

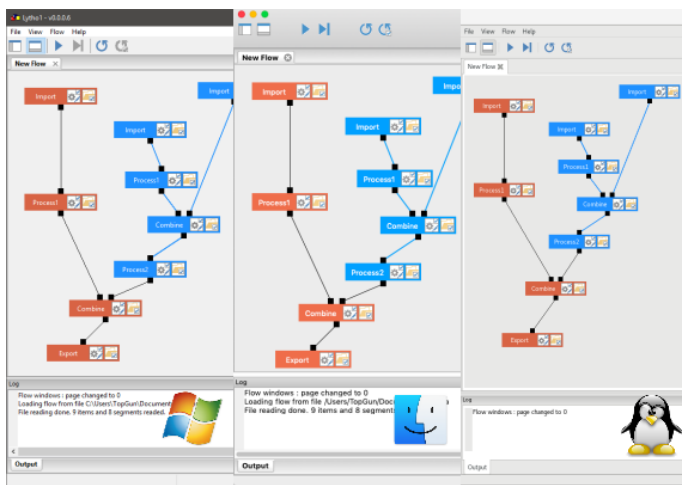


**МИЭТ**

Национальный исследовательский университет «МИЭТ»

Институт интегральной электроники (группы ИВТ-15М, ИВТ-16М, каф. ПКИМС)

# Кроссплатформенная разработка программного обеспечения



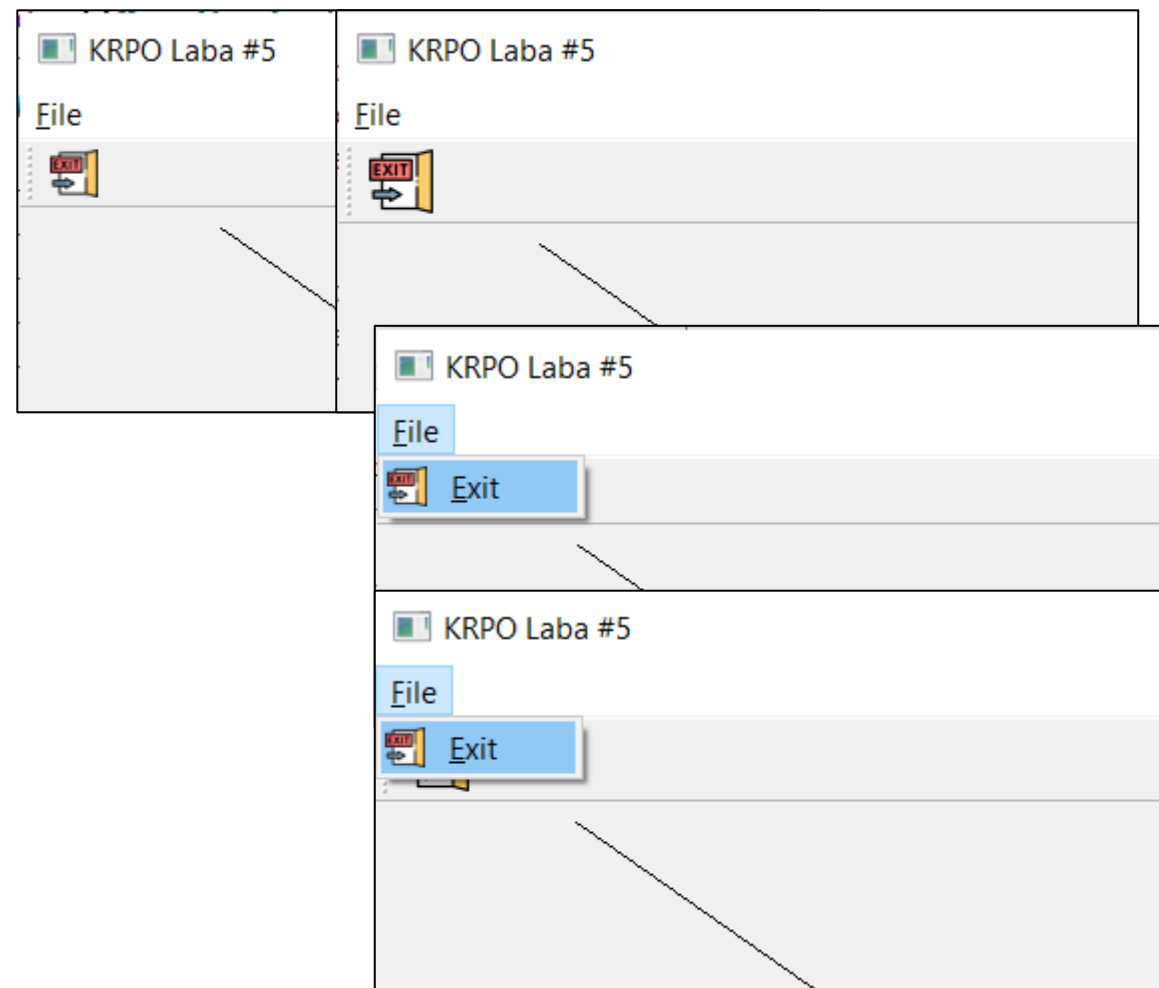
Лабораторная работа №5

**Виджеты отображения текстовых данных**

**Работа с диалогами в Qt**

## Размер иконок в панели инструментов (2)

```
void MainWindow::initToolBar() {  
    QToolBar *toolBar = new QToolBar(this);  
    toolBar->addAction(actFileExit);  
  
    toolBar->setIconSize(QSize(32, 32));  
  
    addToolBar(toolBar);  
}
```



## Варианты работы с текстом в QTextEdit (1)

```
MainWindow::MainWindow() : QMainWindow(NULL) {  
    resize(QSize(800, 600));  
    setWindowTitle("KRPO Laba #5");  
  
    initActions();  
    initMenuBar();  
    initToolBar();  
    initStatusBar();  
    initLog();  
  
    log->insertHtml("<b>info</b> The app has just started!<br>");  
  
    painterWidget = new PainterWidget(this);  
    setCentralWidget(painterWidget);  
}
```

## Варианты работы с текстом в QTextEdit (2)

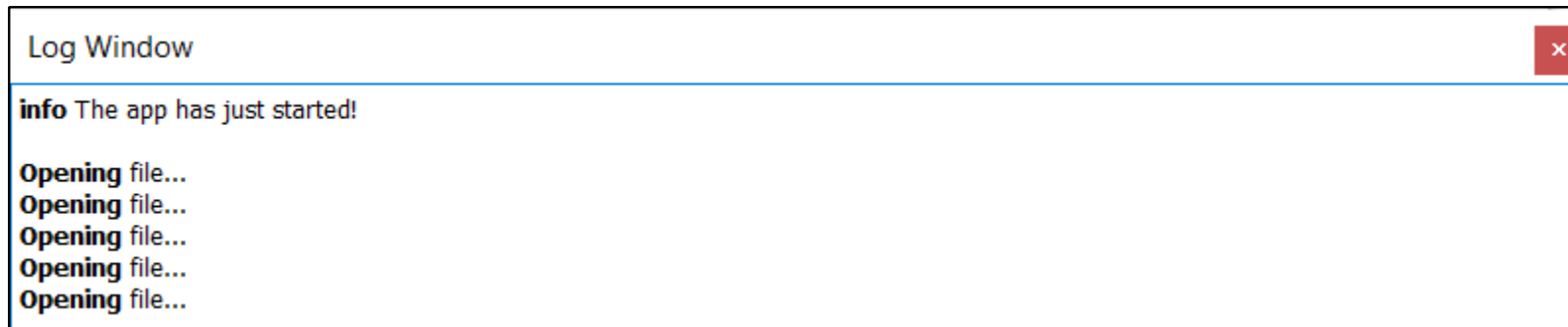
```
void MainWindow::onMenuFileOpen() {  
    log->insertHtml("<b>Opening</b> file...");  
}
```

```
log->insertHtml("<b>Opening</b> file...");
```

```
log->setText("<b>Opening</b> file...");
```

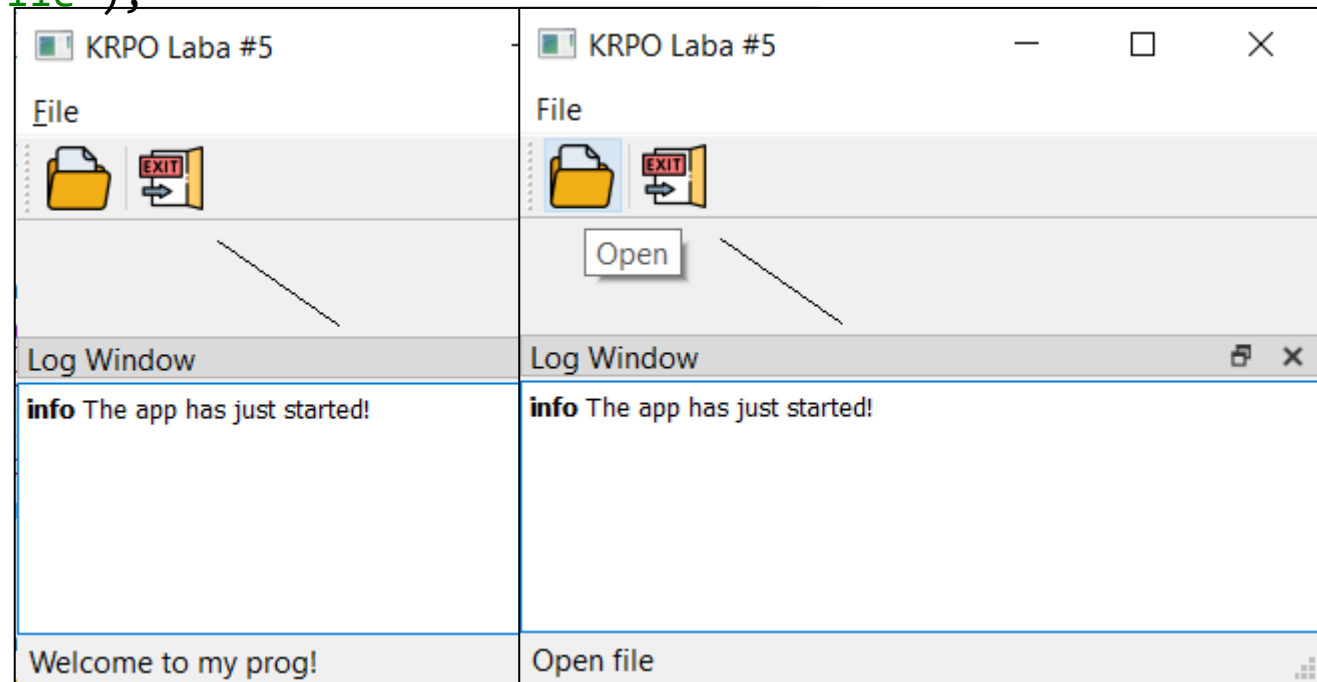
```
log->setPlainText("<b>Opening</b> file...");
```

```
log->append("<b>Opening</b> file...");
```



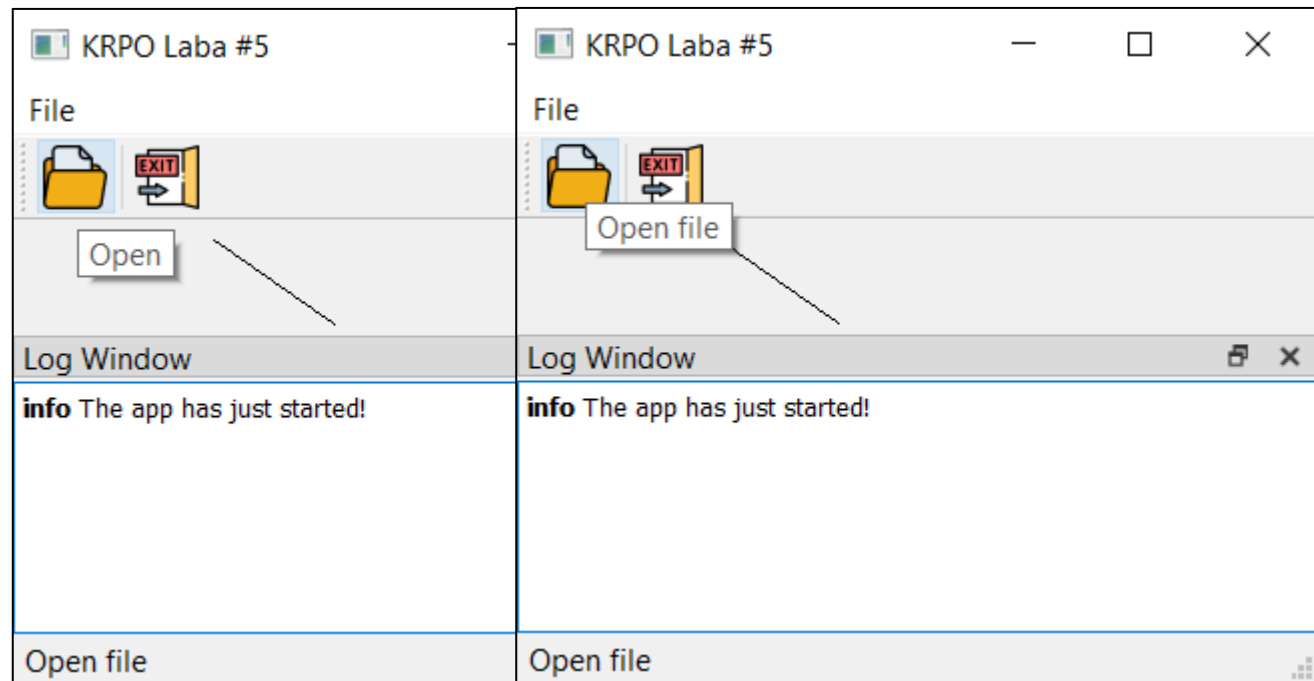
## Вывод подсказок в строку состояния

```
void MainWindow::initActions() {  
    actFileOpen = new QAction(  
        QIcon("/home/student/KRPO_Lab_0x05/task_01/open.png"),  
        "&Open",  
        this);  
    connect(actFileOpen, SIGNAL(triggered()), this, SLOT(onMenuFileOpen()));  
  
    actFileOpen->setStatusTip("Open file");  
  
    ...  
}
```



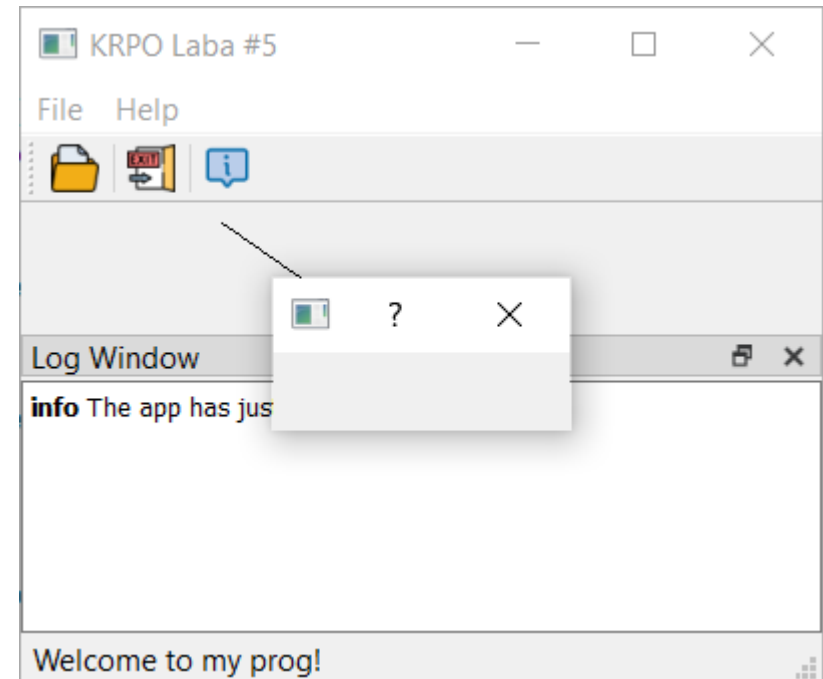
## Изменение всплывающих подсказок в панели инструментов

```
void MainWindow::initActions() {  
    actFileOpen = new QAction(  
        QIcon("/home/student/KRPO_Lab_0x05/task_01/open.png"),  
        "&Open",  
        this);  
    connect(actFileOpen, SIGNAL(triggered()), this, SLOT(onMenuFileOpen()));  
    actFileOpen->setStatusTip("Open file");  
  
    actFileOpen->setToolTip("Open file");  
  
    ...  
}
```



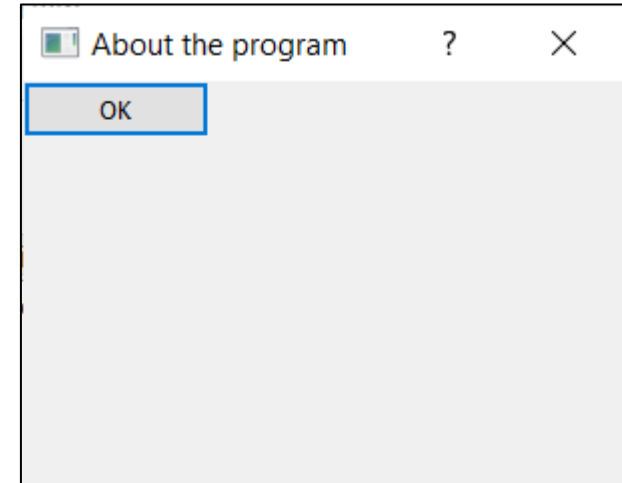
## Вызов диалогового окна

```
void MainWindow::onMenuHelpAbout() {  
  
    QDialog dlg(this);  
    dlg.exec();  
  
}
```



## Возврат результата из диалогового окна

```
void MainWindow::onMenuHelpAbout() {  
    QDialog dlg(this);  
    dlg.setWindowTitle("About the program");  
    dlg.resize(QSize(300, 200));  
  
    QPushButton btnOK("OK", &dlg);  
  
    connect(&btnOK,  
           SIGNAL(clicked()),  
           &dlg,  
           SLOT(accept()));  
  
    dlg.exec();  
}
```





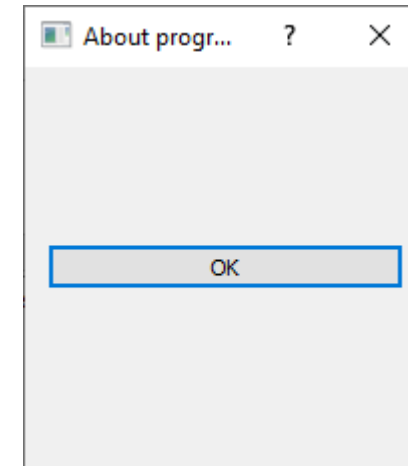
## Диалог «О программе» (1)

```
#pragma once
```

```
#include <QDialog>
```

```
class Dialog_About : public QDialog {  
    Q_OBJECT  
public:  
    Dialog_About(QWidget *parent);  
};
```

task\_02



## Диалог «О программе» (2)

```
#include "Dialog_About.h"
```

```
#include <QLayout>
```

```
#include <QPushButton>
```

```
Dialog_About::Dialog_About(QWidget *parent) : QDialog(parent) {  
    setWindowTitle("About program...");  
    resize(QSize(200, 200));
```

```
    QVBoxLayout *vLayout = new QVBoxLayout(this);
```

```
    QPushButton *btnOk = new QPushButton("OK", this);  
    connect(btnOk, SIGNAL(clicked()), this, SLOT(accept()));  
    vLayout->addWidget(btnOk);
```

```
    setLayout(vLayout);
```

```
}
```

## Текстовая метка: QLabel

```
#include "AboutDialog.hpp"

#include <QVBoxLayout>

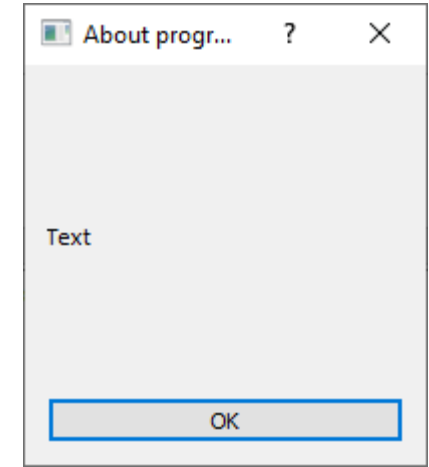
AboutDialog::AboutDialog(QWidget *parent) : QDialog(parent) {
    setWindowTitle("About program...");
    resize(QSize(200, 200));

    QVBoxLayout *vLayout = new QVBoxLayout(this);

    QLabel *label = new QLabel("Text", this);
    vLayout->addWidget(label);

    QPushButton *btnOk = new QPushButton("OK", this);
    connect(btnOk, SIGNAL(clicked()), this, SLOT(accept()));
    vLayout->addWidget(btnOk);

    setLayout(vLayout);
}
```



## Оформление QLabel (1)

```
QLabel *label = new QLabel("Text", this);
```



```
QLabel *label = new QLabel("Text <b>text</b>", this);
```



```
QLabel *label = new QLabel("Text <b>text</b>", this);  
label->setStyleSheet("color: #0f0; background-color: #f0f;");
```

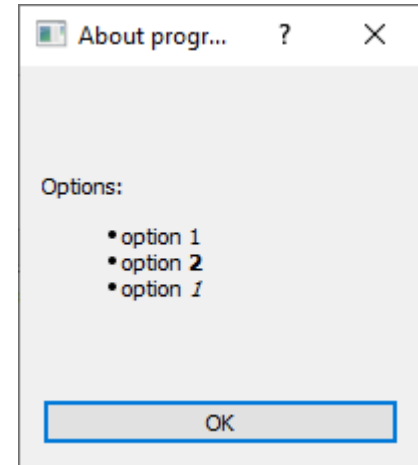
## Оформление QLabel (2)

```
QLabel *label = new QLabel("Text", this);
```



```
QString htmlText =  
    "Options:"  
    "<ul>"  
    "<li>option 1</li>"  
    "<li>option <b>2</b></li>"  
    "<li>option <i>1</i></li>"  
    "</ul>";
```

```
QLabel* label = new QLabel(htmlText, this);
```



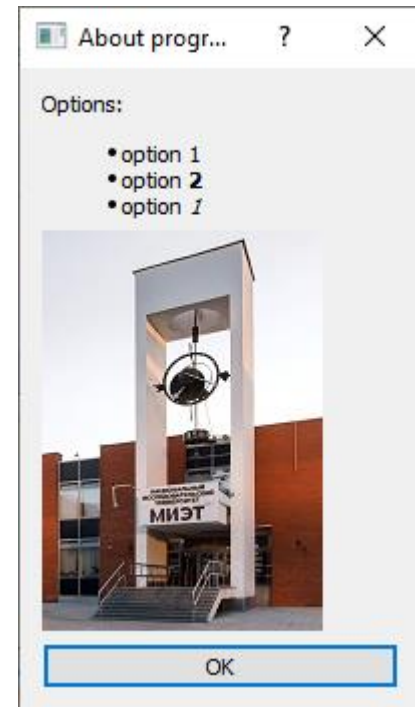
## Оформление QLabel (3)

```
QLabel *label = new QLabel(htmlText, this);  
vLayout->addWidget(label);
```

```
QLabel *label2 = new QLabel(this);  
vLayout->addWidget(label2);
```

```
QImage pic("/home/student/KRPO_Lab_0x05/task_02/miet.png");  
label2->setPixmap(QPixmap::fromImage(pic));
```

```
QPushButton *btnOk = new QPushButton("OK", this);  
connect(btnOk, SIGNAL(clicked()), this, SLOT(accept()));
```



## QLayout внутри QLayout

```
QVBoxLayout *vLayout = new QVBoxLayout(this);
```

```
QHBoxLayout *hLayout = new QHBoxLayout(this);
```

```
QString htmlText =
```

```
    "Options:"
```

```
    "<ul>"
```

```
    "<li>option 1</li>"
```

```
    "<li>option <b>2</b></li>"
```

```
    "<li>option <i>1</i></li>"
```

```
    "</ul>";
```

```
QLabel *label = new QLabel(htmlText, this);
```

```
hLayout->addWidget(label);
```

```
QLabel *label2 = new QLabel(this);
```

```
hLayout->addWidget(label2);
```

```
QImage pic("/home/student/KRPO_Lab_0x05/task_02/miet.png");
```

```
label2->setPixmap(QPixmap::fromImage(pic));
```

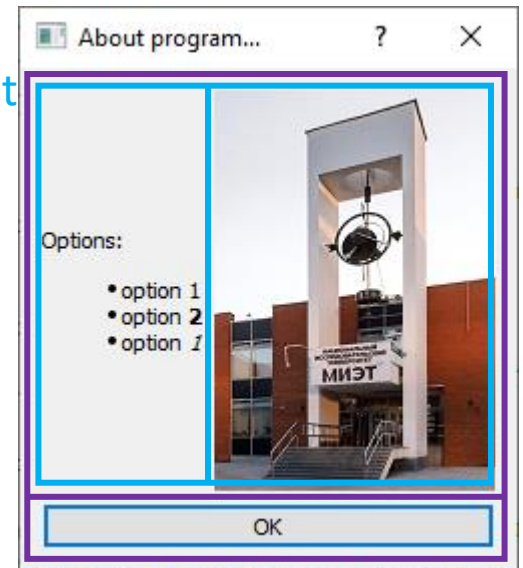
```
vLayout->addLayout(hLayout);
```

```
QPushButton *btnOk = new QPushButton("OK", this);
```

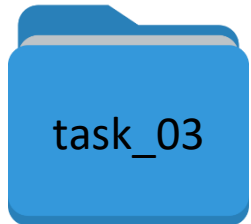
```
connect(btnOk, SIGNAL(clicked()), this, SLOT(accept()));
```

QHBoxLayout

QVBoxLayout



## Стандартные диалоги: диалог открытия файлов (3)



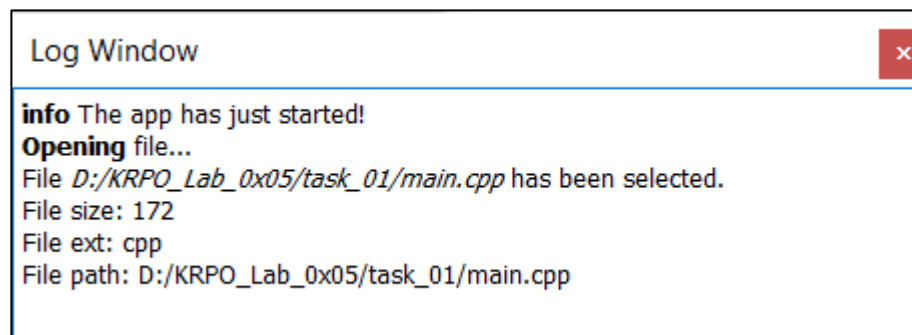
```
void MainWindow::onMenuFileOpen() {  
    log->insertHtml("<b>Opening</b> file...");  
  
    QString fileName = QFileDialog::getOpenFileName(  
        this,  
        "Open code",  
        "/home/student/KRPO_Lab_0x05/task_03/",  
        "C++ Files (*.c *.cpp *.h *.hpp);;All files (*.*)");  
  
    log->insertHtml(QString("File <i>%1</i> has been selected.<br>").arg(fileName));  
}
```



## Работа с классом QFileInfo

```
QFileInfo finfo(fileName);
```

```
log->insertHtml(  
    QString("File size: %1<br>File ext: %2<br>File path: %3<br>")  
        .arg(finfo.size())  
        .arg(finfo.completeSuffix())  
        .arg(finfo.absoluteFilePath())  
);
```



## Работа с файлами в Qt. Класс QFile

```
QFile f(fileName);  
  
if(f.open(QIODevice::ReadOnly | QIODevice::Text)) {  
  
    ...  
  
    f.close();  
}
```

Варианты чтения из файла:

```
QByteArray block = f.read(10);
```

```
QByteArray block = f.readAll();
```



## Чтение текстовых данных из файла

```
QFile f(fileName);

if(f.open(QIODevice::ReadOnly | QIODevice::Text)) {

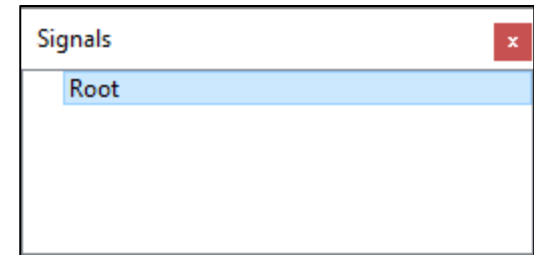
    QTextStream stream(&f);

    while(!stream.atEnd()) {
        log->append(stream.readLine());
    }

    f.close();
}
```

## Вывод структурированных данных. Класс QTreeWidget

```
void MainWindow::initTree() {  
    QDockWidget *dock = new QDockWidget("Signals", this);  
  
    QTreeWidget *tree = new QTreeWidget(this);  
    tree->setHeaderHidden(true);  
  
    QTreeWidgetItem *topItem = new QTreeWidgetItem;  
    topItem->setText(0, "Root");  
    tree->insertTopLevelItem(0, topItem);  
    dock->setWidget(tree);  
  
    addDockWidget(Qt::LeftDockWidgetArea, dock);  
    dock->setAllowedAreas(Qt::LeftDockWidgetArea | Qt::RightDockWidgetArea);  
}
```

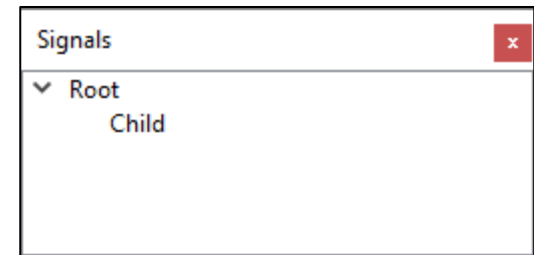


## Добавление элементов к дереву

```
QTreeWidgetItem *topItem = new QTreeWidgetItem;  
topItem->setText(0, "Root");  
tree->insertTopLevelItem(0, topItem);
```

```
QTreeWidgetItem *item = new QTreeWidgetItem;  
item->setText(0, "Child");  
topItem->addChild(item);
```

```
dock->setWidget(tree);
```



## Реализация дерева для проекта с обработкой файла (1)

```
class MainWindow : public QMainWindow {
    Q_OBJECT
private:
    QAction *actFileOpen,
            *actFileExit,
            *actHelpAbout;
    PainterWidget *painterWidget;
    QTextEdit *log;
    QTreeWidget *tree;
public:
    MainWindow();
private:
    void initActions();
    ...
}
```

```
void MainWindow::initTree() {
    QDockWidget *dock = new QDockWidget("Signals", this);

    tree = new QTreeWidget(this);
    tree->setHeaderHidden(true);

    dock->setWidget(tree);

    addDockWidget(Qt::LeftDockWidgetArea, dock);
    ...
}
```

## Реализация дерева для проекта с обработкой файла (2)

```
void MainWindow::onMenuFileOpen() {  
    ...  
    QString fileName = QFileDialog::getOpenFileName(  
        this,  
        "Open code",  
        "/home/student/KRPO_Lab_0x05/task_03/",  
        "C++ Files (*.c *.cpp *.h *.hpp);;All files (*.*)");  
  
    log->insertHtml(QString("File <i>%1</i> has been selected.<br>").arg(fileName));  
  
    QFileInfo finfo(fileName);  
    ...  
  
    QTreeWidgetItem *topItem = new QTreeWidgetItem;  
    topItem->setText(0, finfo.fileName());  
    tree->insertTopLevelItem(0, topItem);  
    ...  
}
```

## Реализация дерева для проекта с обработкой файла (3)

```
int    numberOfPlots = 0;
QString plotName;

stream >> numberOfPlots;
log->append(QString("Plots in file: %1").arg(numberOfPlots));

for(int i = 0; i < numberOfPlots; ++i) {
    stream >> plotName;
    log->append(QString(" Plot %1: %2").arg(i).arg(plotName));

    QTreeWidgetItem *item = new QTreeWidgetItem;
    item->setText(0, plotName);
    topItem->addChild(item);
}
```





## Обработка событий для элементов дерева (1)

```
class MainWindow : public QMainWindow {  
    Q_OBJECT  
private:  
    QAction *actFileOpen,  
    ...  
private slots:  
    void onMenuFileOpen();  
    void onMenuFileExit();  
    void onMenuHelpAbout();  
    void onSelectTreeWidgetItem(QTreeWidgetItem *item, int column);  
};
```

## Обработка событий для элементов дерева (2)

```
void MainWindow::initTree() {
    QDockWidget *dock = new QDockWidget("Signals", this);

    tree = new QTreeWidget(this);
    tree->setHeaderHidden(true);

    connect(tree,
            SIGNAL(itemActivated(QTreeWidgetItem*, int)),
            this,
            SLOT(onSelectTreeItem(QTreeWidgetItem*, int)));

    dock->setWidget(tree);

    addDockWidget(Qt::LeftDockWidgetArea, dock);
    ...

void MainWindow::onSelectTreeItem(QTreeWidgetItem *item, int column) {
    log->append(QString("Item %1 activated").arg(item->text(0)));
}
```

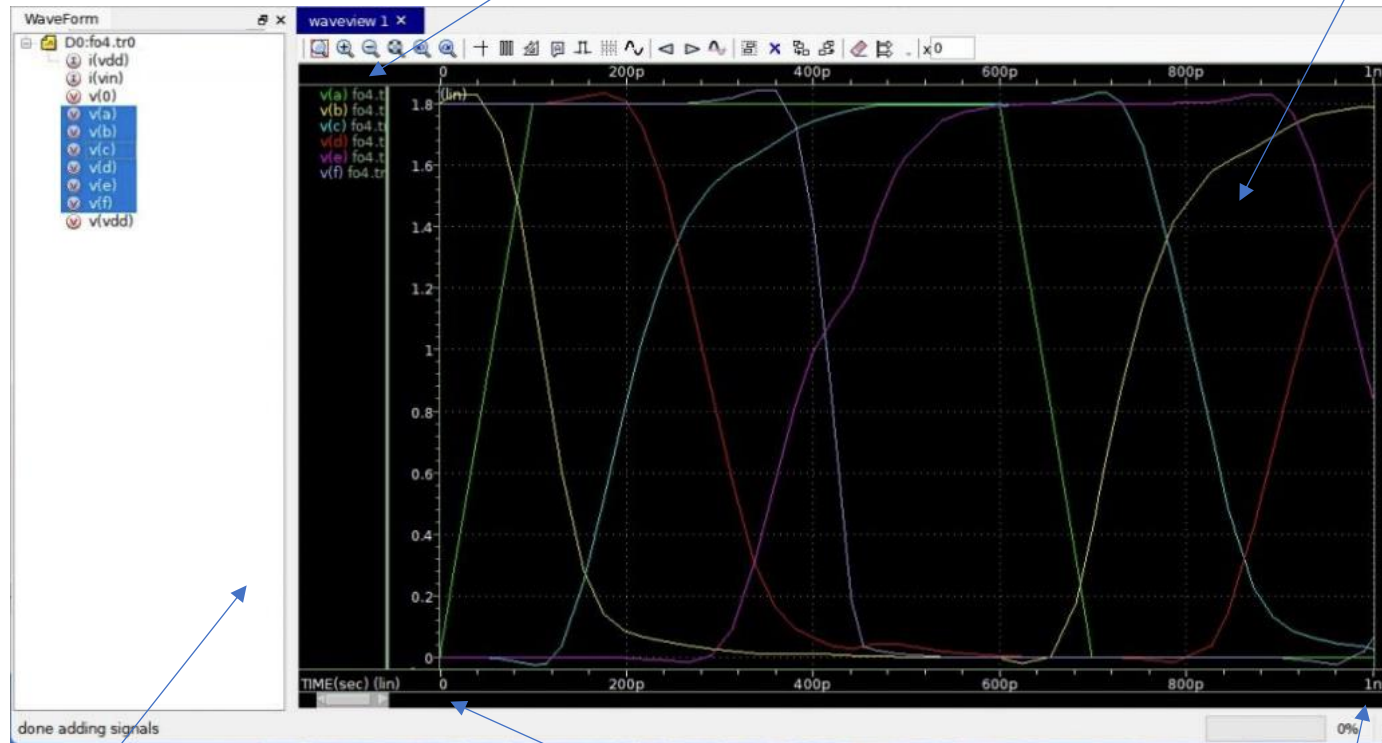
## Рисование с QPainter

```
void PainterWidget::paintEvent(QPaintEvent *event) {  
    QPainter p(this);  
  
    QPen pen(QColor(0, 0, 0) , 2, Qt::SolidLine);  
    p.setPen(pen);  
  
    p.drawLine(QPoint(10, 10), QPoint(100, 100));  
  
    pen.setWidth(4);  
    p.setPen(pen);  
  
    p.drawLine(QPoint(50, 50), QPoint(200, 50));  
  
    pen.setColor(QColor("#FF00FF"));  
    p.setPen(pen);  
  
    p.drawLine(QPoint(50, 50), QPoint(50, 100));  
}
```



## Форматы хранения результатов моделирования

Клиентская часть окна:



Пристыковывающееся окно с древовидным списком, в котором отображаются имя файла и перечень доступных сигналов. Клик по имени графика, и он добавляется в список отображаемых. Клик ещё раз - и он больше не рисуется.

Перечень сигналов, которые сейчас рисуются. Графики рисуются во всю область отрисовки

Область графиков (цвет фона - чёрный, для рисования графиков предусмотреть не менее 10 цветов)

Должна иметься возможность кликом выделить график. Тогда он рисуется своим цветом, а остальные - серым.

По осям подписываются только минимум и максимум для **рисуемых** данных

Требуется разработать программу, интерфейс которой напоминает Synopsys Custom WaveView, которая визуализирует результаты схемотехнического проектирования в формате, определяемым вариантом

Номер варианта	Поддерживаемый формат
1, 4, 7, 10, 13	PSF
2, 5, 8, 11, 14	CSV
3, 6, 9, 12, 15	CSDF



# Примеры форматов файлов

## HSPICE PSF:

```
HEADER
"PSFversion" "1.00"
"simulator" "HSPICE"
"runtype" "Transient Analysis"
TYPE
"node" FLOAT DOUBLE PROP(
"key" "node"
)
"branch" FLOAT DOUBLE PROP(
"key" "branch"
)
"sweep" FLOAT DOUBLE
SWEEP
"time" "sweep"
TRACE
"group" GROUP 2
"1" "node"
"2" "node"
VALUE
"time" 0.000000e+00
"group"
0.000000e+00
0.000000e+00
"time" 1.000000e-09
"group"
0.000000e+00
0.000000e+00
...
END
```

## CSV:

```
; Additional parameters:
; Temperature (TEMP) = 2.50000000e+001
; Temperature (TNOM) = 2.50000000e+001
; Local inaccuracy = 9.99999700e-003
; reltol = 1.00000000e-003
; Acceleration level = without acceleration

TIME      'v(1) '      'v(2) '
0.00000000e+000  0.00000000e+000  0.00000000e+000  2.00000000e-009
0.00000000e+000  0.00000000e+000  5.00000000e-009  0.00000000e+000
0.00000000e+000  5.02386518e-009  5.96629438e-002  1.39067797e-003
5.07159553e-009  1.78988831e-001  6.88855948e-003  5.12889016e-009
3.22225398e-001  2.04634393e-002  5.18236485e-009  4.55912118e-001
3.96611966e-002  5.28931422e-009  7.23285558e-001  9.54911926e-002
```

## CSDF:

```
#H
SOURCE='SYMSPICE'
TITLE='* # file name: F:\rc'
SUBTITLE=''
TIME='08:37:36' DATE='9/11/2018'
ANALYSIS='TR'
TEMPERATURE=' 2.500000E+001'
SWEEPVAR='TIME'
COMPLEXVALUES='NO' FORMAT='1 VOLTSorAMPS;EFLOAT'
XBEGIN=' 0.000000e+000' XEND=' 1.000000e-007'
NODES='      2'
#N 'v(1) ' 'v(2) '

#C      0.00000000e+000      2      0.00000000e+000      0.00000000e+000
#C      2.00000000e-009      2      0.00000000e+000      0.00000000e+000
#C      5.00000000e-009      2      0.00000000e+000      0.00000000e+000
#C      5.02386518e-009      2      5.96629438e-002      1.39067797e-003
#C      5.07159553e-009      2      1.78988831e-001      6.88855948e-003
#C      5.12889016e-009      2      3.22225398e-001      2.04634393e-002
```