

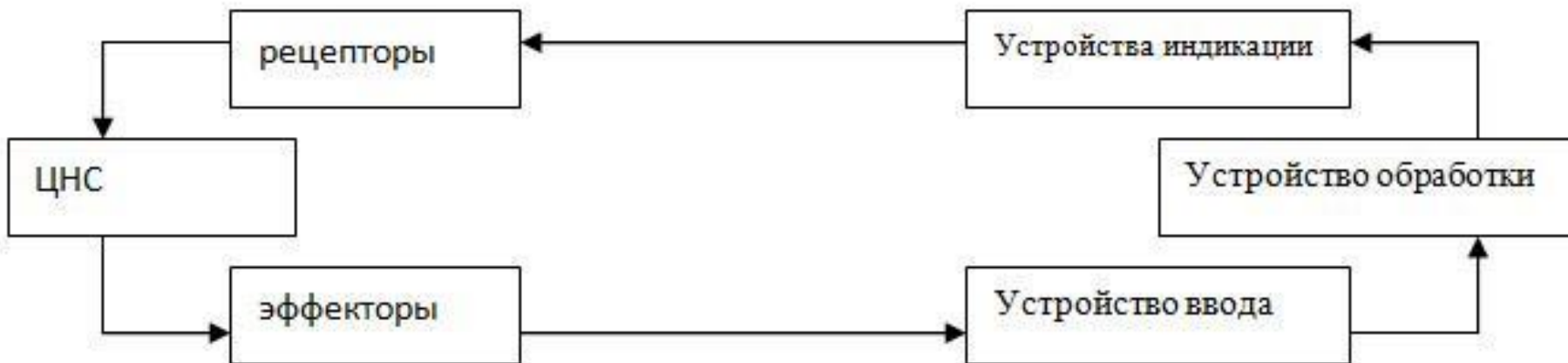
Человеко-машинное взаимодействие

Лекция 1

Мерзлякова Екатерина Юрьевна

к.т.н. доцент ПМиК

Основные вопросы человеко-машинного взаимодействия



Пользовательский интерфейс программы - это совокупность элементов, позволяющих пользователю программы управлять ее работой и получать требуемые результаты.

Основные вопросы человеко-машинного взаимодействия

НО СЗЙЧ4С Н4 Э70Й
С7Р0К3 84Ш Р4ЗУМ
ЧN7437 Э70
4870М47NЧ3СКН,
Н3 349УМЫ84ЯСЬ 06
Э70М. ГОР9НСЬ.
ЛНШЬ ОПР393ЛЗННЫЗ
ЛЮ9N МОГУ7 ПРОЧN747Ь
Э70.



QtCreator

- программирования графического пользовательского интерфейса;
- сетевого программирования (сокеты, работа с СУБД, HTTP, XML, JSON);
- работы с мультимедийными данными;
- программирования под мобильную платформу;
- интернационализации приложений;
- рефлексивного программирования (поддержка динамической типизации, получение информации о типах, создание объектов по имени класса и изменение их свойств).

Установка Qt Creator 5.2

http://download.qt-project.org/official_releases/online_installers.



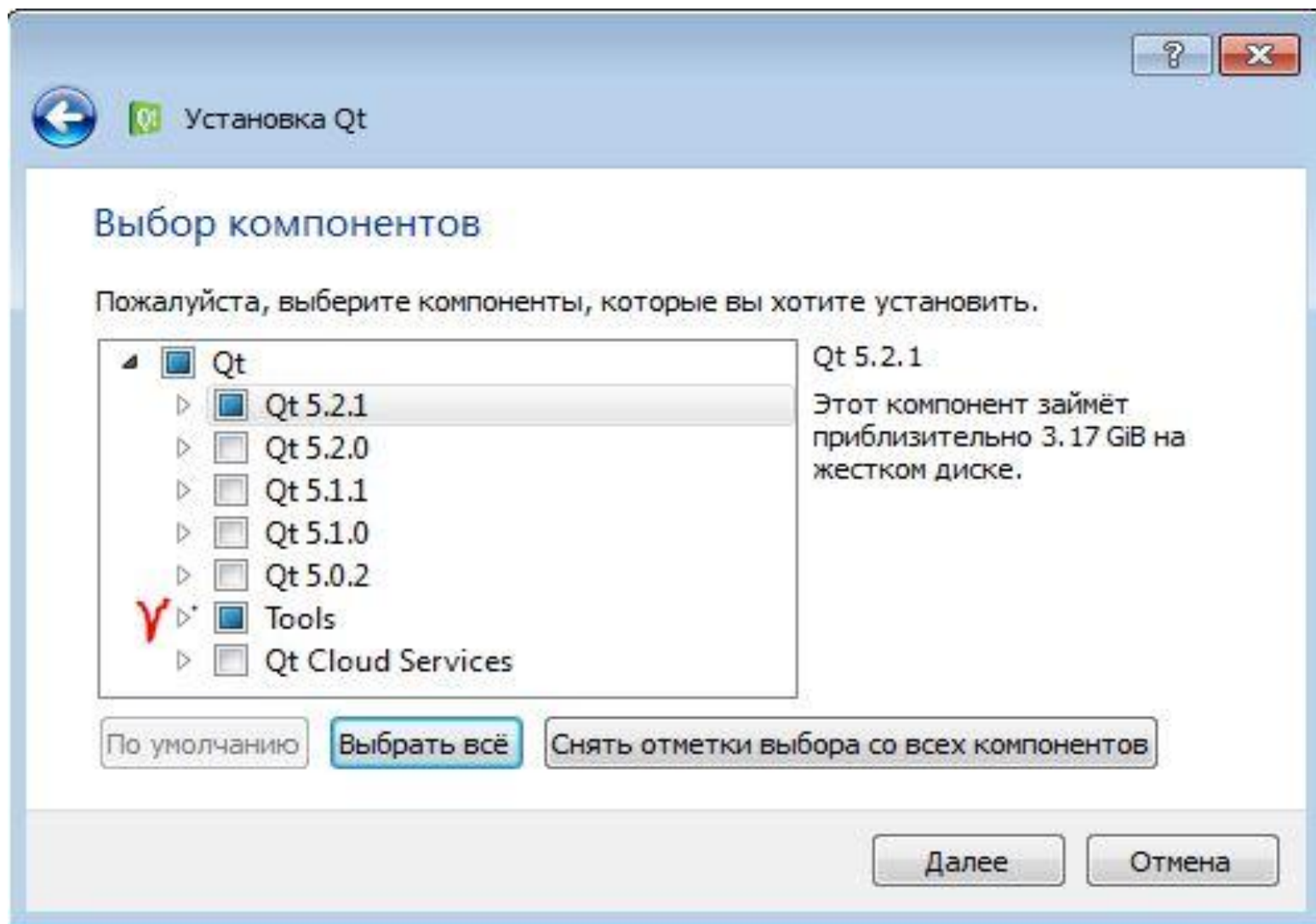
Qt Downloads

[Qt Home](#) [Bug Tracker](#) [Code Review](#)

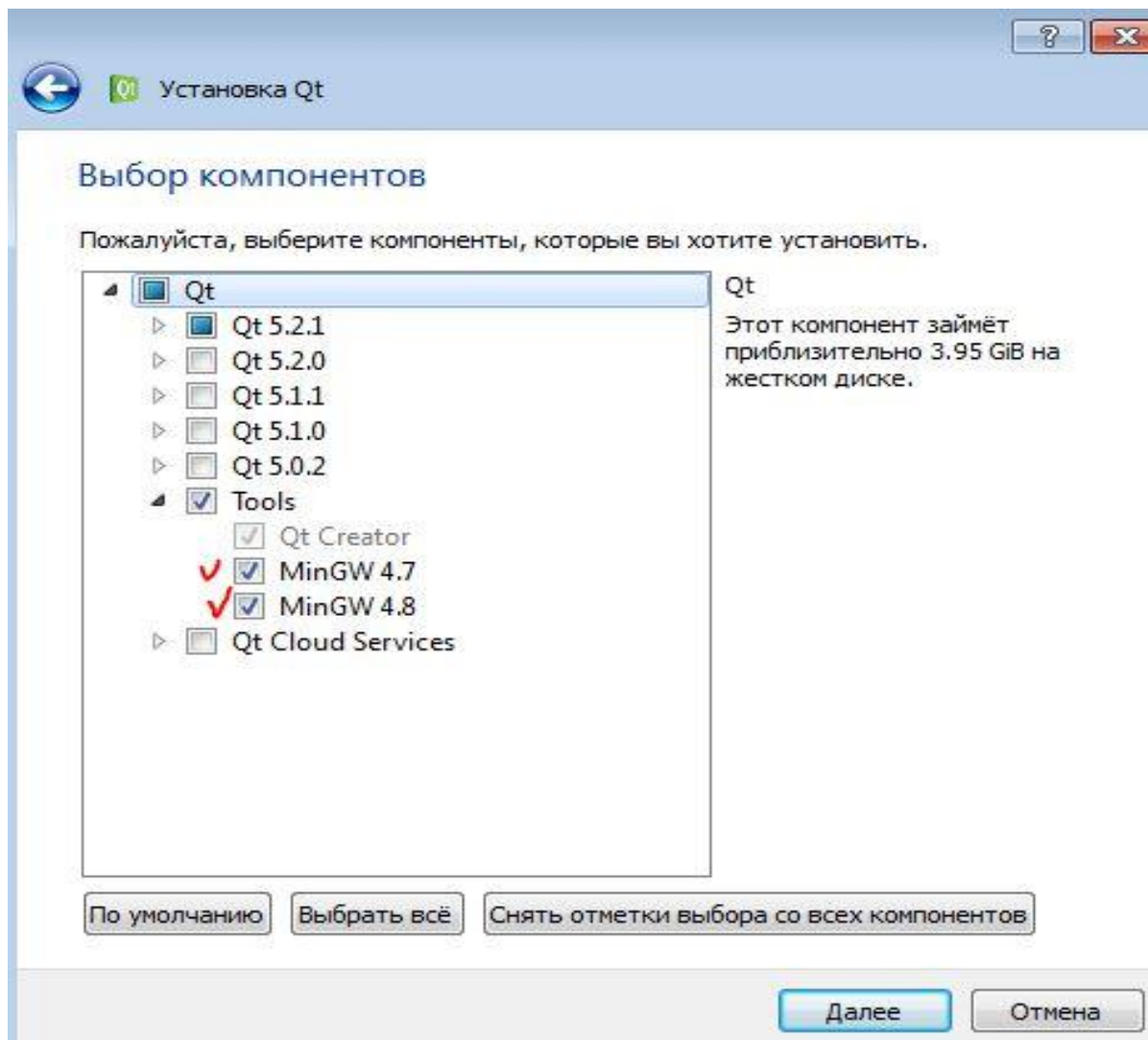


Name	Last modified	Size
↑ Parent Directory		-
📁 1.5/	11-Dec-2013 16:28	-
📁 1.4/	05-Feb-2014 08:41	-
📄 qt-opensource-windows-x86-1.5.0-1-online.exe	04-Feb-2014 13:22	14M
📄 qt-opensource-mac-x64-1.5.0-1-online.dmg	04-Feb-2014 13:29	9.3M
📄 qt-opensource-linux-x86-1.5.0-1-online.run	04-Feb-2014 13:27	23M
📄 qt-opensource-linux-x64-1.5.0-1-online.run	04-Feb-2014 13:27	22M
📄 md5sums.txt	05-Feb-2014 08:42	308

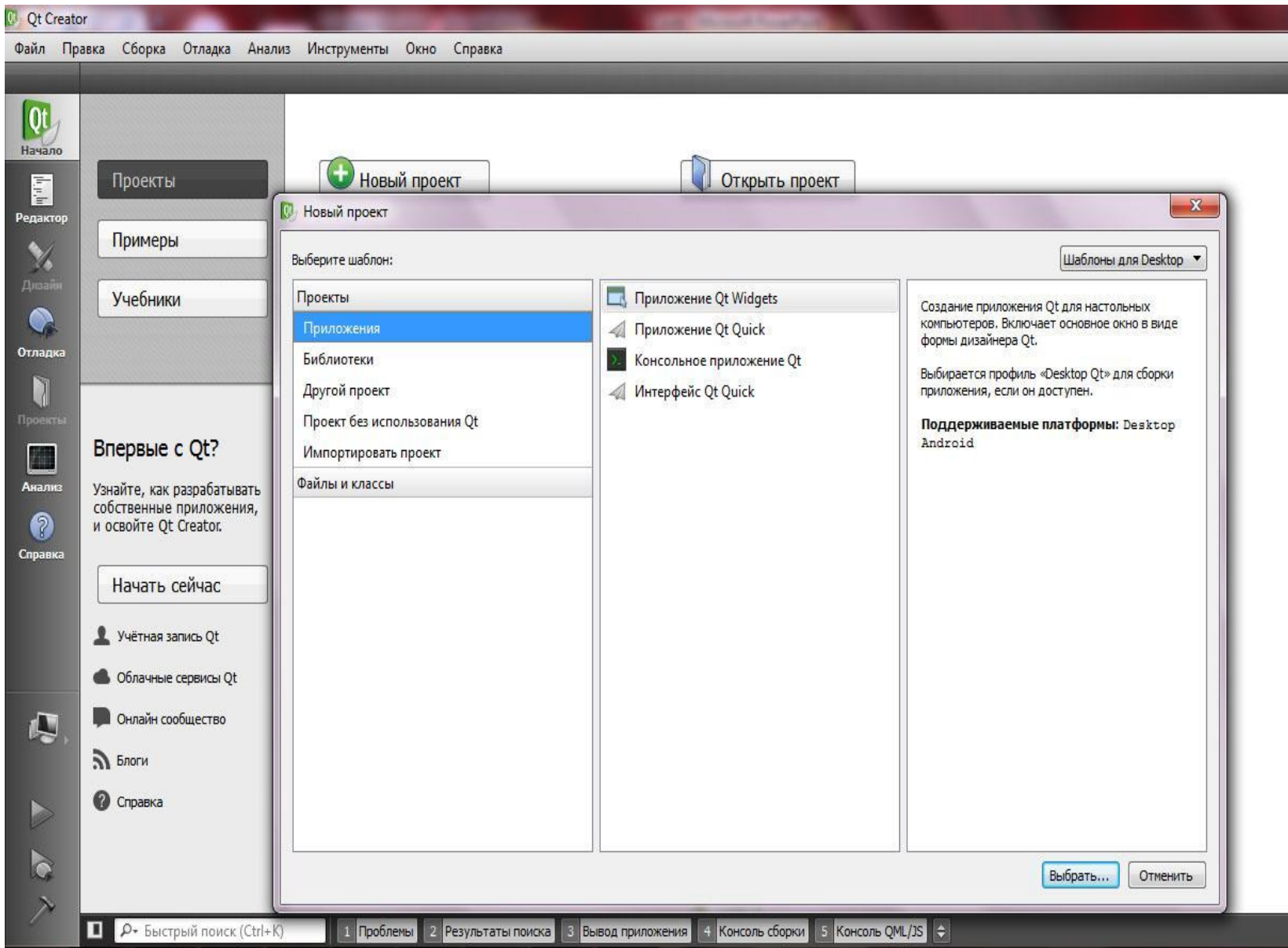
Установка Qt Creator 5.2



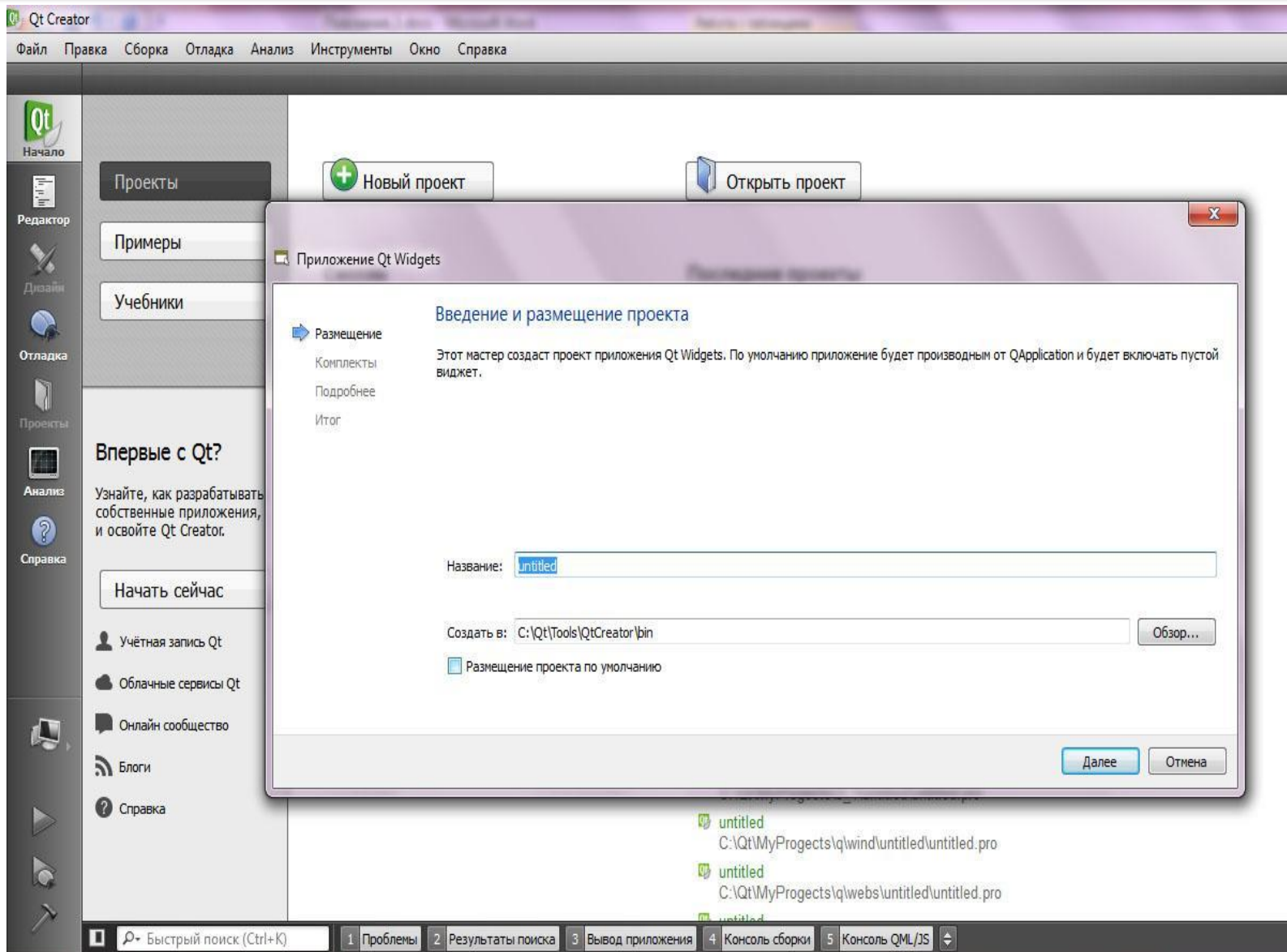
Установка Qt Creator 5.2



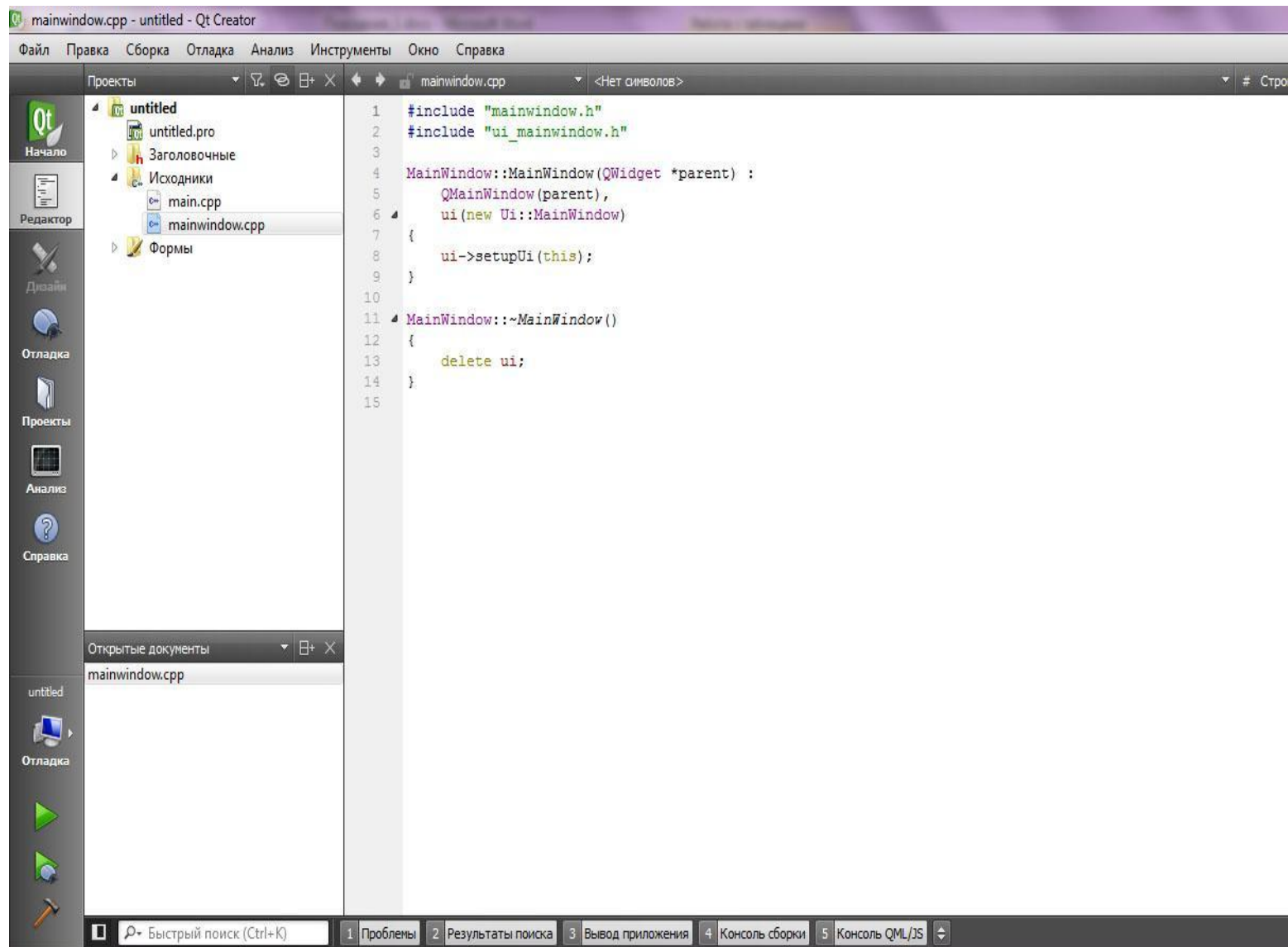
Создание приложения



Создание приложения



Создание приложения



Интеграция справки по Qt

```
def MAINWINDOW_H
.ne MAINWINDOW_H

.ude <QMainWindow>

space Ui {
MainWindow;

MainWindow : public QMainWindow

}_OBJECT

.c:
explicit MainWindow(QWidget *parent = 0);
MainWindow();

ite:
Ui::MainWindow *ui;

.f // MAINWINDOW_H
```

QMainWindow Class

Qt 5.3 > Qt Widgets > C++ Classes > QMainWindow

Qt 5.3.2 Reference Document

The QMainWindow class provides a main application window. [More...](#)

Header: #include
<QMainWindow>

qmake: QT += widgets

Inherits: QWidget.

- List of all members, including inherited members

Public Types

enum DockOption {
AnimatedDocks,
AllowNestedDocks,
AllowTabbedDocks,
ForceTabbedDocks,
VerticalTabs }

Contents

- Public Types
- Properties
- Public Functions
- Public Slots
- Signals
- Protected Functions
- Detailed Description
- Qt Main Window Framework
- Creating Main Window Components
- Creating Menus
- Creating Toolbars
- Creating Dock Widgets
- The Status Bar

Литература по Qt

- Бланшет Ж., Саммерфилд М. QT 4: программирование GUI на C++. КУДИЦ-Пресс, 2008.
- Саммерфилд М. Qt Профессиональное программирование. Символ-Плюс, 2011. 552 с.
- Шлее М. Qt 4.5. Профессиональное программирование на C++. БХВ-Петербург, 2009. 896 с.
- <http://doc.crossplatform.ru/qt/4.6.x/examples.html> - Примеры программ на Qt, учебное пособие.
- http://www.opennet.ru/docs/RUS/qt3_prog/qt3.html - Разработка графического интерфейса с помощью библиотеки Qt3.

Панели вывода. Панель проблемы

The image shows a screenshot of the Qt IDE interface. At the top, a blue banner contains the title "Панели вывода. Панель проблемы". Below this, the main workspace is divided into several panels:

- Code Editor:** Displays C++ code with line numbers 11, 12, and 13. Line 11 contains the statement `return a.exec();`, which is underlined in red, indicating an error.
- Open Documents:** A list on the left side shows the following files: `main.cpp`, `mainwindow.cpp`, `mainwindow.h`, and `mainwindow.ui`.
- Problems Panel:** Located at the bottom right, it lists four compilation errors, each with a red error icon and a message: `stray '\320' in program`. The first error is highlighted in blue. The file path `C:\Qt\MyProjects\lek1\untitled\main.cpp` is shown below the first error.
- Bottom Panel:** A navigation bar at the very bottom contains five tabs: `Быстрый поиск (Ctrl+K)`, `1 Проблемы` (with a blue circle around the number 4), `2 Результаты поиска`, `3 Вывод приложения`, `4 Консоль сборки`, and `5 Консоль QML/JS`.

Панели вывода. Вывод приложения

The screenshot displays the Qt Creator IDE interface. On the left, the 'Toolbox' contains widget classes such as 'Vertical Spacer', 'Buttons' (with sub-items: Push Button, Tool Button, Radio Button, Check Box, Command Link Button, Button Box), and 'Item Views (Model-Based)' (with sub-item: List View). The main workspace shows a grid-based widget layout. Below the workspace is the 'Action Editor' and 'Signal and Slot Editor'. The 'Output' panel at the bottom shows the following log entries:

```
untitled
C:\Qt\MyProjects\lek1\build-untitled-Desktop_Qt_5_3_MinGW_32bit-Debug\debug\untitled.exe завершился с кодом 0

Запускается C:\Qt\MyProjects\lek1\build-untitled-Desktop_Qt_5_3_MinGW_32bit-Debug\debug\untitled.exe...
C:\Qt\MyProjects\lek1\build-untitled-Desktop_Qt_5_3_MinGW_32bit-Debug\debug\untitled.exe завершился с кодом 0

Запускается C:\Qt\MyProjects\lek1\build-untitled-Desktop_Qt_5_3_MinGW_32bit-Debug\debug\untitled.exe...
C:\Qt\MyProjects\lek1\build-untitled-Desktop_Qt_5_3_MinGW_32bit-Debug\debug\untitled.exe завершился с кодом 0

Запускается C:\Qt\MyProjects\lek1\build-untitled-Desktop_Qt_5_3_MinGW_32bit-Debug\debug\untitled.exe...
C:\Qt\MyProjects\lek1\build-untitled-Desktop_Qt_5_3_MinGW_32bit-Debug\debug\untitled.exe завершился с кодом 0
```

The bottom status bar includes a search field and a tabbed interface with the following tabs: 1 Проблемы, 2 Результаты поиска, 3 Вывод приложения, 4 Консоль сборки, 5 Консоль QML/JS.

Панели вывода. Результат поиска

The image shows a screenshot of an IDE's search results panel. The panel is titled "Результаты поиска" (Search Results) and includes a history dropdown set to "Новый поиск" (New Search). The search area is configured with the following options:

- Область: Все проекты (Scope: All projects)
- Искать: (Search for: empty text box)
- Учитывать регистр (Match case)
- Слова полностью (Match whole words)
- Использовать регулярные выражения (Use regular expressions)
- Шаблон: * (Pattern: *)

Buttons for "Найти" (Find) and "Найти и заменить" (Find and replace) are visible at the bottom right of the search area. The background shows a code editor with lines 11 and 12, and a file explorer on the left with open documents: main.cpp*, mainwindow.cpp, mainwindow.h, and mainwindow.ui. The bottom status bar shows a search icon and the text "Быстрый поиск (Ctrl+K)", along with a tab bar containing: 1 Проблемы (4), 2 Результаты поиска, 3 Вывод приложения, 4 Консоль сборки, and 5 Консоль QML/JS.

Панели вывода. Консоль сборки

The screenshot displays the Qt Creator IDE interface. The top-left pane shows a code editor with lines 11 and 12. The bottom-left pane lists open documents: main.cpp*, mainwindow.cpp, mainwindow.h, and mainwindow.ui. The bottom-right pane, titled 'Консоль сборки' (Build Console), shows the following output:

```
.. \untitled\mainwindow.cpp:10:1: error: stray '\270' in program
.. \untitled\mainwindow.cpp:10:1: error: stray '\320' in program
.. \untitled\mainwindow.cpp:10:1: error: stray '\260' in program
Makefile.Debug:347: recipe for target 'debug/mainwindow.o' failed
mingw32-make[1]: *** [debug/mainwindow.o] Error 1
mingw32-make[1]: Leaving directory 'C:/Qt/MyProjects/lekc1/build-untitled-Desktop_Qt_5_3_MinGW_32bit'
Makefile:34: recipe for target 'debug' failed
mingw32-make: *** [debug] Error 2
21:12:28: Процесс «C:\Qt\Tools\mingw482_32\bin\mingw32-make.exe» завершился с кодом 2.
Ошибка при сборке/установке проекта untitled (комплект: Desktop Qt 5.3 MinGW 32bit)
Во время выполнения этапа «Сборка»
21:12:28: Прошло времени: 00:04.
```

The bottom status bar includes a search field with the text 'Быстрый поиск (Ctrl+K)' and a tabbed interface with the following tabs: 1 Проблемы (4), 2 Результаты поиска, 3 Вывод приложения, 4 Консоль сборки, and 5 Консоль QML/JS.

Режим дизайна

The screenshot shows the Qt IDE interface in Design Mode. The top menu bar includes: Файл, Правка, Сборка, Отладка, Анализ, Инструменты, Окно, Справка. The toolbar shows icons for file operations and a dropdown menu for the current file, 'mainwindow.cpp', with '<Нет символов>' (No symbols).

The left sidebar contains navigation icons and labels: 'Qt' logo, 'Начало' (Start), 'Редактор' (Editor), 'Дизайн' (Design), 'Отладка' (Debug), and 'Проекты' (Projects).

The central pane displays a project tree for 'untitled' with the following structure:

- untitled (Qt icon)
 - untitled.pro (Qt icon)
 - Заголовочные (h icon)
 - Исходники (C++ icon)
 - main.cpp (C++ icon)
 - mainwindow.cpp (C++ icon)
 - Формы (Form icon)
 - mainwindow.ui (Form icon, selected)

The right pane shows the source code for 'mainwindow.cpp' with line numbers 1 through 15:

```
1 #include "mainwindow.h"
2 #include "ui_mainwindow.h"
3
4 MainWindow::MainWindow(QWidget *parent) :
5     QMainWindow(parent),
6     ui(new Ui::MainWindow)
7 {
8     ui->setupUi(this);
9 }
10
11 MainWindow::~MainWindow()
12 {
13     delete ui;
14 }
15
```

A tooltip is visible over the 'mainwindow.ui' file in the project tree, displaying the path: C:\Qt\MyProjects\lekc1\untitled\mainwindow.ui.

Режим дизайна

mainwindow.ui - untitled - Qt Creator

Файл Правка Сборка Отладка Анализ Инструменты Окно Справка

mainwindow.ui

Фильтр

- Layouts
 - Vertical Layout
 - Horizontal Layout
 - Grid Layout
 - Form Layout
- Spacers
 - Horizontal Spacer
 - Vertical Spacer
- Buttons
 - Push Button
 - Tool Button
 - Radio Button
 - Check Box
 - Command Link Button
 - Button Box
- Item Views (Model-Based)
 - List View
 - Tree View
 - Table View
 - Column View
- Item Widgets (Item-Based)
 - List Widget
 - Tree Widget
 - Table Widget
- Containers
 - Group Box
 - Scroll Area

Пишите здесь

Объект	Класс
MainWindow	QMainWindow
centralWidget	QWidget
menuBar	QMenuBar
mainToolBar	QToolBar
statusBar	QStatusBar

Фильтр

MainWindow : QMainWindow

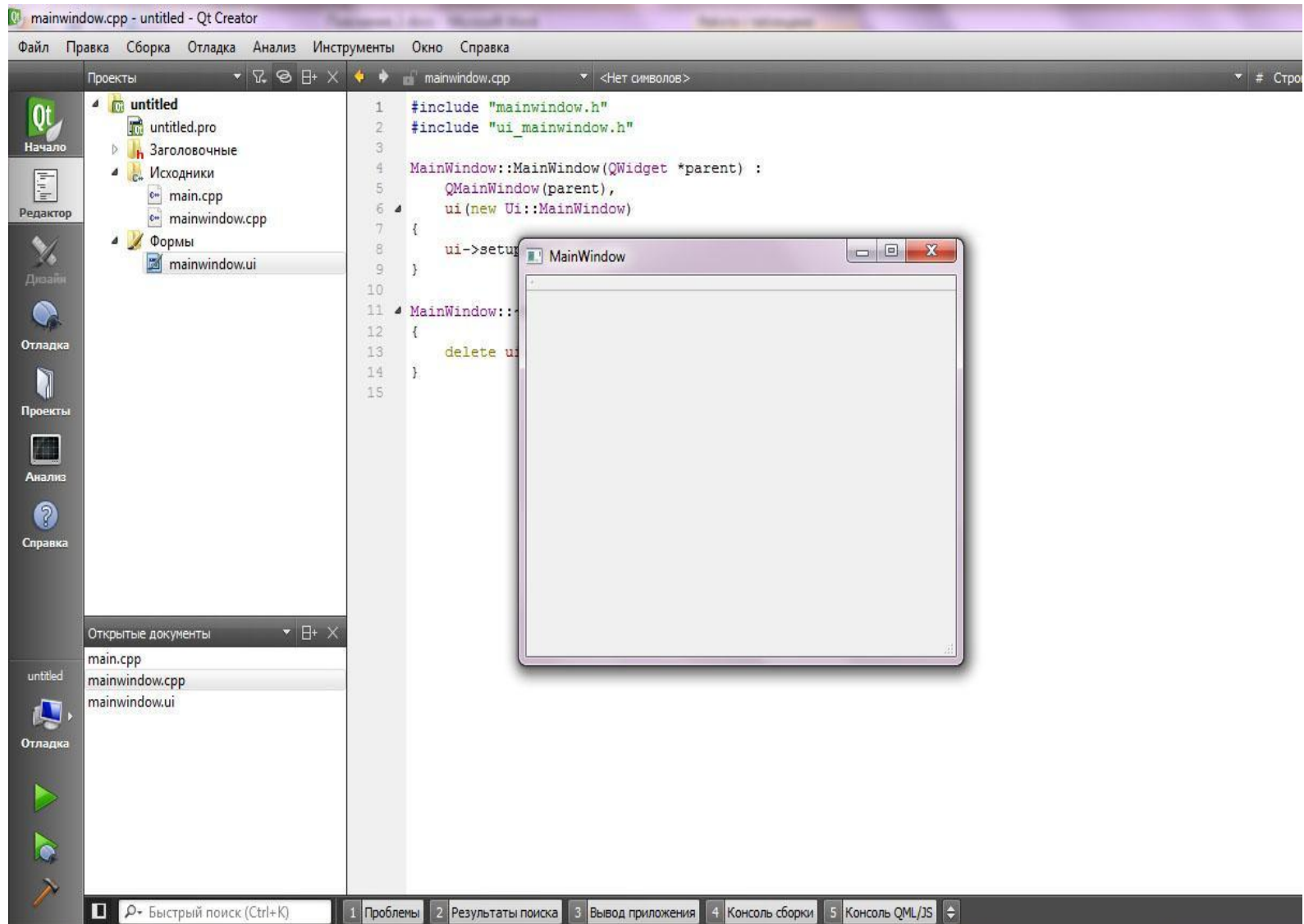
Свойство	Значение
QObject	
objectName	MainWindow
QWidget	
windowModality	NonModal
enabled	<input checked="" type="checkbox"/>
geometry	
X	0
Y	0
Ширина	400
Высота	300
sizePolicy	
Горизонтальная ...	Preferred
Вертикальная по...	Preferred
Горизонтальное ...	0
Вертикальное ра...	0
minimumSize	0 x 0
maximumSize	16777215 x 16777215
sizeIncrement	0 x 0
baseSize	0 x 0
palette	Унаследован
font	A [MS Sh
cursor	Arrow
mouseTracking	<input type="checkbox"/>

Имя Используется Текст Горячая клавиша Триггерное Подсказка

Редактор действий Редактор сигналов и слотов

Быстрый поиск (Ctrl+K) 1 Проблемы 2 Результаты поиска 3 Вывод приложения 4 Консоль сборки 5 Консоль QML/JS

Главное окно



main.cpp

```
1  #include "mainwindow.h"
2  #include <QApplication>
3
4  int main(int argc, char *argv[])
5  {
6      QApplication a(argc, argv);
7      MainWindow w;
8      w.show();
9
10     return a.exec();
11 }
12 |
```

mainwindow.cpp

```
<  mainwindow.cpp  <Выберите символ>
1  #include "mainwindow.h"
2  #include "ui_mainwindow.h"
3
4  MainWindow::MainWindow(QWidget *parent) :
5      QMainWindow(parent),
6      ui(new Ui::MainWindow)
7  {
8      ui->setupUi(this);
9  }
10
11  MainWindow::~MainWindow()
12  {
13      delete ui;
14  }
```

Заголовок окна

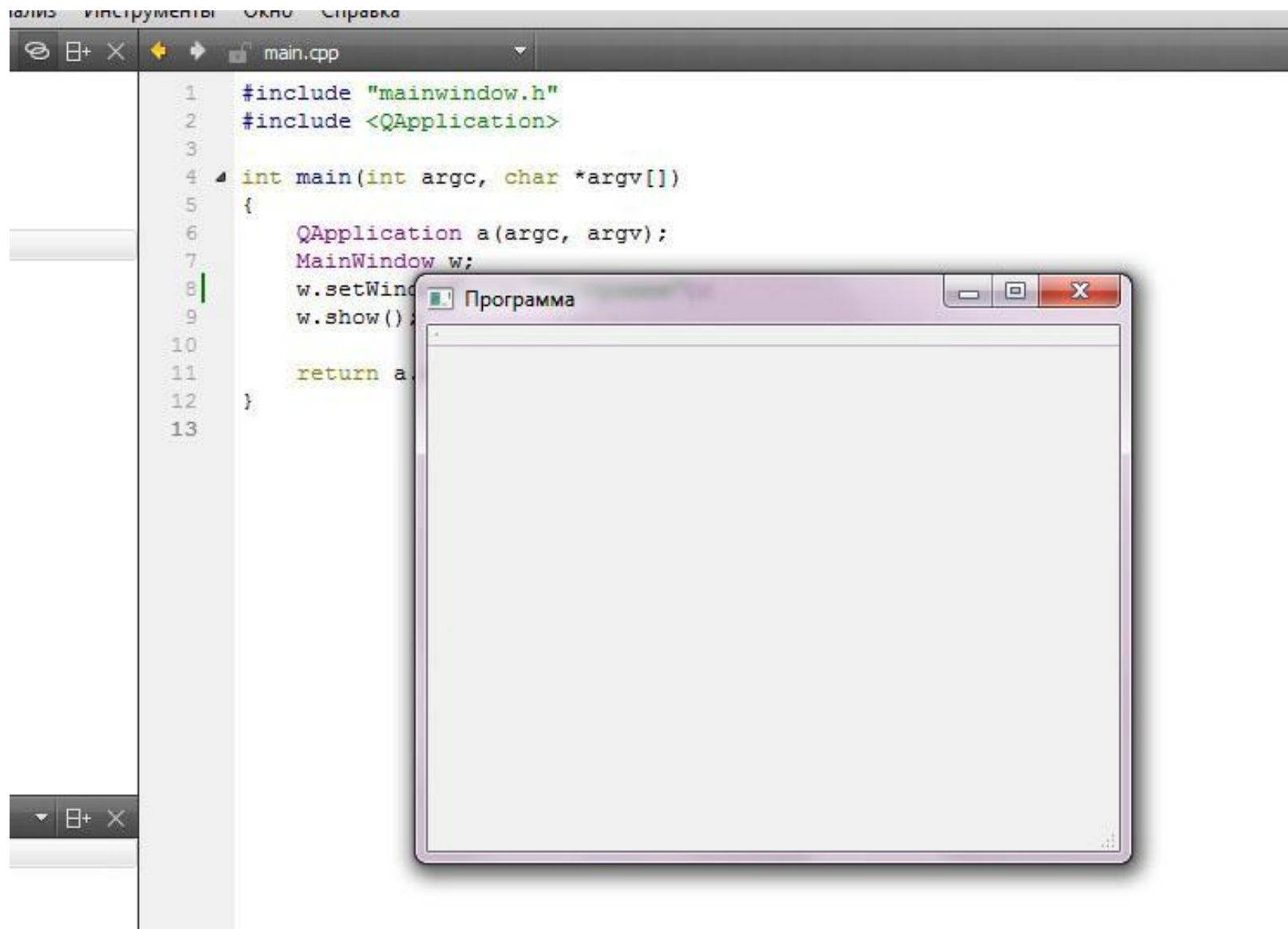
```
1  #include "mainwindow.h"
2  #include <QApplication>
3
4  int main(int argc, char *argv[])
5  {
6      QApplication a(argc, argv);
7      MainWindow w;
8      w.setWindowTitle
9      
10
11     return a.exec();
12 }
13
```

Заголовок окна

```
1  #include "mainwindow.h"
2  #include <QApplication>
3
4  int main(int argc, char *argv[])
5  {
6      QApplication a(argc, argv);
7      MainWindow w; void setTitle(const QString &)
8      w.setWindowTitle();
9      w.show();
10
11     return a.exec();
12 }
13
```

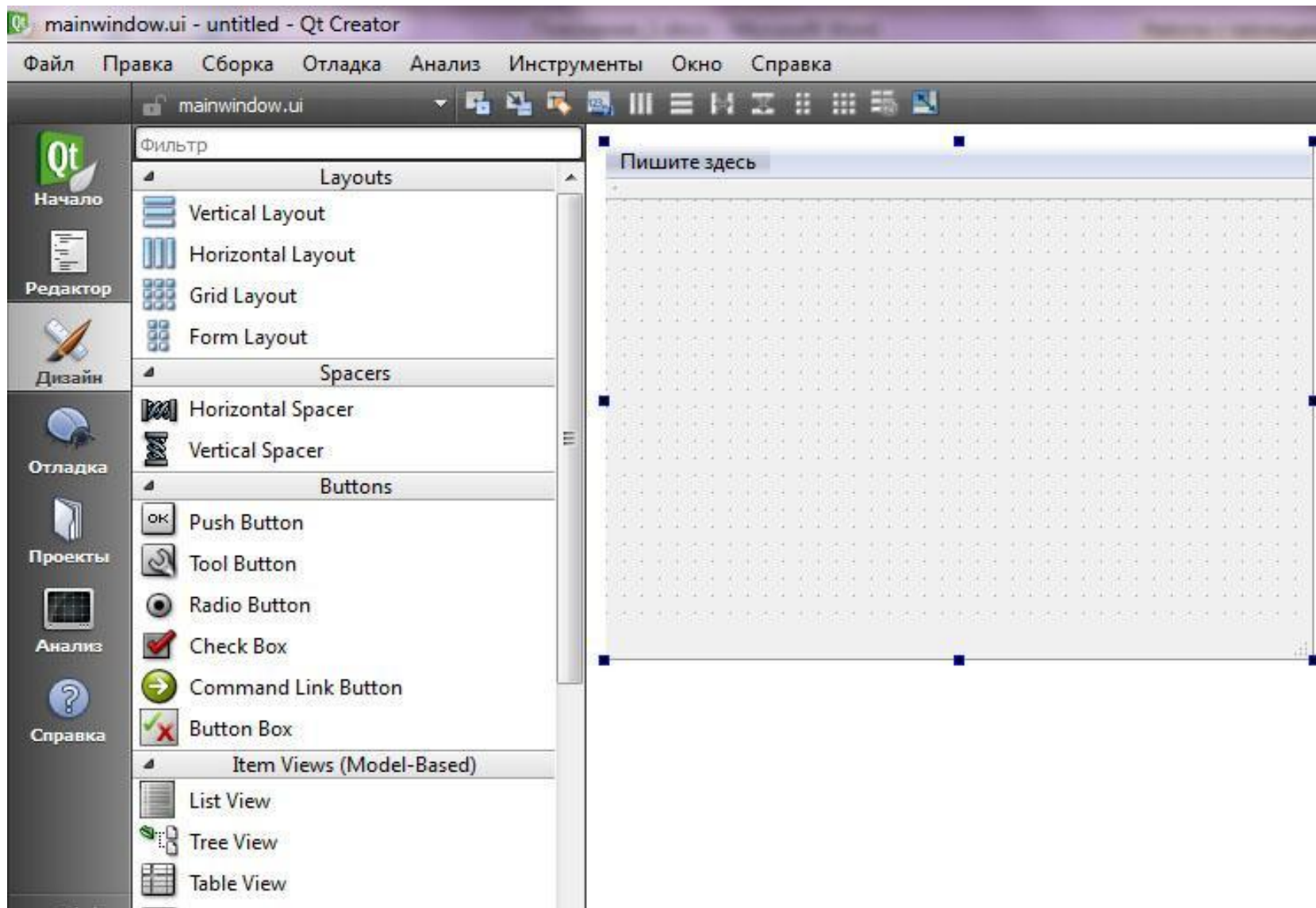
Заголовок окна

```
Файлы  Инструменты  Окно  Справка
main.cpp
1  #include "mainwindow.h"
2  #include <QApplication>
3
4  int main(int argc, char *argv[])
5  {
6      QApplication a(argc, argv);
7      MainWindow w;
8      w.setWindowTitle("Программа");
9      w.show();
10
11     return a.exec();
12 }
13
```



The image shows a Qt IDE interface. The top menu bar includes 'Файлы', 'Инструменты', 'Окно', and 'Справка'. Below it is a toolbar with icons for search, zoom, and navigation. The main editor area displays the source code for 'main.cpp'. The code includes 'mainwindow.h' and 'QApplication', and defines a 'main' function that creates a 'QApplication' object, a 'MainWindow' object, sets the window title to 'Программа', and shows the window. A floating window titled 'Программа' is visible in the foreground, showing a blank white area, which is the result of the code execution.

Заголовок окна



Типы данных

- **qint8** – целое со знаком (8 битов), аналог типа signed char;
- **quint8** и **uchar** – целое неотрицательное (8 битов, unsigned char);
- **qint16** – целое со знаком (16 бит, short);
- **quint16** и **ushort** – целое неотрицательное (16 бит, unsigned short);
- **qint32** – целое со знаком (32 бита, int);
- **quint32** и **uint** – целое неотрицательное (32 бита, unsigned int);
- **qint64** – целое со знаком (64 бита, long);
- **quint64** и **ulong** – целое неотрицательное (64 бита, unsigned long);
- **qlonglong** – эквивалент quint64;
- **qulonglong** – эквивалент quint64;
- **qreal** – вещественное число, аналог double, за исключением платформ с ARM архитектурой процессоров, в этом случае тип qreal определен как float.

Массивы и списки

- ❑ вектор **QVector<T>**,
 - ❑ список **QList<T>**,
 - ❑ двусвязный список **QLinkedList<T>**
 - ❑ низкоуровневый класс для работы с массивами переменной длины **QVarLengthArray<T>**.
-
- ❑ Для работы со списком строк имеется специальный
 - ❑ класс **QStringList**.

Общий алгоритм работы со списком QList

1. Объявить объект – список элементов нужного типа, например:

```
QList <int> intList; //список целых чисел  
QList <QDate> dateList; //список дат  
QList <QString> strList; //список строк
```

Общий алгоритм работы со списком QList

2. Заполнить список значениями. Добавить элемент в конец списка можно методом **append()** или оператором **<<**

```
intList<<1<<10<<-20;  
dateList.append(QDate(2008, 03, 31));  
strList.append(tr("Строка"));
```

Для добавления элемента в начало списка существует метод **prepend()**;

Общий алгоритм работы со списком QList

3. Выполнить необходимые операции над списком, например

```
//заменить существующее значение  
strList[3] = "Другая строка";  
  
// более быстрый вариант  
strList.at(3) = "Другая строка";  
  
intList.removeAt(1); // удалили элемент  
  
// поменяли элементы местами  
dateList.swap(0, 2);  
strList.clear(); // очистили список
```

Пример работы со списком QList

```
1  #include <QStringList>
2  #include <QVector>
3  #include <QtDebug>
4  using namespace std;
5  int main() {
6  QStringList list;
7  list<<"abc"<<"ab"<<"cde";
8
9  QVector<QString> vect(3);
10 vect[0] = "abc";
11 vect[1] = "ab";
12 vect[2] = "cde";
13
```

Пример работы со списком QList

```
14 bool x = qEqual(list.begin(),
15                 list.end(),
16                 vect.begin());
17 qDebug() << list << endl << vect << endl << x; // true
18 qDebug() << "                " << endl;
19 list.replace(0, "lala"); // или list[0] =
   "lala"
20 x = qEqual(list.begin(),
21            list.end(),
22            vect.begin());
23 qDebug() << list << endl << vect << endl << x; // false
24
25 getchar();
26 return 0;
27
28 }
```