



[kju:t]

---

кросс-платформенный открытый инструментарий  
разработки программного обеспечения на языке C++

# История Qt

- Разработку начали в 1991 году Eirik Chambe-Eng и Haavard Nord (Норвегия)
- 1994 - основана компания Troll Tech, позднее Trolltech
- 2001 - выпущена Qt 3.0, появилась поддержка OS X
- 2005 - выпущена Qt 4.0
- Июнь 2008 - Nokia приобрела Trolltech и переименовала его в Qt Development Frameworks
- Март 2011 - Nokia продала коммерческую часть Qt финской компании Digia
- 3 июля 2013 - выход Qt 5.1
- 2014 – “Digia Qt” переименована в “The Qt Company” –дочерняя от “Digia”
- На данный момент последняя версия – Qt 5.11

# Лицензии

- Свободная лицензия LGPL 3.0:
  - Доступен исходный код
  - Исходный код можно модифицировать и распространять модифицированную версию
  - Может быть использована в свободном ПО
  - Может быть использована в коммерческом ПО без обязательного раскрытия исходного кода (кроме случая разработки устройства – с января 2016)
- Коммерческая лицензия
  - Возможность статической линковки
  - Поддержка

# Кросс-платформенный

- Поддерживаются платформы:
  - Windows
  - Unix/X11
  - OS X
  - Встраиваемые и мобильные платформы
    - Embedded Linux
    - Android
    - iOS
    - BlackBerry 10/QNX
    - Экспериментальная поддержка Web Assembly
    - Экспериментальная поддержка в сторонних проектах
      - webOS
      - Amazon Kindle

# Инструментарий / Framework

- QtCreator

- Qt Designer

- Qt

- Qt

- Qt

- Qt

- Qt

- Qt

- Qt

- Qt

- Qt

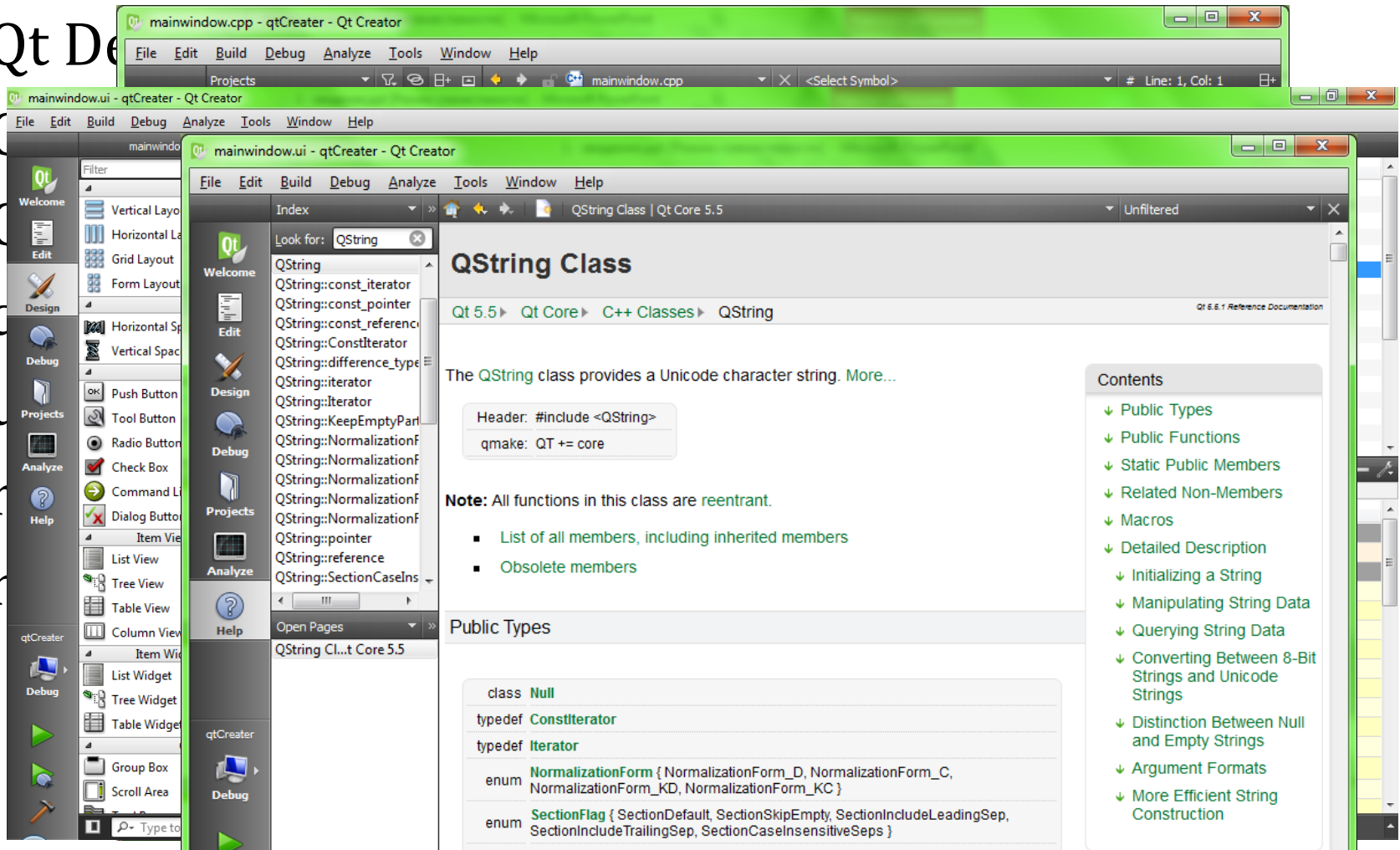
- Qt

- Qt

- Qt

- Qt

- Qt



# Модули

- QtCore — классы ядра библиотеки, используемые другими модулями;
- QtGui — компоненты графического интерфейса;
- QtWidgets — содержит классы для классических приложений на основе виджетов, выделен из QtGui в Qt 5;
- Qt QML — модуль для поддержки QML;
- QtNetwork — набор классов для сетевого программирования. Присутствуют классы для работы с протоколами FTP и HTTP. Для работы с протоколами TCP/IP предназначены такие классы, как QTcpServer, QTcpSocket для TCP и QUdpSocket для UDP;
- QtOpenGL — набор классов для работы с OpenGL;
- QSql — набор классов для работы с базами данных с использованием языка структурированных запросов SQL.
- QtScript — классы для работы с Qt Scripts;
- QtSvg — классы для отображения и работы с данными Scalable Vector Graphics (SVG);
- QtXml — модуль для работы с XML, поддерживается SAX и DOM модели работы;
- QtDesigner — классы создания расширений QtDesigner'a для своих собственных виджетов;
- QtUiTools — классы для обработки в приложении форм Qt Designer;
- Qt3Support — модуль с классами, необходимыми для совместимости с библиотекой Qt версии 3.x.x;
- QTest — классы для поддержки модульного тестирования;
- QtWebKit — модуль WebKit, интегрированный в Qt и доступный через её классы;
- QtXmlPatterns — модуль для поддержки XQuery 1.0 и XPath 2.0;
- Phonon — модуль для поддержки воспроизведения и записи видео и аудио, как локально, так и с устройств и по сети;
- QtCLucene — модуль для поддержки полнотекстового поиска, применяется в новой версии Assistant в Qt 4.4;
- ActiveQt — модуль для работы с ActiveX и COM технологиями для Qt-разработчиков под Windows.
- QtDeclarative — модуль, предоставляющий декларативный фреймворк для создания динамичных, настраиваемых пользовательских интерфейсов.

# Поддержка в других языках программирования

- Ada
- C# & .NET
- D
- Harbour
- Haskell
- Java
- Lisp
- Lua
- Pascal
- Perl
- Python
- R
- Ruby
- Scheme

# Программирование с Qt

---



# Hello, world!

## Консольная версия

```
#include <QCoreApplication>
#include <iostream>

int main(int argc, char *argv[])
{
    QCoreApplication a(argc,
argv);

    std::cout<<"Hello, world!";

    return a.exec();
}
```

## GUI версия

```
#include <QApplication>
#include <QMessageBox>

int main(int argc, char *argv[])
{
    QApplication app(argc, argv);

    QMessageBox::information(NULL,
        "GUI", "Hello, world!");

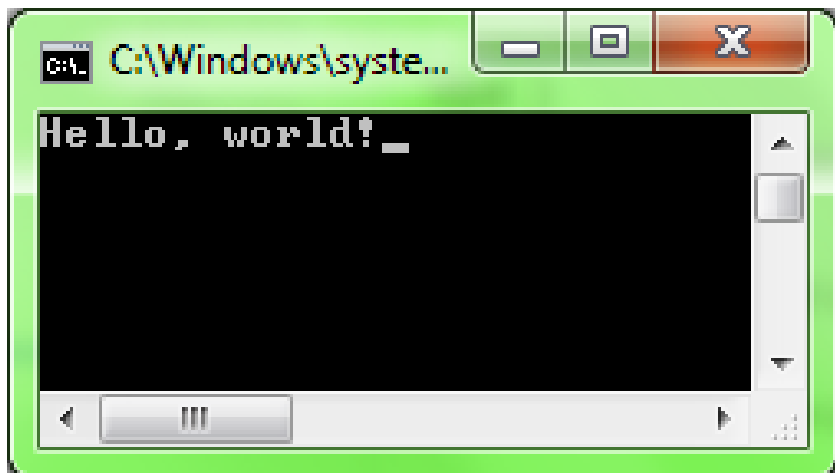
    return app.exec();
}
```

# Hello, world!

## Консольная версия

```
#include <QCoreApplication>
#include <iostream>

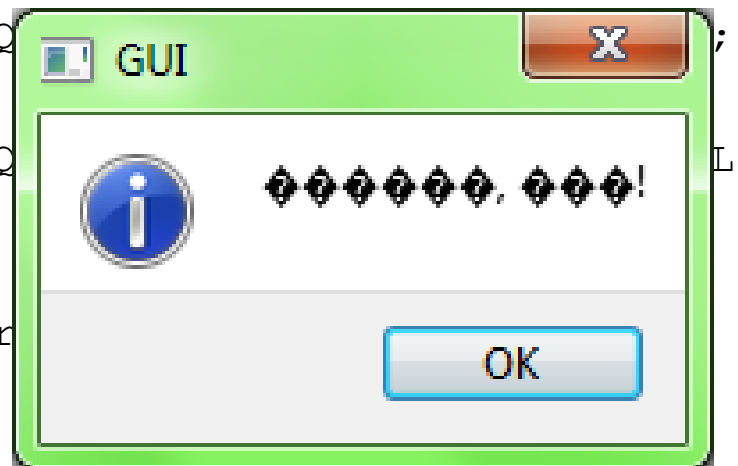
int main(int argc, char *argv[])
{
```



## GUI версия

```
#include <QApplication>
#include <QMessageBox>

int main(int argc, char *argv[])
{
    Q
    Q
    r
}
```

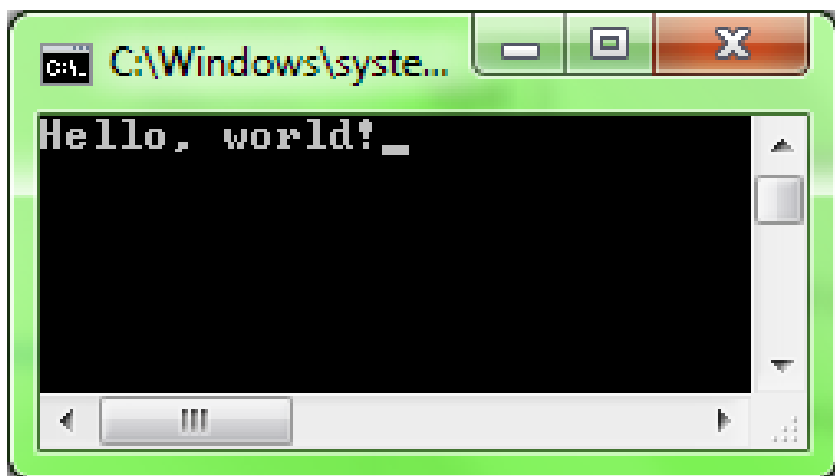


# Hello, world!

## Консольная версия

```
#include <QCoreApplication>
#include <iostream>

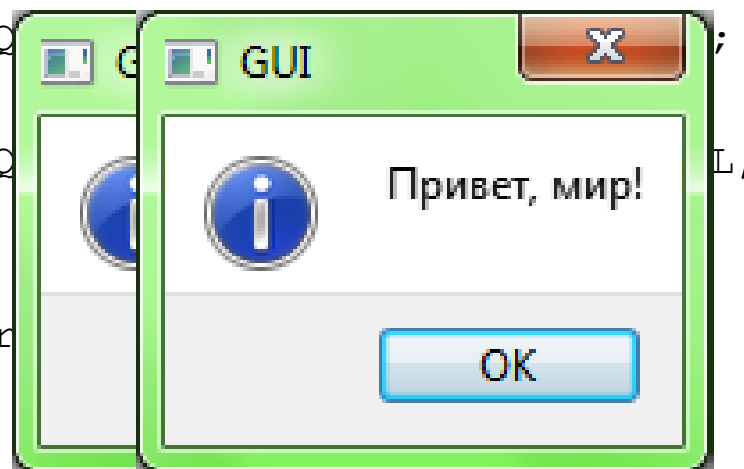
int main(int argc, char *argv[])
{
```



## GUI версия

```
#include <QApplication>
#include <QMessageBox>

int main(int argc, char *argv[])
{
    Q
    Q
    r
}
```



***файлы проекта должны быть  
сохранены в UTF8***

# Файлы проекта

- \*.h – заголовочные файлы C++, содержат объявления классов и функций
- \*.cpp – исходные файлы C++, содержат определения функций
- \*.ui – классы форм Qt, редактируются Qt Designer
- \*.qrc – ресурсы Qt
- \*.pro – файл проекта Qt
- \*.qml – описание интерфейса Qt Quick
- \*.pro.user – настройки Qt Creator

# .pro – Настройки проекта Qt

```
QT += core gui sql
TARGET = QCOptimizer
TEMPLATE = app
INCLUDEPATH += ../OptLib/

SOURCES += main.cpp\
           mainwindow.cpp \
           plugins.cpp

HEADERS += mainwindow.h \
           InterfacePlugin.h \
           Plugins.h

FORMS += mainwindow.ui

RESOURCES += qcoptimizer.qrc

CONFIG(debug)
{
    DESTDIR=../../bin/debug
}

CONFIG(release)
{
    DESTDIR=../../bin/release
}

LIBS += -lOptLib -L$$DESTDIR
include(../fileWidget/fileWidget.pri)
```

# Начнём с QString

---

# Класс QString

Текстовые строки в Qt представляются классом **QString**

Внутри QString текст хранится в UTF-16LE

Как создать:

```
QString s1="hello", s2=QObject::tr("привет");
```

Длина строки:

```
int len=s1.size(); // Длина строки, аналог .length()  
bool b=s2.isEmpty(); // Пустая или нет
```

До	<code>QString().isNull(); // true</code>	<code>QString().isEmpty(); // true</code>	
S	<code>QString("").isNull(); // <i>false</i></code>	<code>QString("").isEmpty(); // <i>true</i></code>	
S			
Q	<code>QString("abc").isNull(); // false</code>	<code>QString("abc").isEmpty(); // false</code>	истрее

# Класс QString - Преобразования

- Кодировки

```
QString s2=QString::fromUtf8("Hello");
```

- QString в число:

```
bool ok;
```

```
double d=QString("1234.56e-02").toDouble(&ok); //12.3456
```

```
int i=QString("17").toInt(&ok); //17
```

```
long hex=QString("FF").toInt(&ok,16); //255
```



# Класс QString - Форматирование

- **Функция arg():**

```
int i; // current file's number
int total; // number of files to process
QString fileName; // current file's name
```

```
QString status = QString("Processing file %1 of %2:
%3").arg(i).arg(total).arg(fileName);
```

- **Статическая функция number():**

```
long a = 63;
QString s = QString::number(a, 16); // s == "3f"
QString t = QString::number(a, 16).toUpper();
// t == "3F"
```

```
double d = 6.3;
QString s = QString::number(d, 'f', 3);
// s == "6.300"
```

# Класс QString – Операции со строками

```
QString str = "and";
str.prepend("rock ");
// str == "rock and"
str.append(" roll");
// str == "rock and roll"
str.replace(5, 3, "&");
// str == "rock & roll"

str = tr("rock ") + tr("and ")
+ tr("roll");
```

```
QString str = "Berlin";
str.fill('z');
// str == "zzzzzz"
str.fill('A', 2);
// str == "AA"
```

```
QString str = " lots\t
of\nwhitespace\r\n ";
str = str.simplified();
// str == "lots of whitespace";
```

## Сравнения

```
if (str == "auto" || str ==
"extern" || str == "static")
{ /* ... */ }
```

```
if (str1 > str2) { /* ... */ } //
быстро, но по кодам
```

```
if (QString::localeAwareCompare
(str1, str2) < 0)
```

по алфавиту

```
{ /* str1 меньше str2 */ }
```

# Класс QString - Поиск

```
QString str = "Peter Pan";
str.contains("peter",
Qt::CaseInsensitive);
//returns true

QString x = "stickyquestion";
QString y = "sti";
x.indexOf(y);
// returns 0
x.indexOf(y, 1);
// returns 10
x.indexOf(y, 11);
//returns -1
```

```
QString str = "the minimum";
str.indexOf(QRegExp
("m[aeiou]"), 0);
// returns 4
QString x = "crazy azimuths";
QString y = "az";
x.lastIndexOf(y);
// returns 6
x.lastIndexOf(y, 5);
// returns 2

QString str = "Bananas";
str.endsWith("anas");
// returns true
```

# Класс QString – Разбиение на подстроки

```
QString str;
QString csv = "forename,middlename,surname,phone";
QString path = "/usr/local/bin/myapp";
// First field is empty
str = csv.section(' ', 2, 2); // str == "surname"
str = path.section('/', 3, 4); // str == "bin/myapp"
str = path.section('/', 3, 3, QString::SectionSkipEmpty);
// str == "myapp"
str = csv.section(' ', -3, -2);
// str == "middlename,surname"

QString str = "a,,b,c";
QStringList list1 = str.split(",");
// list1: [ "a", "", "b", "c" ]
QStringList list2 = str.split(",", QString::SkipEmptyParts);
// list2: [ "a", "b", "c" ]
```

# GUI B Qt

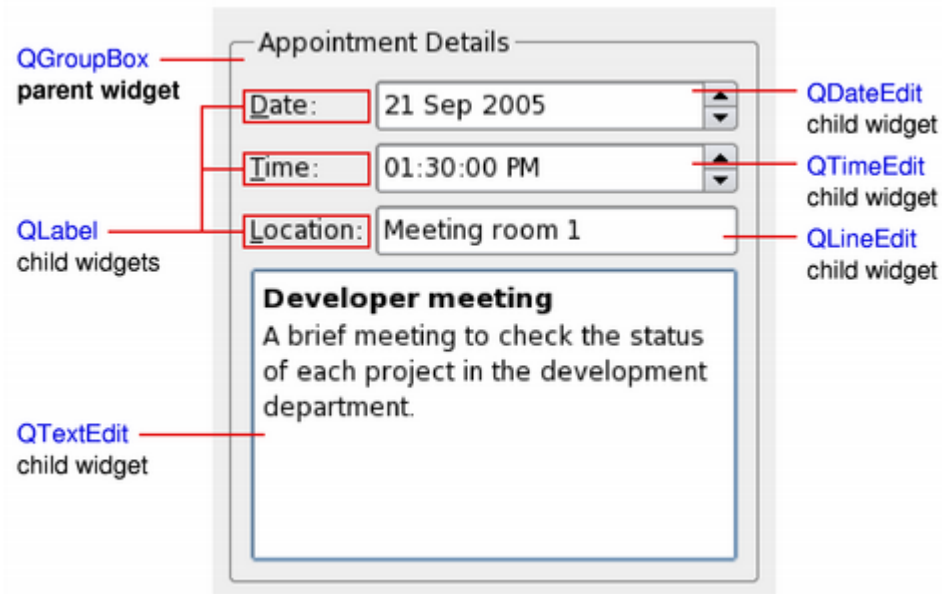
---

# Виджеты

- Атомарная единица пользовательского интерфейса:
  - Получают сообщения от клавиатуры, мыши и другие
  - Рисуют себя на экране
  - Имеют прямоугольную форму
  - Отсортированы по глубине (Z-order)
  - Видимая область обрезается по границам родительского виджета и виджетов, которые находятся перед ним.
- Виджет, который не входит в другой виджет, называется **ОКНОМ**.
  - Окна обычно имеют заголовок и толстую рамку.
- Все виджеты в Qt унаследованы от класса QWidget

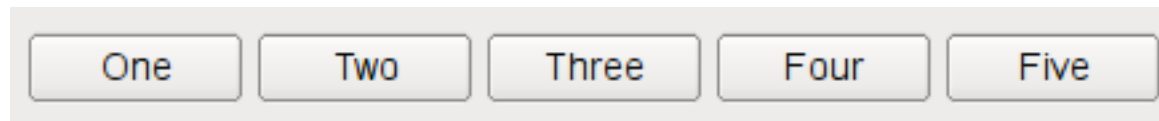
# Дочерние виджеты

- Виджет без родителей – окно верхнего уровня.
  - Заголовок устанавливается `setWindowTitle()`
  - Иконка устанавливается `setWindowIcon()`
- Виджеты, которые не являются окнами, отображаются внутри своих родительских виджетов.

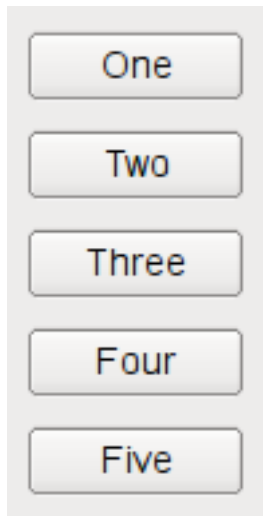


# Расположение виджетов - QLayout

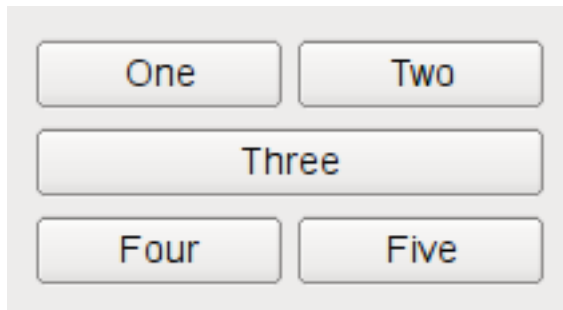
## QHBoxLayout



## QVBoxLayout



## QGridLayout



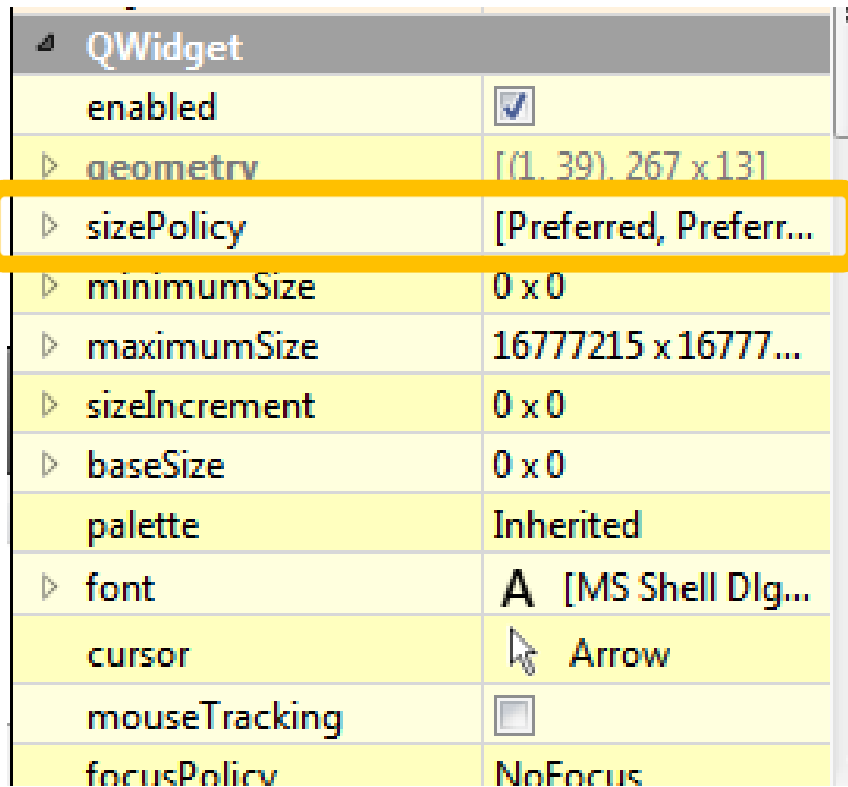
## QFormLayout






# Размер виджетов

- Минимальный размер
  - `minimalSize`
- Максимальный размер
  - `maximalSize`
- Предпочтительный размер
  - `sizeHint`
- Стратегия размера
  - `sizePolicy`:
    - `horizontalPolicy`
    - `verticalPolicy`
    - `horizontalStretch`
    - `verticalStretch`



QWidget	
<code>enabled</code>	<input checked="" type="checkbox"/>
<code>geometry</code>	[(1, 39), 267 x 131]
<code>sizePolicy</code>	[Preferred, Preferr...]
<code>minimumSize</code>	0 x 0
<code>maximumSize</code>	16777215 x 16777...
<code>sizeIncrement</code>	0 x 0
<code>baseSize</code>	0 x 0
<code>palette</code>	Inherited
<code>font</code>	A [MS Shell Dlg...]
<code>cursor</code>	 Arrow
<code>mouseTracking</code>	<input type="checkbox"/>
<code>focusPolicy</code>	NoFocus

# Стратегии размера

Стратегия	Описание
<code>QSizePolicy::Fixed</code>	Допустим только размер, заданный <code>sizeHint</code> .
<code>QSizePolicy::Minimum</code>	<code>sizeHint</code> – минимальный и достаточный. Можно делать больше, но это не даёт преимуществ.
<code>QSizePolicy::Maximum</code>	<code>sizeHint</code> – максимально допустимый размер. Может уменьшаться, если место требуется другим.
<code>QSizePolicy::Preferred</code>	<code>sizeHint</code> – оптимален. Можно уменьшить, если нужно. Можно увеличить, но это не даёт преимуществ.
<code>QSizePolicy::Expanding</code>	<code>sizeHint</code> – хороший размер. Допустимо уменьшать. Желательно увеличить на всё доступное место.
<code>QSizePolicy::MinimumExpanding</code>	Меньше <code>sizeHint</code> нельзя. Желательно увеличить на всё доступное место.
<code>QSizePolicy::Ignored</code>	<code>sizeHint</code> игнорируется. Виджет занимает всё доступное место.

# Алгоритм определения размера

1. Каждому виджету выделяется место согласно `sizeHint` и стратегии размера.
2. Если есть виджеты, у которых установлен `stretchFactor`, то их размеры устанавливаются пропорционально ему.



3. Виджеты у которых `stretchFactor==0` получают дополнительный размер только в том случае, если место больше никому не требуется. В первую очередь место выделяется виджетам со стратегией `Expanding`.
4. Если размер виджета меньше минимального, то ему выделяется его минимальный размер.
5. Если размер виджета больше максимального, то размер уменьшается до максимального.

# Немного ввода

---

# QInputDialog

```
#include <QInputDialog>
```

```
QString heroName = QInputDialog::getText(  
    NULL, // родительский виджет  
    tr("Adventure Game -- окно приветствия"),  
    //заголовок  
    tr("Добрый день! Введите имя героя:"));  
    //приглашение
```

- Другие функции:
  - getInt()
  - getDouble()
  - getItem()

