Qt

Введение
Лекция 2

Кафедра ИВТ и ПМ

2017

Прошлые темы

- Что такое фреимворк?
- Для чего код организуют именно в фреимворки?
- Назовите примеры фреимворков и их назначение.
- Охарактеризуйте фреимворк Qt.

Прошлые темы

- Что такое API?
- Что такое сигналы и слоты?
- Для чего они используются?
- Что такое парадигма программирования?
- Что такое событийно-ориентированное программирование?
- Для чего прежназначен классы QCoreApplication и QApplication?

Hello World

Пример простейшего консольного приложения Qt.

файл проекта

```
QT -= gui  # отключим поддержку GUI


# настройка проекта
CONFIG += c++11 console  # поддержка C++11, консольное прило
CONFIG -= app_bundle     # отключение
SOURCES += main.cpp
```

файл исходных кодов

```
#include <QCoreApplication>
#include <iostream>
int main(int argc, char *argv[]){
    QCoreApplication a(argc, argv);
    std::cout << "Hello world\n";
    return a.exec();
}
```

Простое приложение с GUI

Файл проекта

```
# Простое приложение с GUI
TARGET = qtlab
TEMPLATE = app
QT += core gui widgets
CONFIG += qt
SOURCES += helloworld.cpp
```

helloworld.cpp

```
#include <QtWidgets/QApplication>
#include <QtWidgets/QLabel>
int main(int argc, char *argv[]){
    QApplication app(argc, argv);
    // Динамическое создание интерфейса пользователя
        QLabel label("Hello World!");
    label.show();
        return app.exec();}
```

Undefined reference

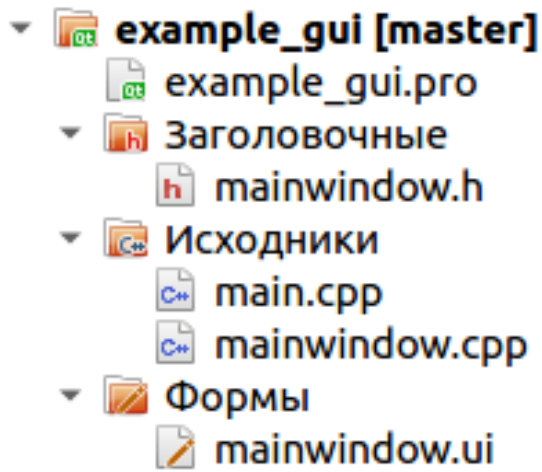
Кроме заголовочных файлов, к проекту нужно подключать соответствующие модули Qt. Если этого не сделать, то

компилятор укажет на то, что используемые имена не определены.

```
In function `main':
/home/sotona/w/lab/cpp/qtlab/build-qtlab-Desktop_Qt_5_9_1_GCC_64bit-Debug/helloworld.o
undefined reference to `QApplication::QApplication(int&, char**, int)'
undefined reference to `QLabel::QLabel(QString const&, QWidget*, QFlags<Qt::WindowType>)'
undefined reference to `QWidget::show()'
undefined reference to `QApplication::exec()'
undefined reference to `QLabel::~QLabel()'
undefined reference to `QApplication::~QApplication()'
```

Построение интерфейса с помощью редактора форм

При создании Qt приложения с GUI (Новый проект -> Приложение Qt Widgets) автоматически создаётся несколько файлов:



- example_gui.pro - файл проекта
- main.cpp - содержит функцию main
- mainwindow.ui - XML файл описывающий окно программы
- mainwindow.cpp, mainwindow.h - класс "главное окно программы"

При создании относительно простых программ с одним главным окном задача разработчика сводится к реализации класса главного окна программы.

main.cpp

```
#include "mainwindow.h"
#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    MainWindow w;
    w.show();

    return a.exec();
}
```

mainwindow.h

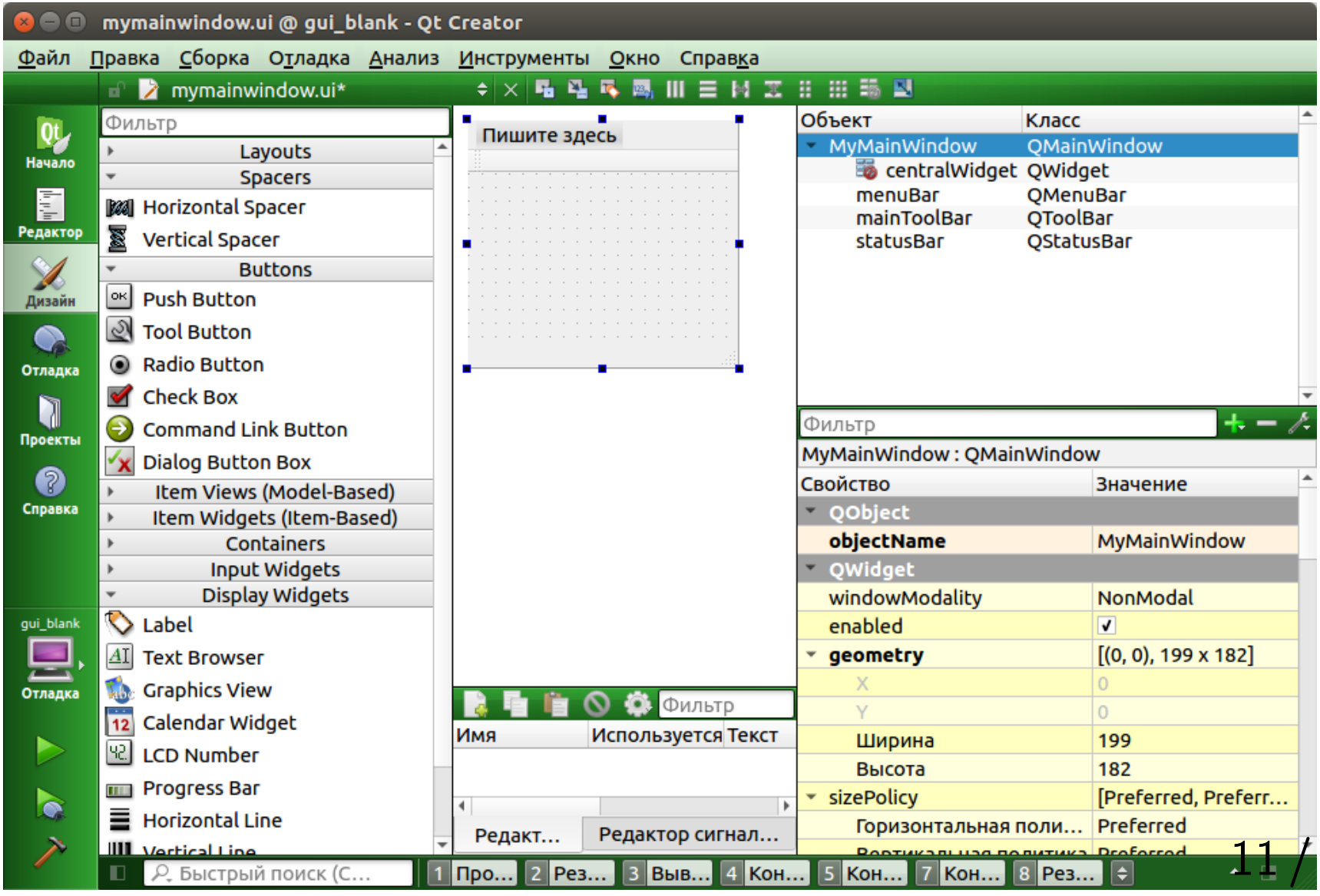
```
#include <QMainWindow>
namespace Ui {class MainWindow;}

class MainWindow : public QMainWindow{
    Q_OBJECT // макрос для создание метаобъекта
public:
    explicit MainWindow(QWidget *parent = 0);
    ~MainWindow();
private:
    // Класс Ui::MainWindow генерируется автоматически из фа
    // в нём описаны все элементы интерфейса, их расположени
    пользователя mainwindow.ui
    Ui::MainWindow *ui;

// другие методы и поля ...
};
```

Диаграмма классов

Как выглядит диаграмма классов этого приложения?

Создание интерфейса пользователя

Файл *.ui - текстовый XML файл, но в QtCreator автоматически открвается в дизайнере форм.



Создание интерфейса пользователя

Элементы инфтерфейса пользователя в Qt называются **виджетами** (widgets). Они представляют собой классы, которые имеют графическое представление (например в виде кнопки).

В самом начале форма - окно программы, уже включает в себя несколько элементов интерфйеса пользователя.

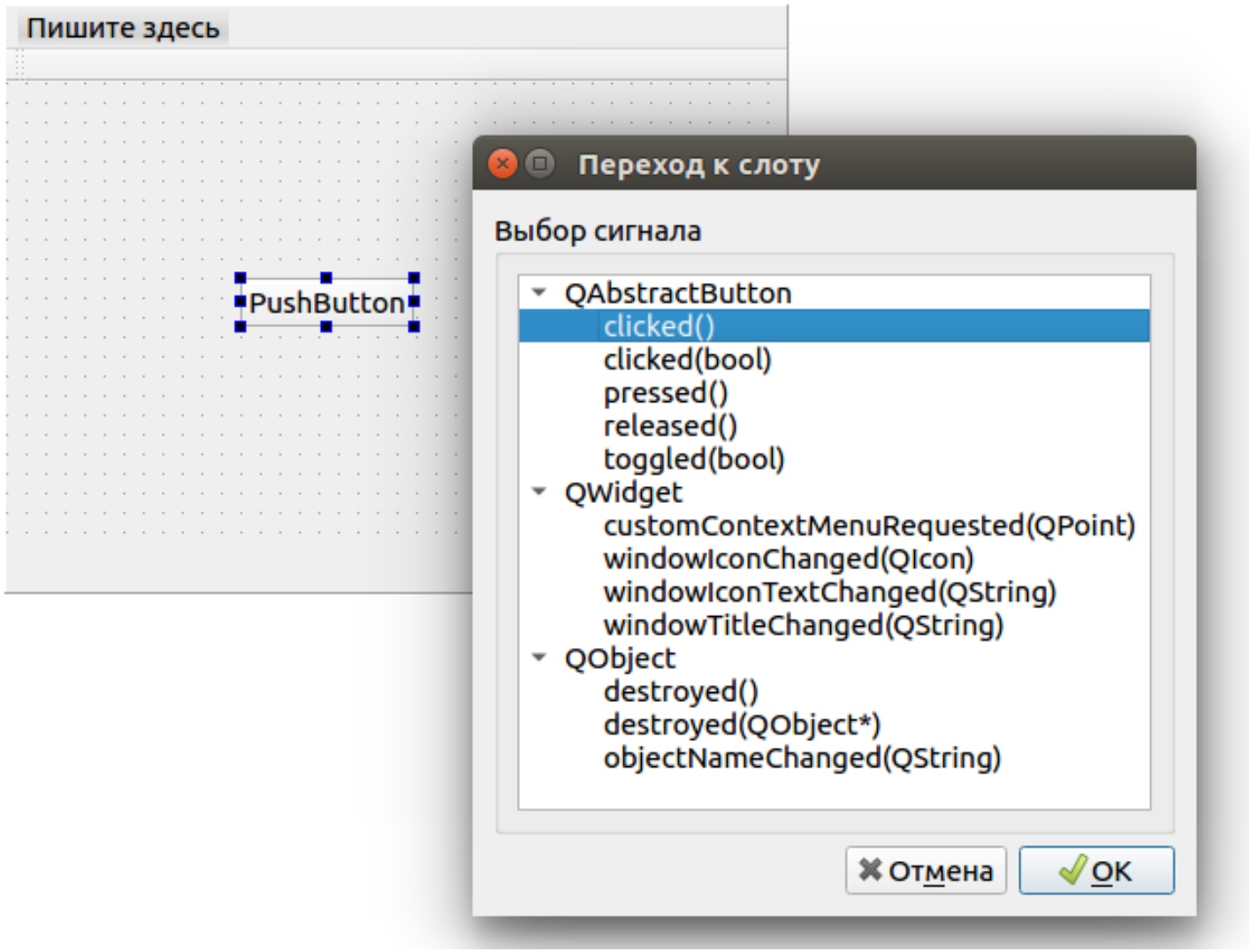
- menuBar - строка меню;
- mainToolBar - панель инструментов;
- statusBar - панель состояния;
- CentralWidget - основной виджет (пустое пространство на форме), который будет содержать остальные элементы интерфейса.

Все эти элементы интерфейса в шаблоне пусты.

Создание обработчиков событий - слотов

- Обработчики событий в фреймворке Qt называются слотами.
- Каждый элемент интерфейса может реагировать на свои наборы событий. Они приведены в справке.
- Обработчики событий - это методы класса "главное окно".
- Эти методы создаются в редакторе форм, и автоматически добавляются в класс. "Перейти к слоту"в контекстном меню элемента GUI
  Разработчику остаётся описать реакцию на событие внутри метода.
- Название методов создаётся их названия элемента GUI. Поэтому рекомендуется давать объектам - элементам интерфейса осмысленные имена, говорящие о их назначении или совершаемом действии.

Создание обработчиков событий - слотов



Создание обработчиков событий - слотов

mainwindow.h

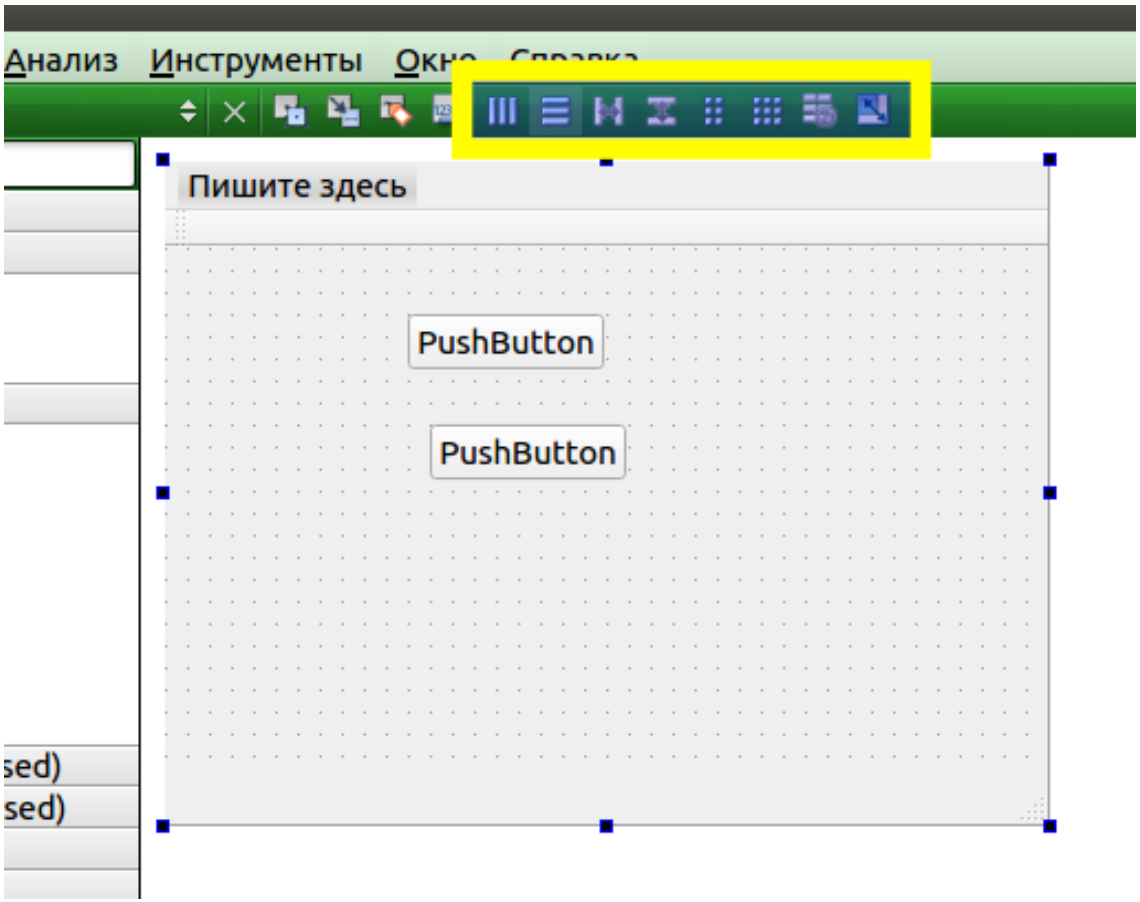
```
...
private slots:
    void on_pushButton_random_clicked();
...
```

mainwindow.cpp

```
...
MyMainWindow::MyMainWindow(QWidget *parent) :
    QMainWindow(parent), ui(new Ui::MyMainWindow){
    // все классы-элементы интерфйеса агрегированы в ui
    ui->setupUi(this);}

void MyMainWindow::on_pushButton_random_clicked(){
    // реакция на нажатие кнопки
}
...
```

Компоновщики виджетов

Менеджеры компоновки (Layouts) - классы управляющие геометрией и расположением других виджетов.



Компоновщики виджетов

Менеджеры компоновки создаются в дизайнере форм. Для подстройки размещения элементов интерфейса их выделяют и выбирают подходящий компоновщик.

Для контроля размера виджета используют настраивают значения полей maximimsize и minuminsize.
Дополнительно для создания "пустопы"вокрыг виджетов применяют "пружины"(Spacers).

Процесс компоновки виджетов идёт поэтапно, снизу вверх: сначала элементы интерфейса объединяются в небольшие группы, затем компонуются сами группы, наконец создаётся компоновщик для всего содержимого главного окна.

Развртывание Qt приложений

Для работы Qt приложения кроме исполняемого файла необходим набор библиотек Qt, так как в исполняемый файл помещается преимущественно код разработчика приложения, а не весь необходимый код фреимворка.

В некоторых дистрибутивах Linux (например Ubuntu) набор библиотек Qt уже установлен в системе и приложения (если версии библиотек совпадают) можно распространять без них.

В ОС Windows часто библиотеки либо не установлены, либо отличаются версией или присутствуют не в полном составе. Поэтому приложение часто распространяют уже с набором dll файлов фреимворка Qt. Для создание такого набора используется программа windeployqt.
Qt for Windows - Deployment Развёртывание приложений Qt в Windows

Развртывание Qt приложений

Qt for Windows - Deployment

Развёртывание приложений Qt в Windows

Ссылки и литература

- Qt Википедия
- OpenSource версия
- Qt wiki

Книги:

- Qt 5.X. Профессиональное программирование на C++. Макс Шлее. 2015 г. 928 с. Книга периодически обновляется с выходом новых версий фреймворка Qt.
- Qt. Профессиональное программирование. Разработка кроссплатформенных приложений на C++. Марк Саммерфилд