Archived:How to create a message box in Qt

Archived: This article is archived because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

Qt Quick should be used for all UI development on mobile devices. The approach described in this article (based on QWidget) is deprecated.

This (archived) code example shows how to create a QMessageBox, a standard dialog for informing the user or for asking the user a question. You can set the message title and use either a custom or predefined icon. You can also specify that the message box is modal.

Various functions

- To set Window Title.

```
msgBox.setWindowTitle("HELP");
```

- This provide the informative text that provides a fuller description for the message.

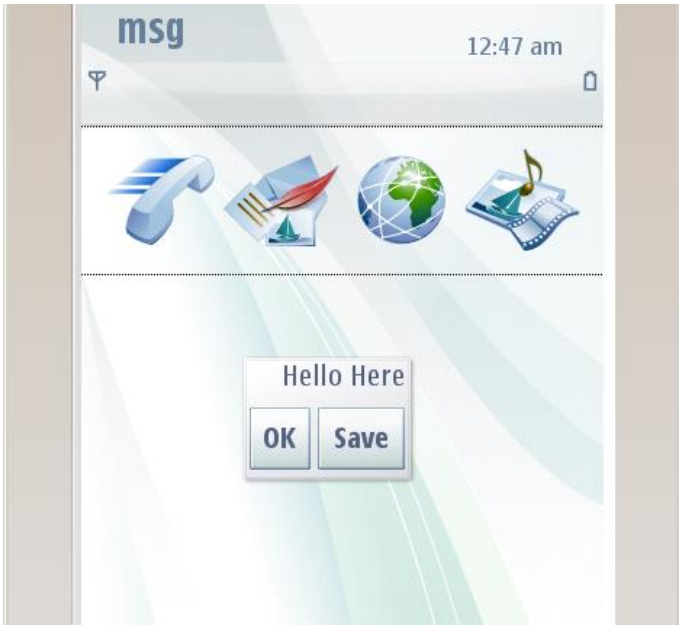
```
msgBox.setInformativeText("Do You Want any Help This???");
```

- Sets the modality of the message box to windowModality.This allow to set the window modality if its true then we can't focus any other window until the msgBox is being closed.

```
msgBox.setWindowModality(Qt::ApplicationModal);
```

- This property holds collection of standard buttons in the message box.

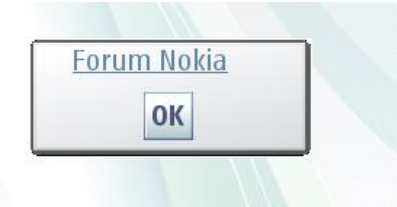
```
msgBox.setStandardButtons(QMessageBox::Ok | QMessageBox::Save);
```



(C) Copyright Nokia 2013. All rights reserved.

Clickable URL in messageBox

```
msgbox->about(0, "Forum.nokia.com", "<a href='http://www.forum.nokia.com'>Nokia Developer</a>");
```



Code

```
#include <QtGui/QApplication>
#include <QMessageBox>
int main(int argc, char *argv[])
{
QApplication app(argc, argv);
QMessageBox msgBox;
msgBox.setText("Hello Here");
msgBox.exec();
app.quit();
return 1;
}
```

Screen Shot

To read More about QMessageBox: click here

Receving Input from user

QMessageBox allows user to set predefined buttons (standard buttons), and then receive input when these button is clicked. **setStandardButtons()** allows to set these buttons, you can add more than one button at a time.

```
        QMessageBox msgBox;
        //Set text
msgBox.setText("Testing..");
        //Set predefined icon, icon is show on left side of text.
msgBox.setIcon(QMessageBox::Information);
        //set inforative text
msgBox.setInformativeText("Just show infornation.");
        //Add ok and cancel button.
msgBox.setStandardButtons(QMessageBox::Ok | QMessageBox::Cancel);
        //Set focus of ok button
msgBox.setDefaultButton(QMessageBox::Ok);

        //execute message box. method exec() return the button value of cliecke button
int ret = msgBox.exec();

        //User get input from returned value (ret). you can handle it here.
 switch (ret) {
   case QMessageBox::Save:
        // Save was clicked
        break;
```


(C) Copyright Nokia 2013. All rights reserved.

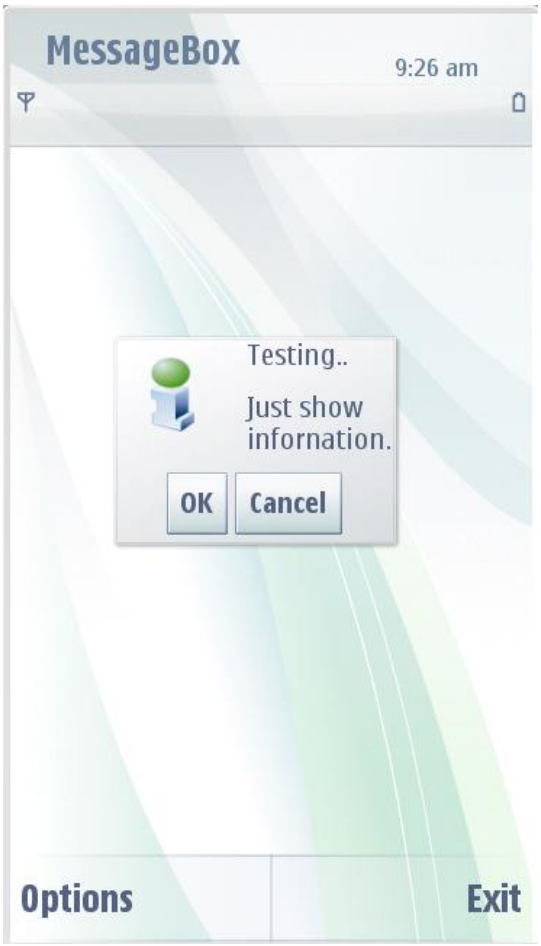
```
    case QMessageBox::Ok:
        // ok was clicked
        break;
    default:
        // should never be reached
        break;
}
```

Static methods of QMessageBox

QMessageBox has three static method, `warning()`, `information()` and `critical()`, which alow to create messagebox, but it does not allow to set custom icon, informative text and detailed text.

```
//Create warning message box
QMessageBox::warning(0,"Warning", "Warning message text");
//Create information message box
QMessageBox::information(0, "Information", "Information message text");
//Create critical message box
QMessageBox::critical(0, "Critical", "Critical message text");
```

ScreenShots



Code Example

Code Example

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.