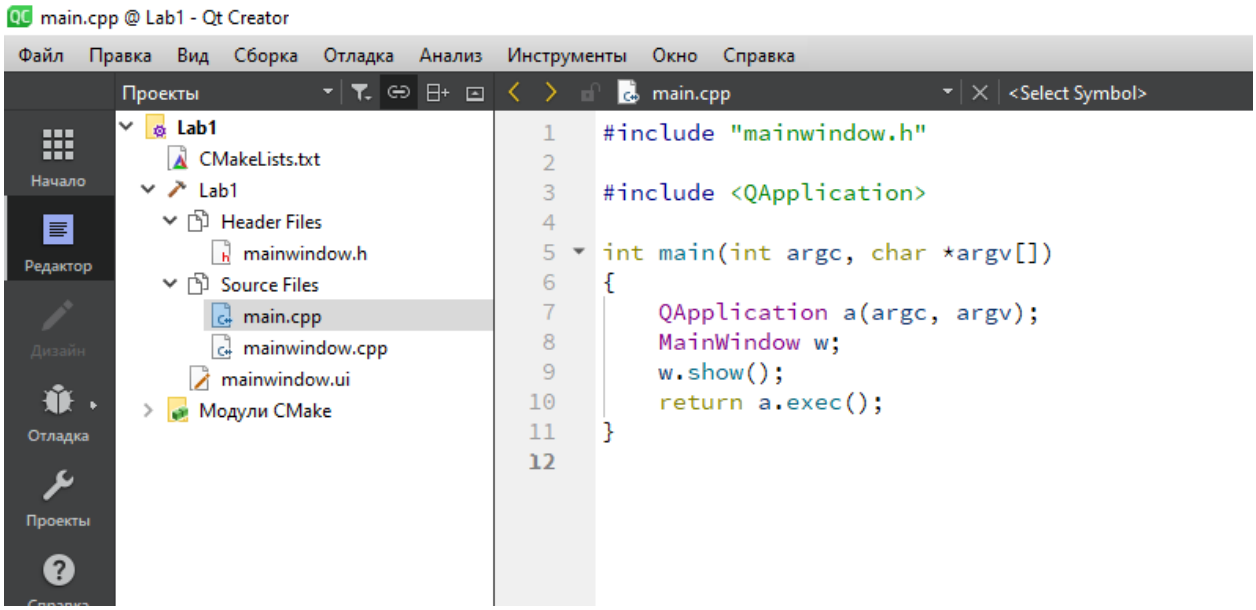
5 – Поле указателя на объект интерфейса. В Qt интерфейс, созданный в редакторе, компилируется в C++ код, благодаря чему, возможно взаимодействие с виджетами как с объектами. Именно поэтому, после создания пользовательского интерфейса в редакторе, необходимо скомпилировать программу.

Остальной код, необходимый для корректной работы графического интерфейса создаётся автоматически в файле исходных кодов mainwindow.cpp:



Код создания инициализации графического интерфейса

Запуск же программы происходит из файла main.cpp, в котором находится главная функция `main()`, внутри которой создаётся объект приложения Qt (QApplication), запускающий базовые функции Qt, а также объект нашего MainWindow, в котором создаётся интерфейс и обрабатываются события нажатия кнопок:



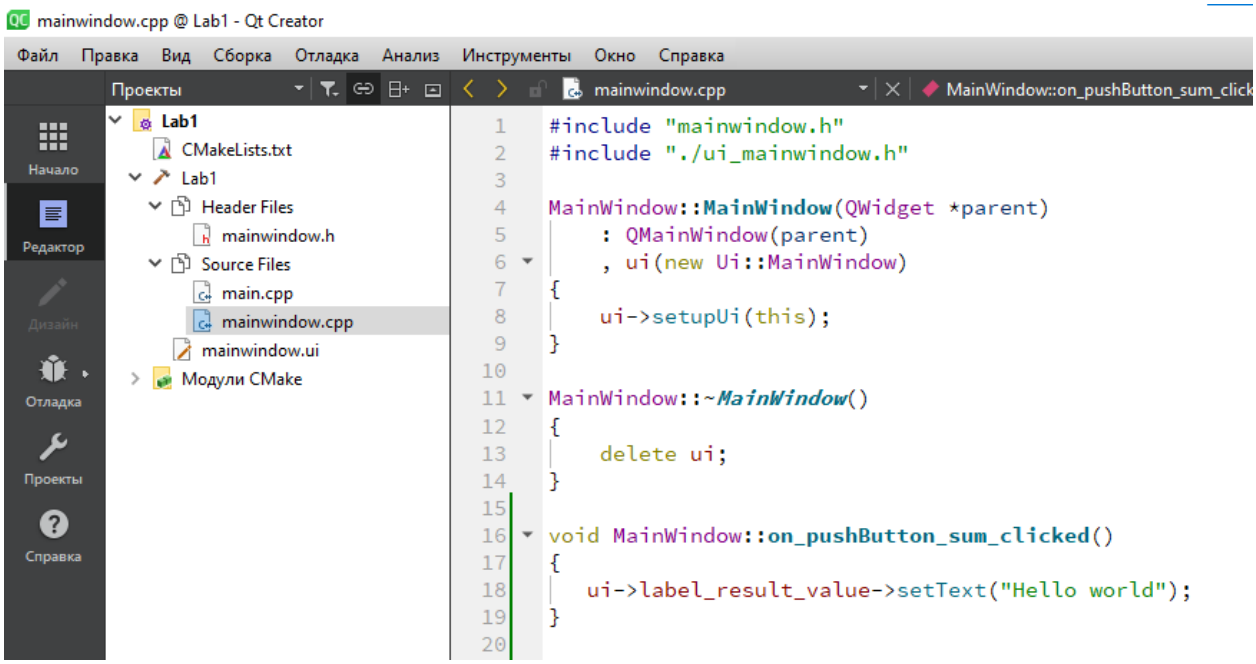
Файл main.cpp

Теперь вернёмся обратно в созданный обработчик нажатия кнопки в mainwindow.cpp и для начала сделаем простой вывод текста на экран, по нажатию на кнопку.

Для вывода информации в label необходимо обратиться к объекту интерфейса программы (*ui), вызвать указатель на объект необходимого компонента, в данном случае – `label_result_value`, у которого вызвать метод `setText()`, передав в который строку с текстом для вывода:

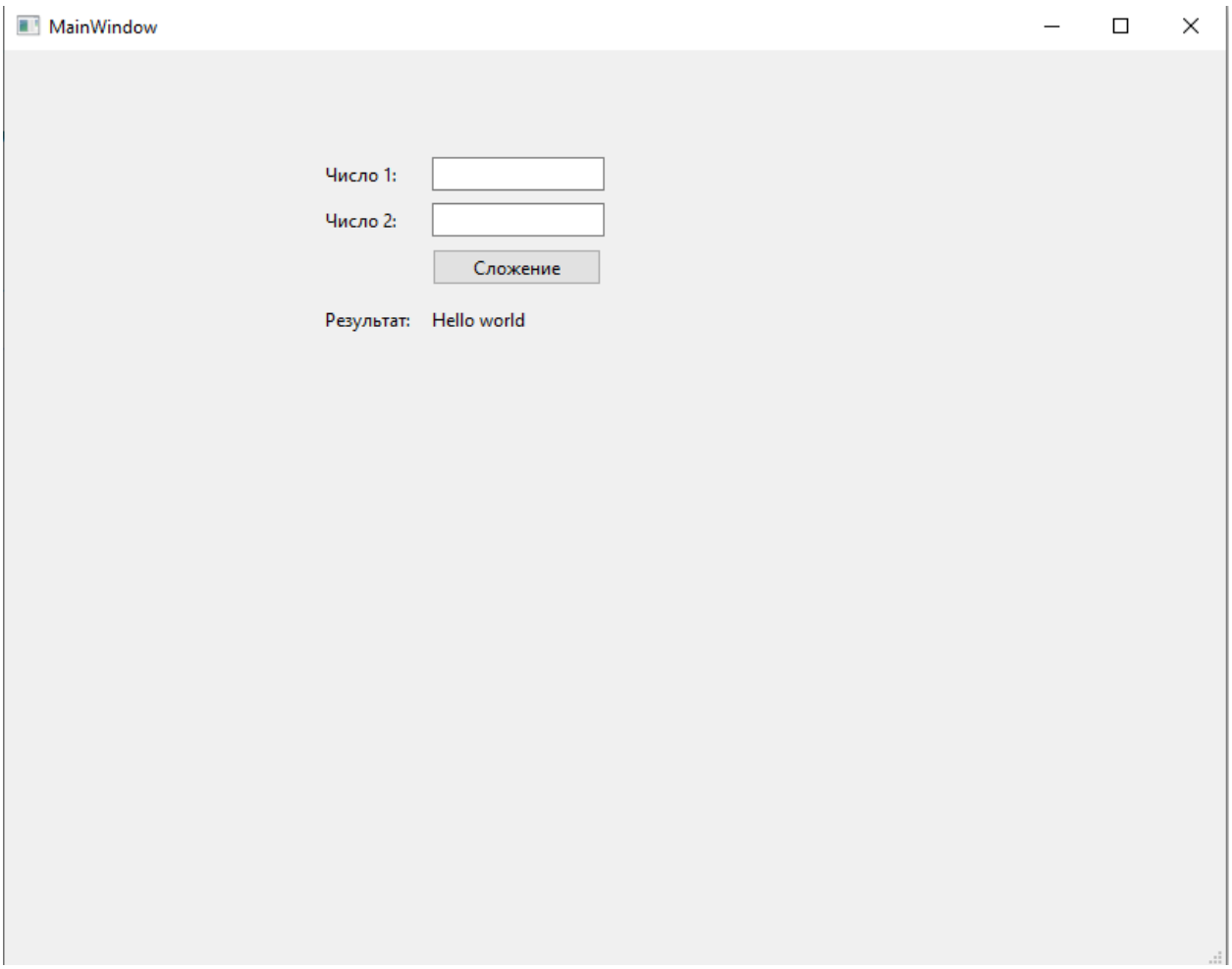
```
ui->label_result_value->setText("Hello world");
```

Таким образом получаем следующий код:



Вывод текста на экран

В результате запуска программы и нажатия на кнопку сложения, в поле результата появится текст “Hello world”:



Результат нажатия на кнопку

Однако, нам необходимо выполнить сложение двух чисел, введённых пользователем. Для этого необходимо обратиться к элементам lineEdit и вызвать у них метод `text().toInt()`:

```
11  MainWindow::~MainWindow()
12  {
13      delete ui;
14  }
15
16  void MainWindow::on_pushButton_sum_clicked()
17  {
18      int value_1 = ui->lineEdit_value_1->text().toInt();
19      int value_2 = ui->lineEdit_value_2->text().toInt();
20  }
21
22
```

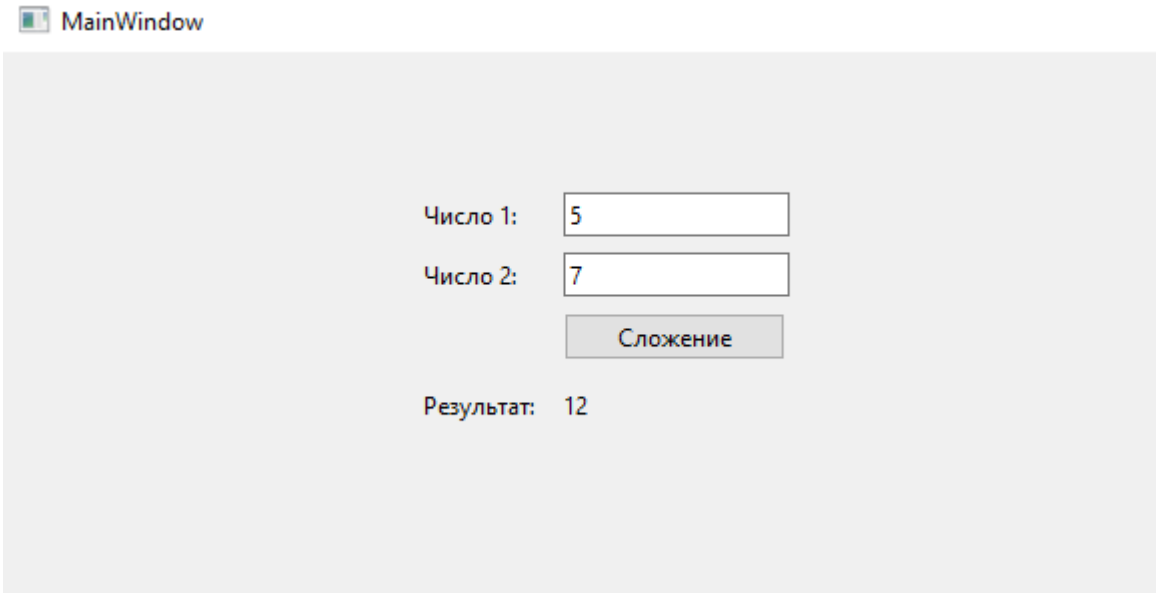
Получение числового значения, введённого пользователем

Следующим шагом необходимо выполнить сложение и вывести число на экран. Вывод числа производится путём вызова у объекта label метода `setNum()`, получая следующий код:

```
16  void MainWindow::on_pushButton_sum_clicked()
17  {
18      int value_1 = ui->lineEdit_value_1->text().toInt();     // получение первого числа
19      int value_2 = ui->lineEdit_value_2->text().toInt();     // получение второго числа
20
21      int sum = value_1 + value_2;    // считаем сумму двух чисел
22
23      ui->label_result_value->setNum(sum);    // выводим полученную сумму на экран
24  }
25
```

Код суммы двух чисел

Запустив программу и введя 2 числа в соответствующие поля, после нажатия на кнопку суммы, будет выведена сумма этих чисел:



Результат выполнения программы

Таким образом был создан графический интерфейс пользователя и написан код для сложения двух чисел.

Задание

Используя полученные знания, добавьте к кнопке сложения дополнительно кнопки для действий «вычитание», «умножение» и «деление», указав корректные названия элементов, как в интерфейсе программы, так и в коде программы. Обратите внимание, что делить на ноль нельзя!

Контрольные вопросы для защиты работы

1. Реализация дополнительной математической операции.