MessageBox Window

A MessageBox is a small dialog box in the GUI application that display a message to the user and waits for a response. It is commonly used to:

- Show alerts, warnings, or errors.
- Ask for confirmation before performing an action
- Provide information or notification

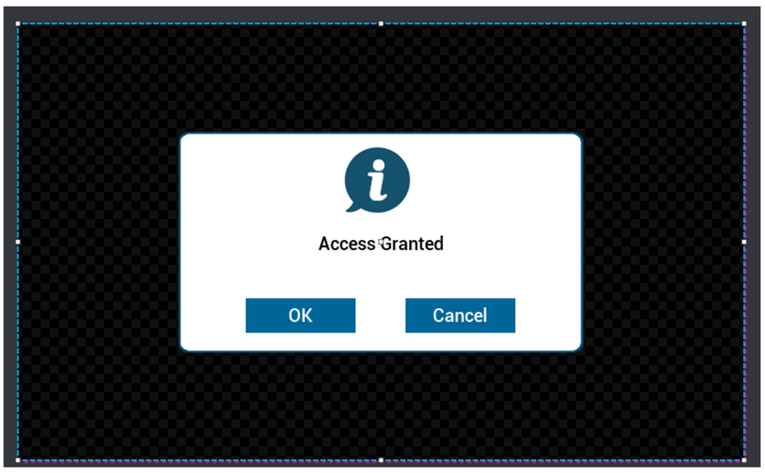
A MessageBox key features:

- Title – the text displayed in the title bar of the dialog
- Message – the main content shown to the user
- Buttons – Options like OK, Cancel, Yes/No, Retry, etc.
- Icons – Visual indicators (information, warning, error, or question)

TouchGFX did not provide MessageBox widget, however it provided a modal window that can be used to create a messagebox.

How It Works?

The MessageBoxWindow is a TouchGFX custom container base on modal window that featured a standar messagebox, except title. It has an icon that can be changed base on message type (information, error, waring, or question/confirmation). It can display text message up to 64 characters with auto-warp capability. The messagebox will have 2 button (OK and Cancel) when show question or confirmation message and give the user status response, click the OK button or Cancel button.



The MessageBox will have *ShowMessageBox* function to display the MessageBox

```
void MessageBoxWindow::ShowMessageBox (MessageTypeEnum MSG_Type, const char*
MsgText)
{
    int x;
    boolean OneButtonFlag; //Only OK button will be displayed

    MSGType = MSG_Type;

    //Clear OK Button Status
    OKButtonState=0;
    OneButtonFlag=true;
    switch (MSG_Type)
    {
    case MSG_OK:
        MessageIcon.setBitmap(touchgfx::Bitmap(BITMAP_INFOICON_ID));
        break;
    case MSG_ERROR:
```

```
        MessageIcon.setBitmap(touchgfx::Bitmap(BITMAP_ERRORICON_ID));
        break;
    case MSG_WARNING:
        MessageIcon.setBitmap(touchgfx::Bitmap(BITMAP_WARNINGICON_ID));
        break;
    case MSG_QUESTION:
        MessageIcon.setBitmap(touchgfx::Bitmap(BITMAP_QUESTIONICON_ID));
        OneButtonFlag=false;
        break;
    }

    if(OneButtonFlag){
        CancelflexButton.setVisible(false);
        x=339;
    }
    else{
        CancelflexButton.setVisible(true);
        x = 251;
    }
    OKflexButton.setXY(x,302);
    Unicode::strncpy(MessageTextBuffer, MsgText, MESSAGETEXT_SIZE);
    this->setVisible(true);
    this->invalidate();
}
```

The function has 2 parameters: *MSG_Type* and *MsgText*. Msg_Type will define the icon should be displayed:

- MSG_OK, for OK or information message
- MSG_ERROR, for error message
- MSG_WARNING, for warning message
- MSG_QUESTION, for question or confirmation message

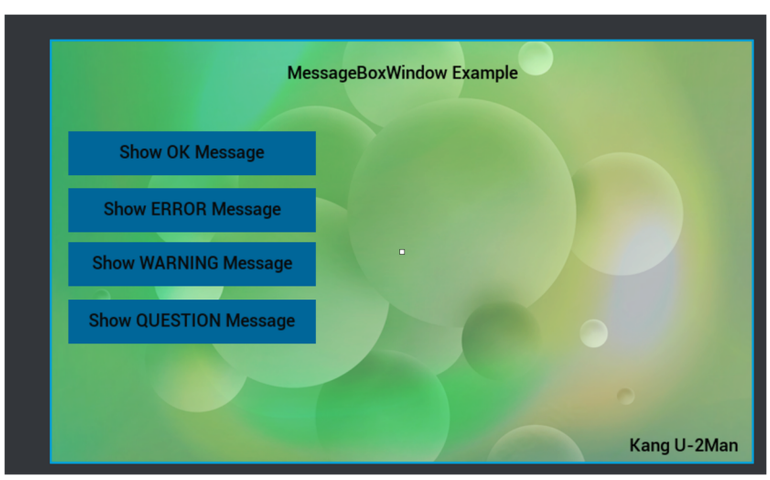
The *OKButtonState* variable will be set when user choose OK button in MSG_QUESTION.

```
void MessageBoxWindow::OKButton_Execute()
{
    //Close the messageBox
    this->setVisible(false);
    this->invalidate();

    //Set OK Button status
    if(MSGType==MSG_QUESTION)
    {
        OKButtonState =1;
    }
}
```

How To Use?

It is very easy to use MessageBoxWindow in the application. Just call the function from a button clicked interaction.



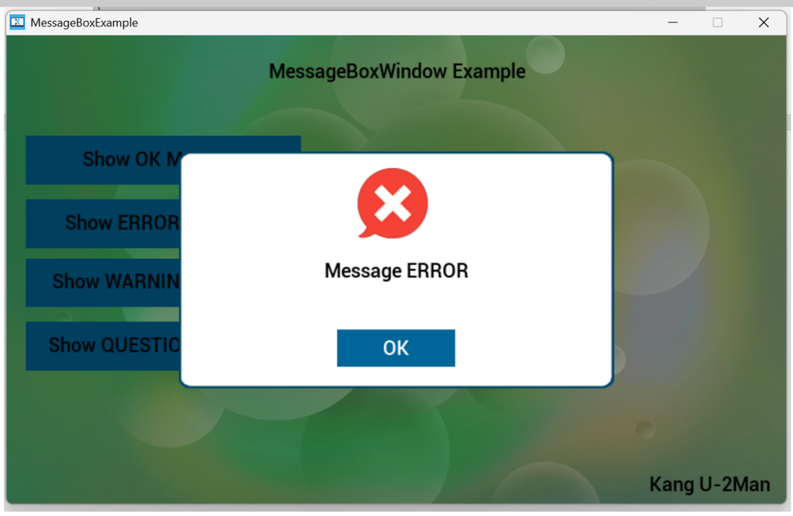

```
void Screen1View::ShowOKMessage()
{
```

```
    messageBoxWindow.ShowMessageBox(MSG_OK,"Message OK");
}

void Screen1View::ShowERRORMessage()
{
    messageBoxWindow.ShowMessageBox(MSG_ERROR,"Message ERROR");
}

void Screen1View::ShowWARNINGMessage()
{
    messageBoxWindow.ShowMessageBox(MSG_WARNING,"Message WARNING");
}

void Screen1View::ShowQUESTIONMessage()
{
    messageBoxWindow.ShowMessageBox(MSG_QUESTION,"Are you sure to exit?");
}
```



To check when user click OK button in MSG_QUESTION type, just check the *OKButtonState* variable from *handleTickEvent* function.

```
void Screen1View::handleTickEvent()
{
    if(messageBoxWindow.OKButtonState){
        messageBoxWindow.ShowMessageBox(MSG_OK,"You click OK Button");
        messageBoxWindow.OKButtonState=0;
    }
}
```

Customization

This messageBox currently develop under 800x480 screen resolution, to use with lower resolution need to resize the modalwindow1 (the image BGFull.png), the image1 (MessageBoxBG.png), the icon size, the button size and maybe the font size. The button position also need to be adjusted follow the screen resolution. The code should change for x position of OK button.

```
    if(OneButtonFlag){
        CancelflexButton.setVisible(false);
        x=339;
    }
    else{
        CancelflexButton.setVisible(true);
        x = 251;
    }
```