



Contents

- [1 Overview](#)
- [2 What is TactileFeedback API](#)
- [3 Implementing TactileFeedback functionality](#)
 - ◆ [3.1 TactileFeedbackAppView.h](#)
 - ◆ [3.2 TactileFeedbackAppView.cpp](#)
- [4 Useful functions](#)
 - ◆ [4.1 MTouchFeedback](#)
 - ◆ [4.2 MCoeControlHitTest](#)
- [5 Keywords](#)
 - ◆ [5.1 Headers](#)
 - ◆ [5.2 Classes](#)
 - ◆ [5.3 Libraries](#)
- [6 Example Application](#)
- [7 Related links](#)
- [8 Reference list](#)

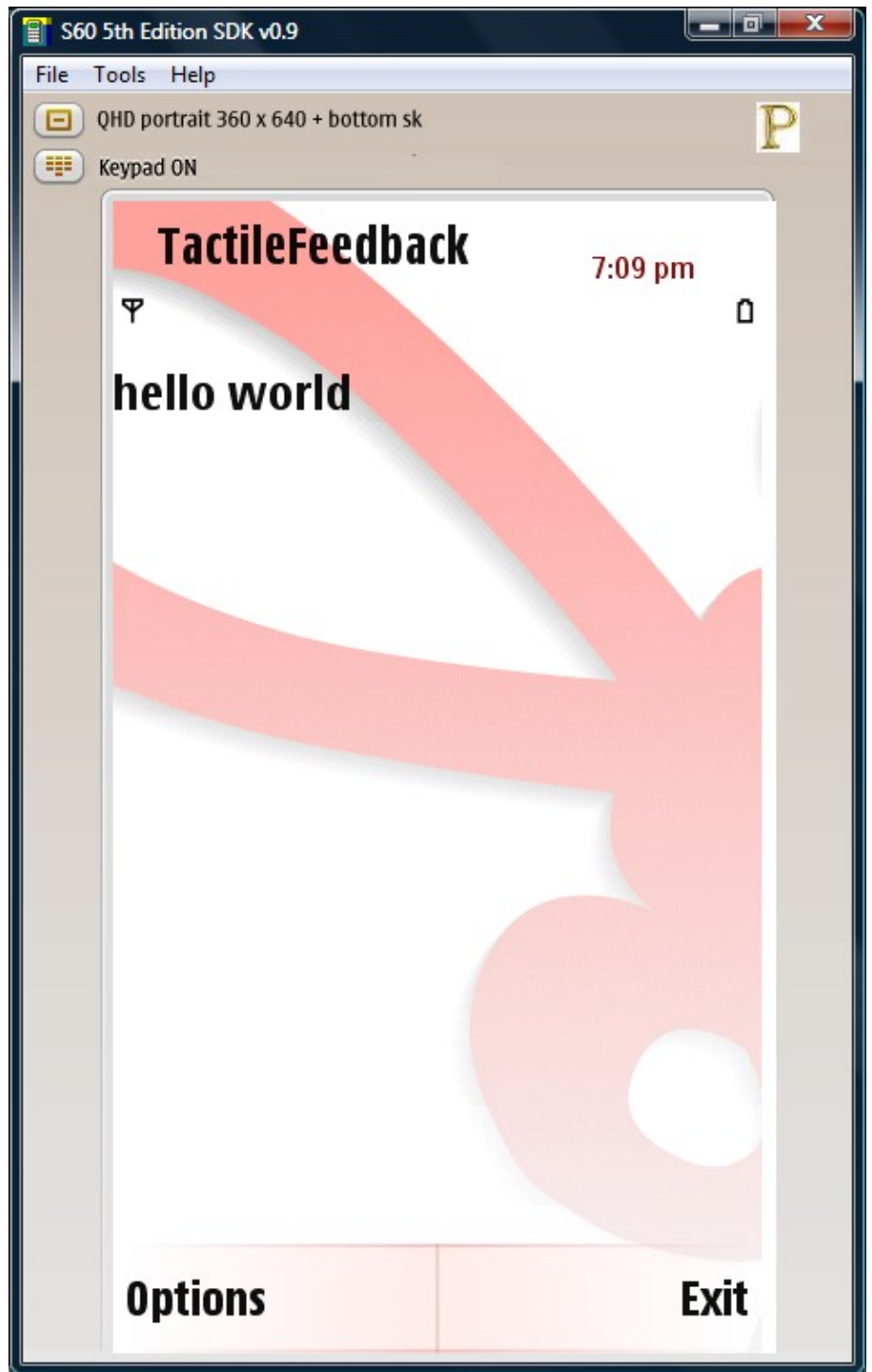
Overview

Tactile Feedback API is a new feature offered by S60 5th edition Touch UI concept for giving a feeling of touch on the screen. The device will respond to the touch inputted by the end user. The Feedback could be in the form of sound or vibration or both. In this article we will learn how to play with Tactile Feedback API with different methods.

What is TactileFeedback API

Tactile Feedback works in two ways, when the user touches the screen or when the user touches any control on the screen. On S60 5th edition platform, most of the UI components has inbuilt feature of Tactile feedback.

From the user experience perspective, care should be taken while implementing Tactile Feedback



functionality in the applications.

In the above figure, we will be sending feedback when the pointer event occurs inside the label control's hit region.

Implementing TactileFeedback functionality

TactileFeedbackAppView.h

- We will inherit our class from `MCoeControlHitTest` interface and implement its pure virtual method `HitRegionContains()` method to receive and handle events for Label control.
- Declare object of `MTouchFeedback` and use it to get instance of `TouchFeedback`.

```

...
...
#include <touchfeedback.h>
#include <eiklabel.h> // For label component

// CLASS DECLARATION
class CTactileFeedbackAppView : public CCoeControl, MCoeControlHitTest
{
    ....
    ....
//For label control
    TInt CountComponent( TInt aIndex ) const;
    CCoeControl* ComponentControl( TInt aIndex ) const;

//From MCoeControlHitTest
    TBool HitRegionContains( TPoint& aPoint, const CCoeControl& /*aControl*/ ) const;

private:
    ....
    ....
MTouchFeedback* iTouchFeedback; // For Tactile feedback
    CEikLabel; // For Label Control
};

```

TactileFeedbackAppView.cpp

- Create Label control in `ConstructL()` as shown below.
- Set "this" as an observer for checking hit region.
- Acquire `TactileFeedback` instance using `MTouchFeedback::Instance()`;

```

// -----
// CTactileFeedbackAppView::ConstructL()
// Symbian 2nd phase constructor can leave.
// -----
//
void CTactileFeedbackAppView::ConstructL(const TRect& aRect)
{
    // Create a window for this application view
    CreateWindowL();

    _LIT(KTextHelloWorld, "hello world");
    iLabel = new (ELeave) CEikLabel;
    iLabel->SetContainerWindowL( *this );
    iLabel->SetTextL(KTextHelloWorld);
    iLabel->SetHitTest( this );

    // Set the windows size
    SetRect( aRect );

    iTouchFeedback = MTouchFeedback::Instance();
}

```

Working_with_Tactile_Feedback_Client_API_-_S60_Touch_UI

```
}
```

- Implement `HitRegionContains()` callback function to determine whether the pointer event occurred in the label control's hit region or not.

```
// -----  
// From class MCoeControlHitTest.  
// Check if a pointer event occurred inside the label region  
// -----  
//  
TBool CTactileFeedbackAppView::HitRegionContains( const TPoint& aPoint, const CCoeControl& /*aCon  
{  
    return iLabel->Rect().Contains( aPoint );  
}
```

- Handle pointer events and send tactile feedback when the pointer event occurs inside the label control.

```
// -----  
// CTactileFeedbackAppView::HandlePointerEventL()  
// Called by framework to handle pointer touch events.  
// -----  
//  
void CTactileFeedbackAppView::HandlePointerEventL(const TPointerEvent& aPointerEvent)  
{  
    TPoint point = aPointerEvent.iPosition;  
    if(aPointerEvent.iType == TPointerEvent::EButton1Down)  
    {  
        if( HitRegionContains( point, *iLabel ))  
            iTactileFeedback->InstantFeedback(ETouchFeedbackBasic);  
    }  
    // Call base class HandlePointerEventL()  
    CCoeControl::HandlePointerEventL(aPointerEvent);  
}
```

- Implement `SizeChanged` for label control.

```
// -----  
// CTactileFeedbackAppView::SizeChanged()  
// Called by framework when the view size is changed.  
// -----  
//  
void CTactileFeedbackAppView::SizeChanged()  
{  
    iLabel->SetExtent( TPoint(0,0), iLabel->MinimumSize());  
}
```

- Definition for the following functions for Label control.

```
TInt CTactileFeedbackAppView::CountComponentControls() const  
{  
    return 1; // return number of controls inside this container  
}  
  
CCoeControl* CTactileFeedbackAppView::ComponentControl(TInt aIndex) const  
{  
    switch ( aIndex )  
    {  
        case 0:  
            return iLabel;  
    }
```

```
        default:  
            return NULL;  
    }  
}
```

- Make sure to delete **iLabel** in the destructor of the class.
- Do not delete the object **iTactileFeedback** of **MTouchFeedback**.

```
// -----  
// CTactileFeedbackAppView::~CTactileFeedbackAppView()  
// Destructor.  
// -----  
//  
CTactileFeedbackAppView::~CTactileFeedbackAppView()  
{  
    if(iLabel)  
        delete iLabel, iLabel = NULL;  
}
```

Useful functions

MTouchFeedback

- Instance()
- SetFeedbackArea()
- RemoveFeedbackArea()
- InstantFeedback()
- FeedbackEnabledForThisApp()

MCoeControlHitTest

- HitRegionContains()

Keywords

Headers

- #include <touchfeedback.h>
- #include <eiklabel.h>

Classes

- MTouchFeedback
- CEikLabel

- MCoeControlHitTest

Libraries

- touchfeedback.lib
- eikcoctl.lib

Example Application

- [TactileFeedback.zip](#)

Related links

- [A tour to the S60 Touch UI components](#)
- [Working with Toolbar API - S60 Touch UI](#)
- [Working with LongTapDetector API - S60 Touch UI](#)
- [Working with Stylus Pop-up Menu API - S60 Touch UI](#)
- [Working with Adaptive Search feature - S60 Touch UI](#)
- [Working with ChoiceList API - S60 Touch UI](#)
- [Working with Generic Button API - S60 Touch UI](#)
- [Working with SingleStyleTreeList with Hierarchical Lists API - S60 Touch UI](#)
- [Working with SingleColumnStyleTreeList with Hierarchical Lists API - S60 Touch UI](#)

Reference list

- S60 5th edition SDK help
- [S60 5th Edition C++ Developer's Library v1.0](#)