ID	...	**Creation date**	21 April 2009
Platform	S60 3rd Edition FP1, S60 3rd Edition FP2, S60 5th Edition	**Tested on devices**	Emulator
Category	Qt for S60	**Subcategory**	Application

Keywords (APIs, classes, methods, functions): QSignalMapper,QPushButton

Introduction

he QSignalMapper class bundles signals from identifiable senders.

This class collects a set of parameterless signals, and re-emits them with integer, string or widget parameters corresponding to the object that sent the signal.

Preconditions

- Download and Install latest version Qt for Symbian - Installation packages which has links on how to install the latest version

Source Code

Main.cpp

```
#include <QtGui/QApplication>
#include "buttonwidget.h"
int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    QStringList fonts;
     fonts << "Nokia" << "QT for S60" << "Python" << "J2ME";
    ButtonWidget w(fonts);
    w.show();
    return a.exec();
}
```

ButtonWidget.h

```
#ifndef BUTTONWIDGET_H
#define BUTTONWIDGET_H

#include <QtGui/QWidget>
#include<QSignalMapper>
#include<QPushButton>
#include<QGridLayout>
#include<QStringList>
class ButtonWidget : public QWidget
{
    Q_OBJECT

public:
    ButtonWidget(QStringList texts,QWidget *parent = 0);

signals:
    void clicked(const QString &text);

private:
    QSignalMapper *signalMapper;
};

#endif // BUTTONWIDGET_H
```

ButtonWidget.cpp

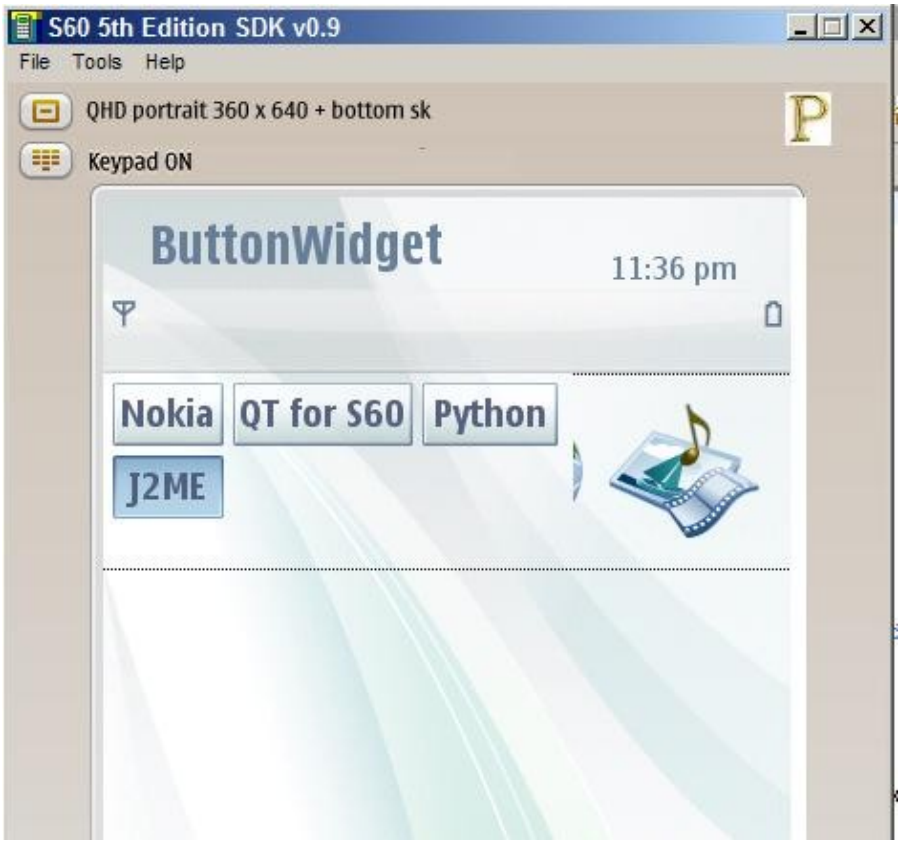
```
#include "buttonwidget.h"

ButtonWidget::ButtonWidget(QStringList texts,QWidget *parent)
    : QWidget(parent)
{
   signalMapper = new QSignalMapper();

     QGridLayout *gridLayout = new QGridLayout;
     for (int i = 0; i < texts.size(); ++i) {
         QPushButton *button = new QPushButton(texts[i]);
         connect(button, SIGNAL(clicked()), signalMapper, SLOT(map()));
         signalMapper->setMapping(button, texts[i]);
         gridLayout->addWidget(button, i / 3, i % 3);
     }

     connect(signalMapper, SIGNAL(mapped(const QString &)),
             this, SIGNAL(clicked(const QString &)));

     setLayout(gridLayout);
}
ButtonWidget::~ButtonWidget()
{
        if(signalMapper)
        {
             delete signalMapper;
        }
}
```

Explanation

A list of texts is passed to the constructor. A signal mapper is constructed and for each text in the list a QPushButton is created. We connect each button's clicked() signal to the signal mapper's map() slot, and create a mapping in the signal mapper from each button to the button's text. Finally we connect the signal mapper's mapped() signal to the custom widget's clicked() signal. When the user clicks a button, the custom widget will emit a single clicked() signal whose argument is the text of the button the user clicked.

ScreenShot

More about QSignalMapper.



More about QSignalMapper.

Related Links

Mapping of StandardItemModel via DataWidgetMapper