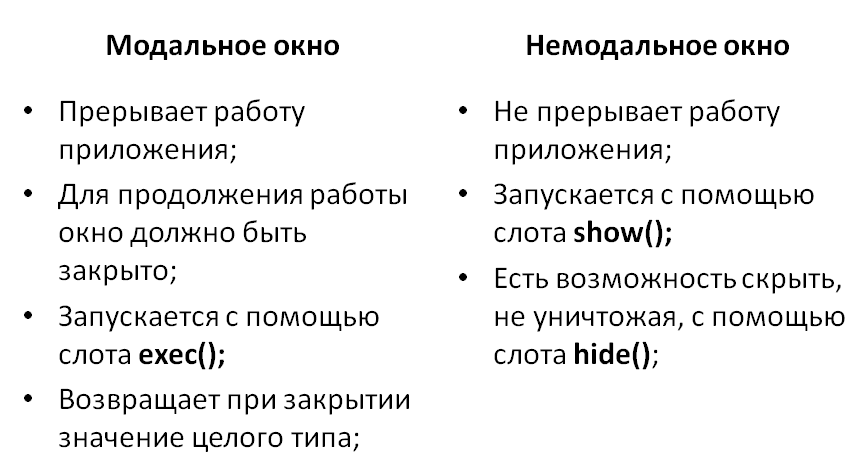
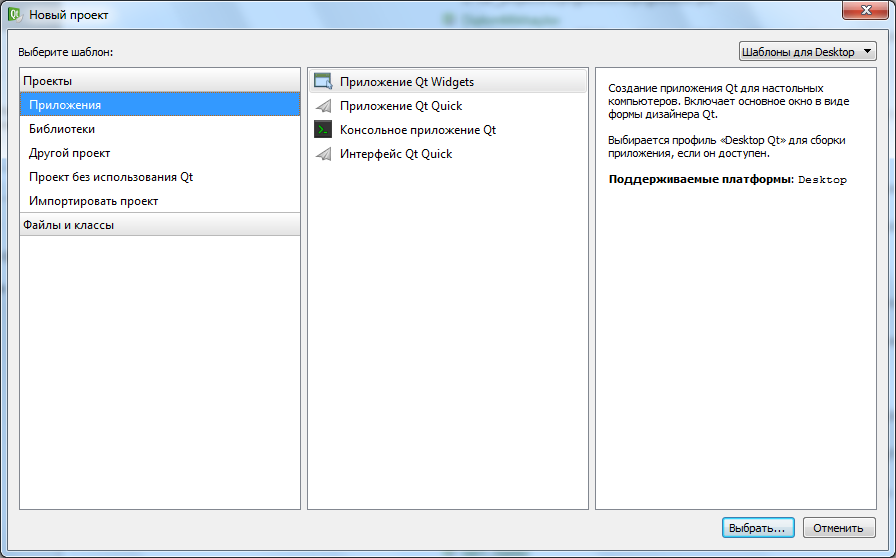
# QDialog, QAction.

## Виды диалоговых окон

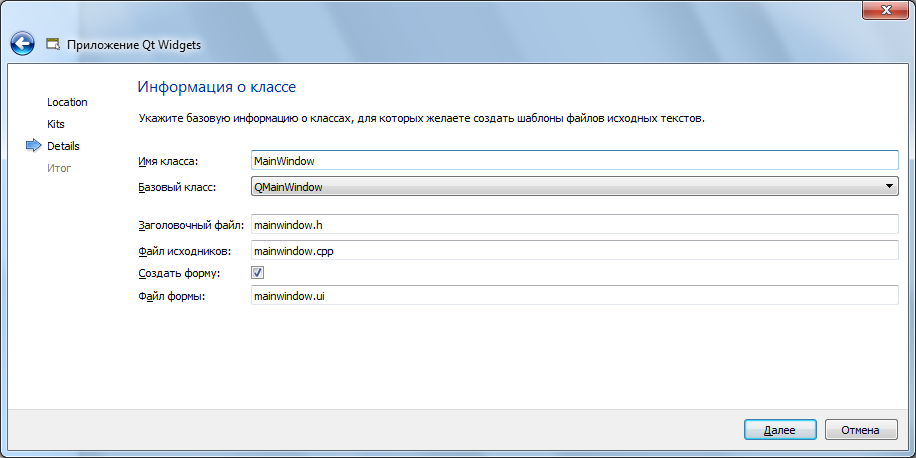


## Создаем проект MyDialog

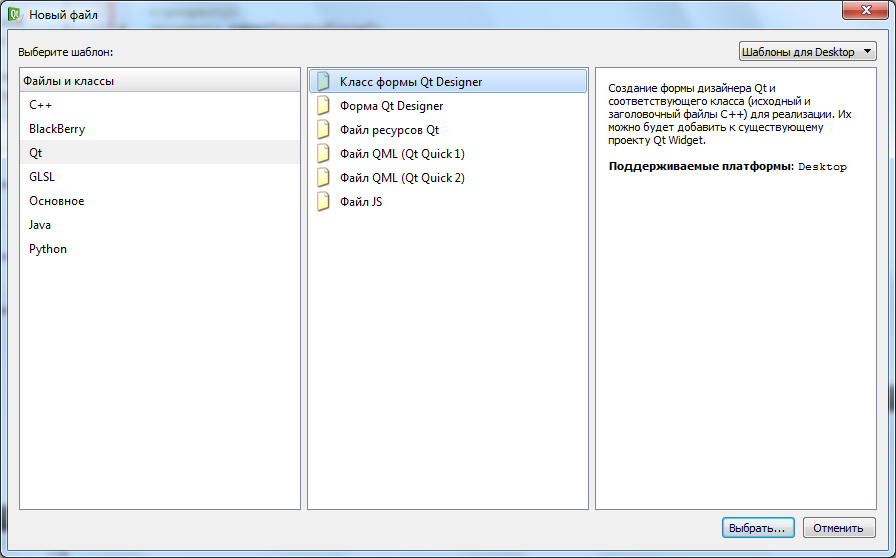
* Создаем проект

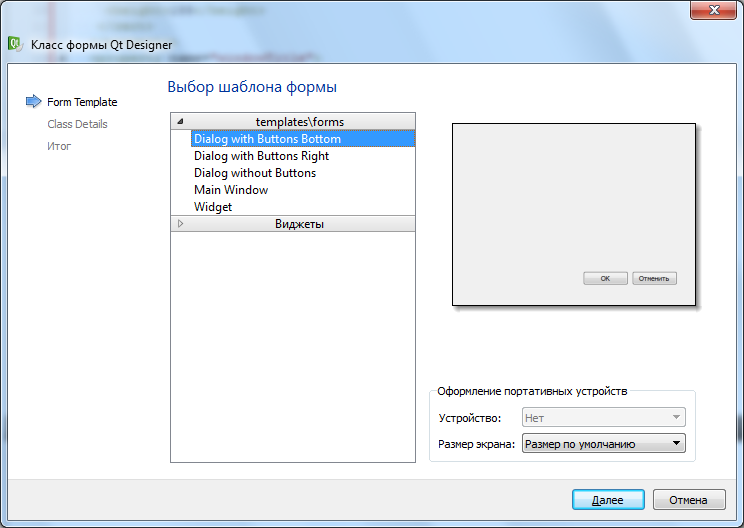


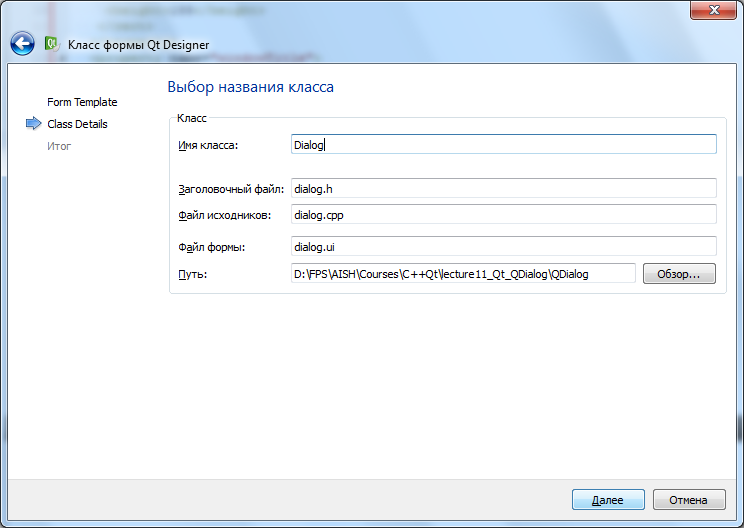
…

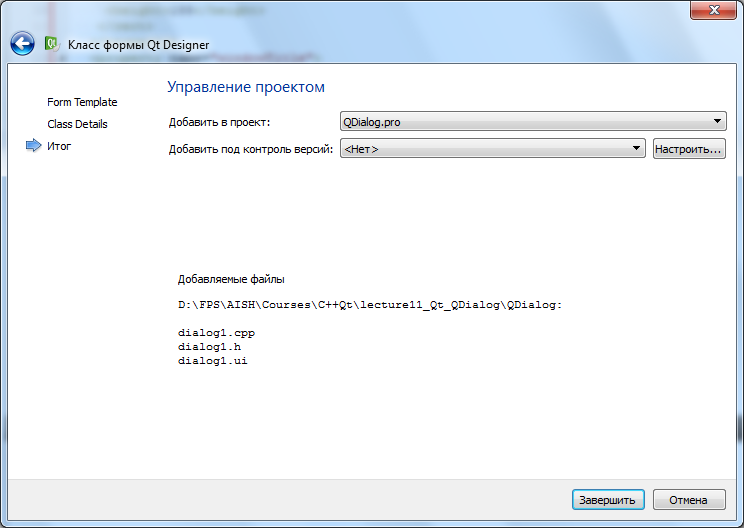


* Заполняем в дизайнере форму mainwindow.ui, как показано на слайде.
* Создаем 2 слота-обработчика событий
  + 1-й – обработчик события нажатия кнопки
  + 2-й – обработчик, который будет обрабатывать информацию, поступающую от нашего диалогового окна (receive(QString))
* Создаем наше диалоговое окно









* Заполняем в дизайнере форму dialog.ui, как показано на слайде.
* Для того, чтобы открыть диалоговое окно и прочитать заданную строку при нажатии кнопки в mainwindow.ui, необходимо заполнить слот-обработчик события нажатия кнопки класса MainWindow.
  1. Создайте объект класса Dialog. Параметром конструктора передайте строку, которую Вы предварительно ввели в QLineEdit формы класса MainWindow.
  2. Вызовите диалоговое окно методом QDialog::exec().
  3. Соответственно, в конструкторе класса Dialog выведите вместо звездочек переданную строку.
* Для того, чтобы передать информацию из диалога в MainWindow выполняем следующие действия:
* 1-й вариант
  + Перегружаем в классе Dialog слот accept();
  + Генерируем в нем сигнал sendString(QString) класса Dialog, передающий строку из QLineEdit;
  + Не забываем в нем вызвать метод close() для закрытия окна.
  + В классе MainWindow связываем sendString(QString) и слот MainWindow receive(QString).
  + **Подумайте**, где именно нужно вызвать метод connect() для связи и слота.
  + В слоте receive(QString) передайте QLabel полученную строку.
* 2-й вариант
  + Создаем в классе Dialog строку m\_str, куда будем записывать данные QLineEdit, и метод, осуществляющий доступ к ней.
  + Соединяем сигнал QLineEdit valueChanged(QString) со слотом класса Dialog, в котором записывает содержимое QLineEdit в строку.
  + В классе MainWindow осуществите вызов диалоговго окна следующим образом:

if ( diag->*exec*() )

{

QString str = diag->getStr();

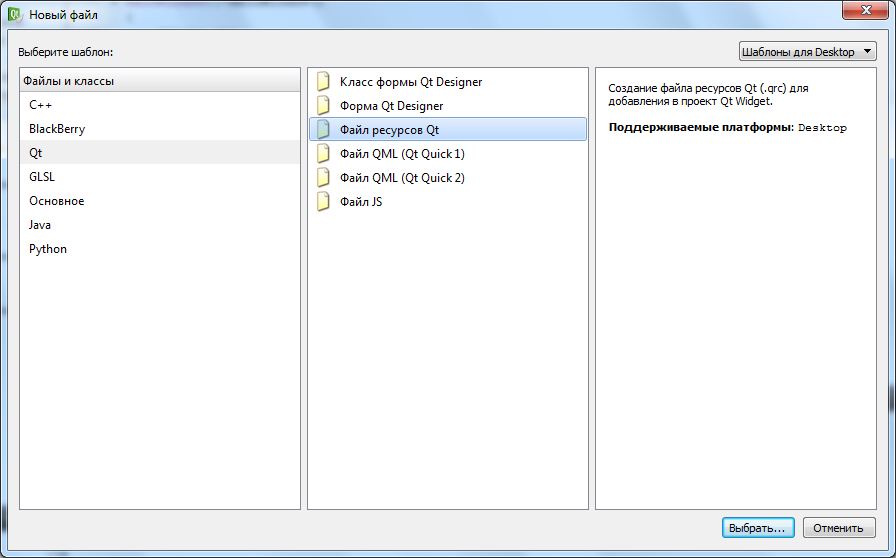
// передайте QLabel полученную строку

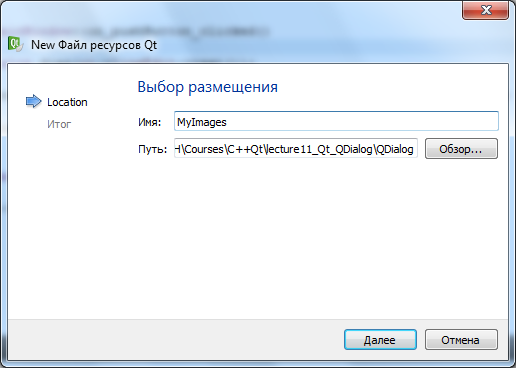
}

## Система ресурсов Qt

### Как использовать редактор ресурсов в QtCreator-е

* Создаем файл .qrc и добавляем его в проект:



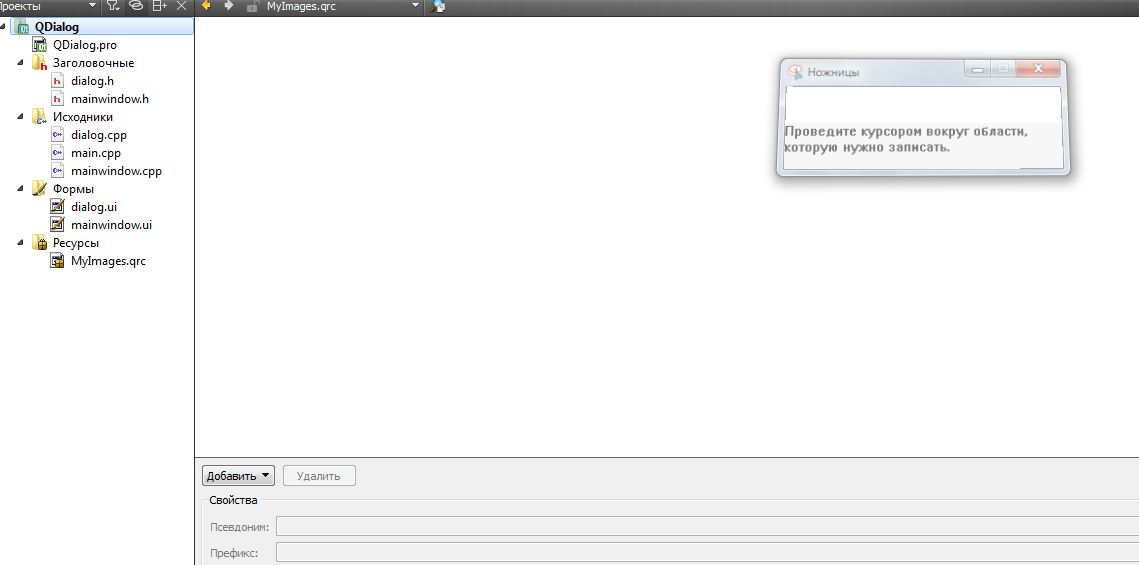


В результате:

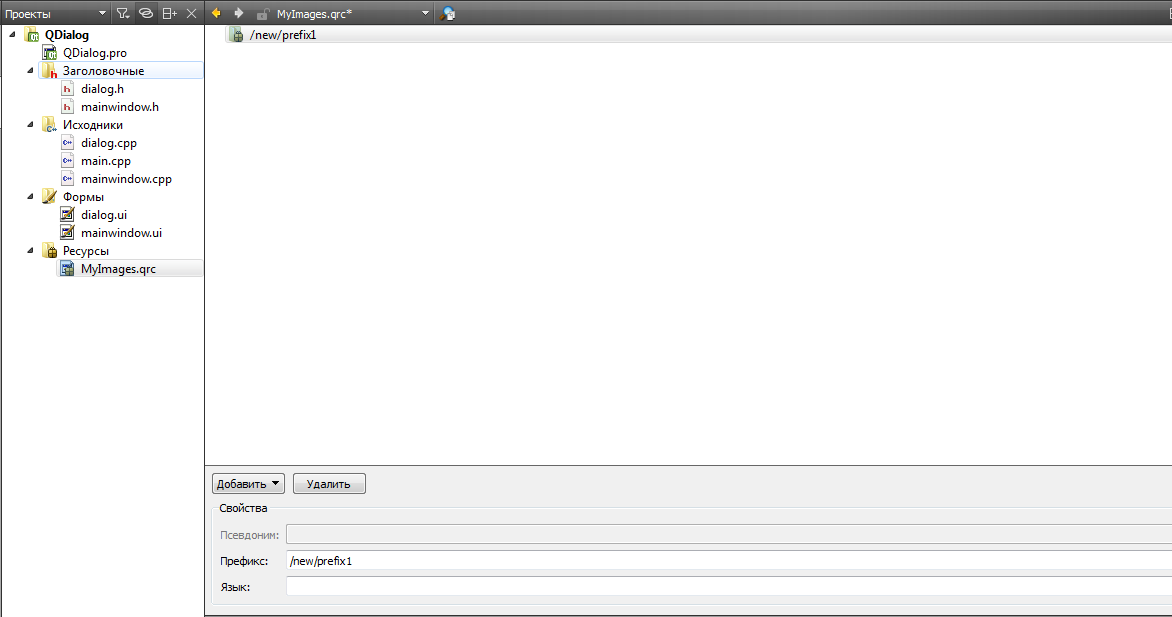
* в окне «Проекты» появится виртуальная папка «Ресурсы»
* будет создан практически пустой файл с заданным именем и расширением .qrc и помещен в виртуальную папку «Ресурсы»

**Важно!**Все, требуемые для создания ресурсов, файлы должны храниться или в той же директории, где создается файл .qrc, или в одном из подкаталогов. Создадим в папке проекта каталог, например, Images и перепишем туда иконку и картинки из моей папки

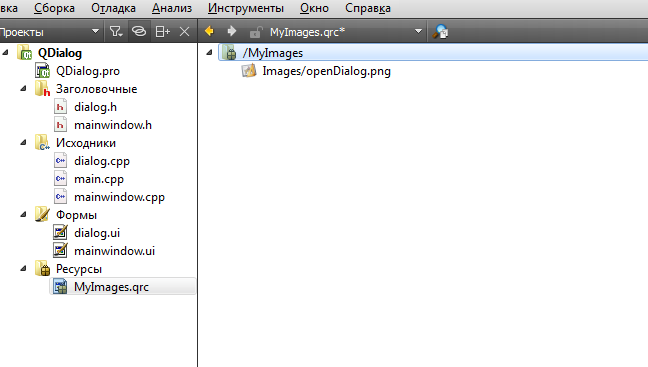
* Для перехода в режим редактирования ресурсов дважды кликните файл\*.qrc или правую кнопку мыши/из контекстного меню –> открыть. Откроется редактор ресурсов



* Нажмите кнопку «Добавить» и выберите «Добавить префикс».



* Замените тот префикс, который предлагает редактор ресурсов **/new/prefix1**, например, на:**/MyImages**
* Нажмите кнопку «Добавить» и выберите «Добавить файл» и выберите требуемый файл (например, иконку).

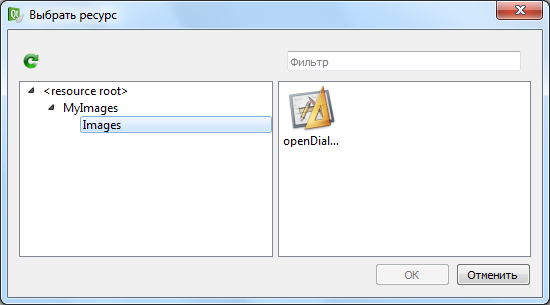


* Пересоберите проект.
* Ресурсы готовы к использованию

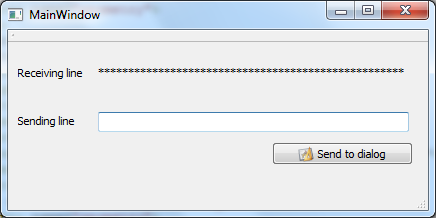
### Картинка на кнопке

Порядок действий:

* открыть форму в режиме дизайнера и выделить кнопку
* в свойствах “icon” выбрать из ресурсов требуемое изображение



* ОК

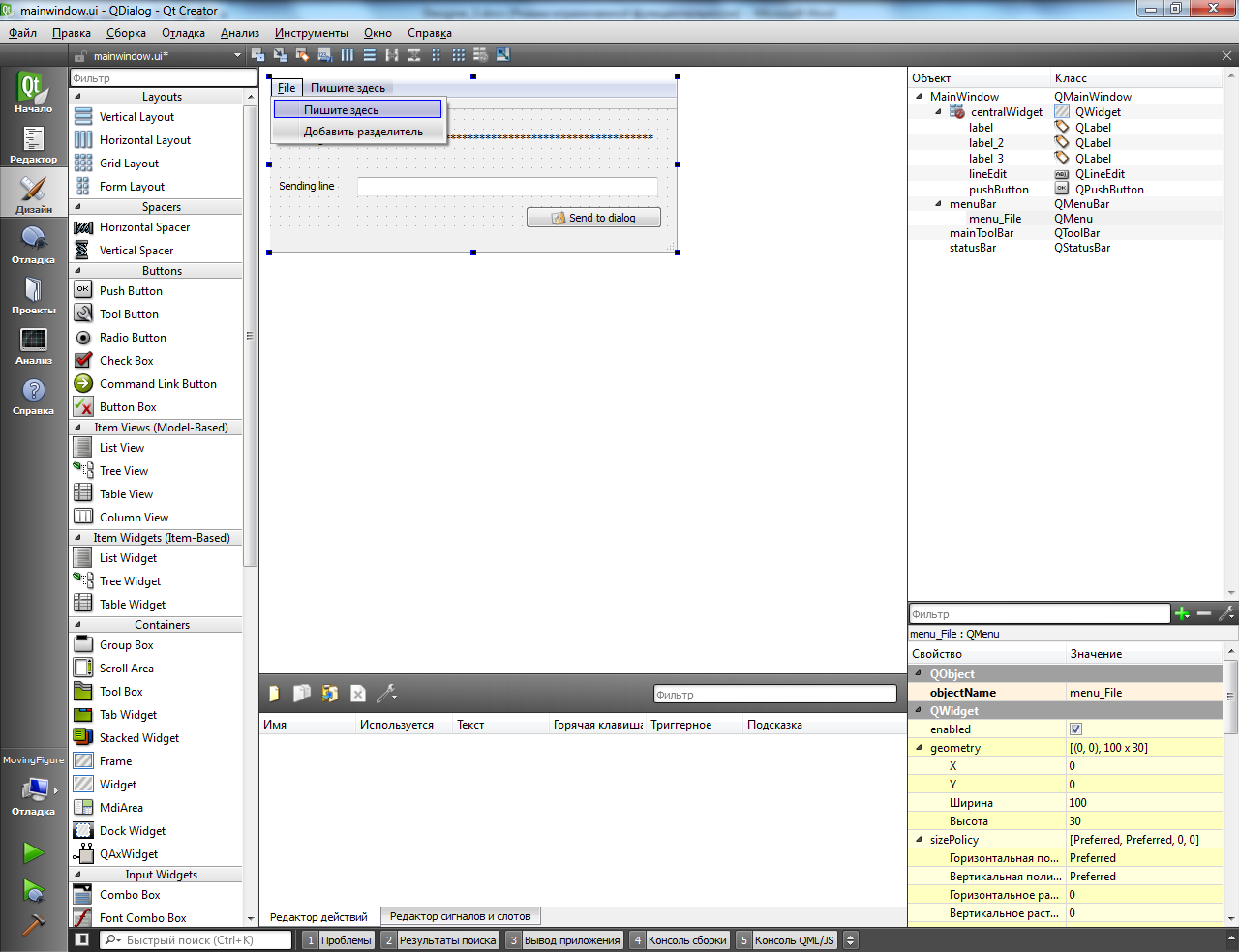


## Меню. QMenuBar, QMenu

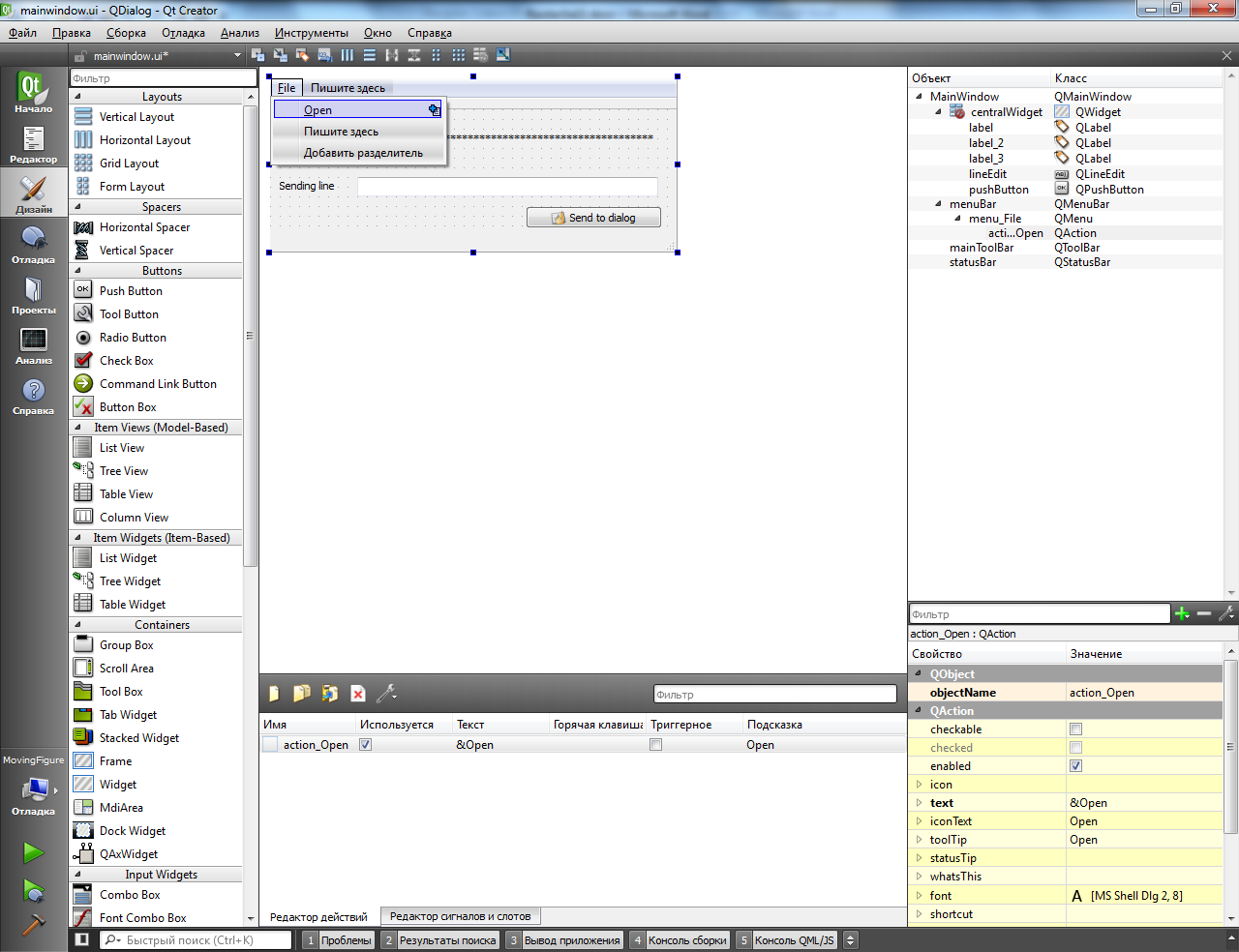
Создадим пункт меню. **При его создании дизайнер автоматически также создает объект-действие**!

Последовательность действий:

* Перейдите в режим дизайнера
* Сначала задаем заголовок выпадающего меню верхнего уровня, например “&File” (там, где «Пишите здесь»)

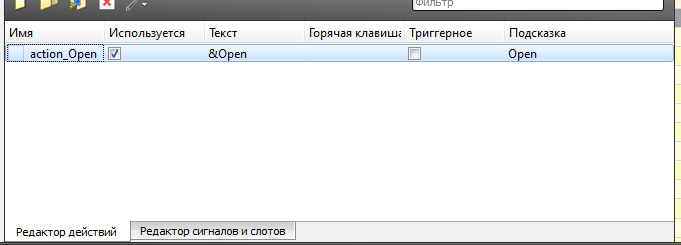


* Затем задаем название пункта меню (Open)

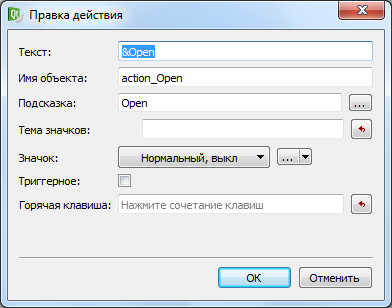


Действием уже можно пользоваться, но у него гораздо больше возможностей, чем то, что задано по умолчанию при создании объекта таким способом =>всегда можно что-то добавить или изменить с помощью редактора действий:

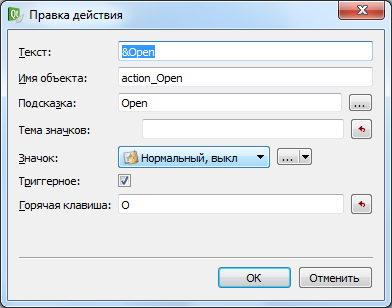
* Сделайте активной закладку «Редактор действий»



* Для редактирования созданного действия double-кликните в соответствующей строке
* Откроется диалоговое окно:



* Настраиваем иконку, выбираем триггерное действие



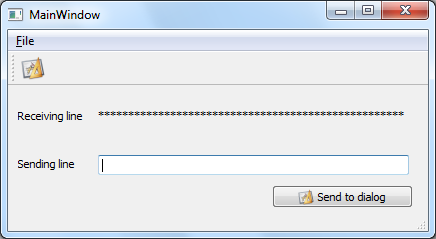
* ОК
* Добавляем connect следующего вида

connect(ui->action\_Open, SIGNAL(triggered()), SLOT(pushButton\_sendToDialog\_clicked()));

## Tool Bar

Добавим наше действие в toolbar. Для этого нужно

* Сделать активной закладку «Редактор действий»
* «Перетащить» существующее действие на строку toolbar-а



В принципе, можно было сначала создать действие, а потом «перетащить» его на строку меню.

## Статусная строка Status Bar

Статусная строка (подсказка внизу главного окна) поддерживается классом QStatusBar.

* + - * Для изменения Temporary части статусной строки (подсказки) в процессе выполнения программы можно использовать слот   
        QStatusBar::showMessage(const QString &*message*, int*timeout* = 0)