Tactile Feedback support in MIDlets in S60 5th Edition devices

Overview

S60 devices from S60 5th Edition onwards may include Touch UI-compliant screens. To eliminate the passive feeling of touch screens, touch-enabled devices may provide tactile feedback by utilizing the TactileFeedback class in Nokia UI API. Tactile feedback consists of little signs (audio, vibration) to accompany touch events such as dragging and tapping. When used correctly, they present a more responsive interaction experience for the end user.

Tactile feedback extensions work with both LCDUI and eSWT. In native applications and with most high-level UI components, tactile feedback is handled automatically.

Note: Tactile feedback does not work in MIDlets running on S60 5th Edition 0.9 SDK. Creating TactileFeedback class instances might make the MIDlet crash.

The tactile feedback in MIDlets is handled with one class of Nokia UI API, com.nokia.mid.ui.TactileFeedback. Tactile feedback can be generated in two ways:

- By producing direct feedback from the application, for example:

```
tactileFeedback.directFeedback(TactileFeedback.FEEDBACK_STYLE_BASIC);
```

- By adding feedback areas to area registry, in which case the feedback will be produced by the tactile feedback system automatically when the defined screen area (with defined feedback) is touched, for example:

```
tactileFeedback.registerFeedbackArea(this, 0, 0, getHeight()-40, 40, 40,
TactileFeedback.FEEDBACK_STYLE_BASIC);
```

The tactile feedback has 2 different styles: Basic and Sensitive. Each type is defined in the device settings, not by this API. In Nokia 5800 XpressMusic only Basic style feedback is given.

Source code: PointerMIDlet.java

```
import javax.microedition.lcdui.*;
import javax.microedition.midlet.*;

public class PointerMIDlet extends MIDlet {
    private PointerCanvas canvas;

    public PointerMIDlet() {
        canvas = new PointerCanvas(this);
    }

    protected void startApp() throws MIDletStateChangeException {
        Display.getDisplay(this).setCurrent(canvas);
    }

    protected void pauseApp() {
    }

    protected void destroyApp(boolean p1) throws MIDletStateChangeException {
    }
}
```

Source code: PointerCanvas.java

```
import javax.microedition.lcdui.*;
import com.nokia.mid.ui.TactileFeedback;

public class PointerCanvas extends Canvas implements CommandListener {
    private Command exitCommand;
    private Command clearCommand;
    private Command unregisterCommand;
    private PointerMIDlet midlet;
    private int width, height;
    protected int xcoord, ycoord;
    protected int xcoord_old, ycoord_old;
    protected boolean pressed = false;
    protected boolean dragged = false;
    protected boolean init = false;
    protected boolean touchFeedback = false;
    private TactileFeedback tactileFeedback;

    public PointerCanvas(PointerMIDlet midlet) {
        this.midlet = midlet;
        width = getWidth();
        height = getHeight();
        exitCommand = new Command ("Exit", Command.EXIT, 1);
        clearCommand = new Command ("Clear", "Clear Screen", Command.OK, 1);
        unregisterCommand = new Command("Unregister", "Unregister feedback areas",
            Command.OK, 2);
        this.addCommand(clearCommand);
        this.addCommand(unregisterCommand);
        this.addCommand(exitCommand);
```


(C) Copyright Nokia 2013. All rights reserved.

```
        this.setCommandListener(this);
        tactileFeedback = new TactileFeedback();
        touchFeedback = tactileFeedback.isTouchFeedbackSupported();
        if (touchFeedback) {
            try {
                tactileFeedback.registerFeedbackArea(this, 0, width-40, height-40, 40, 40,
                    TactileFeedback.FEEDBACK_STYLE_SENSITIVE);
                tactileFeedback.registerFeedbackArea(this, 1, 0, height-40, 40, 40,
                    TactileFeedback.FEEDBACK_STYLE_BASIC);
            }
            catch (IllegalArgumentException iae) {
                System.out.println("IllegalArgumentException: " + iae.getMessage());
            }
        }
    }

    public void commandAction(Command c, Displayable d) {
        if (c == clearCommand) {
            init = false;
            repaint();
        }
        else if (c == unregisterCommand) {
            tactileFeedback.unregisterFeedbackArea(this, 0);
            tactileFeedback.unregisterFeedbackArea(this, 1);
        }
        if (c == exitCommand) {
          midlet.notifyDestroyed();
        }
    }

    public void paint(Graphics g) {
        if (!init) {
            width = getWidth();
            height = getHeight();
            g.setColor(255, 255, 255);
            g.fillRect(0, 0, width, height);
            g.setColor(255, 0, 0);
            g.drawRect(0, 0, width - 1, height - 1);
            g.setColor(0, 0, 255);
            g.fillRect(width-40, height-40, 40, 40);
            g.setColor(255, 0, 0);
            g.fillRect(0, height-40, 40, 40);
            init = true;
        }
        g.setColor(0, 0, 0);
        if (pressed) {
            g.drawLine(xcoord, ycoord, xcoord+1, ycoord);
            g.drawLine(xcoord, ycoord+1, xcoord+1, ycoord+1);
        }
        if (dragged) {
            g.drawLine(xcoord_old, ycoord_old, xcoord, ycoord);
        }
        pressed = false;
        dragged = false;
    }

    /**
     * Called when the pointer is dragged.
     */
    protected  void pointerDragged(int x, int y) {
        dragged = true;
        xcoord_old = xcoord;
        ycoord_old = ycoord;
        xcoord = x;
        ycoord = y;
        repaint(xcoord-5, ycoord-5, 10, 10);
    }

    /**
     * Called when the pointer is pressed.
     */
    protected  void pointerPressed(int x, int y) {
        pressed = true;
        xcoord = x;
        ycoord = y;
        repaint();
    }

    /**
     * Called when the pointer is released.
     */
    protected  void pointerReleased(int x, int y) {
    }

    protected void keyPressed(int keyCode) {
    }

    protected void sizeChanged(int x, int y) {
        init = false;
        tactileFeedback.directFeedback(TactileFeedback.FEEDBACK_STYLE_BASIC);
        repaint();
    }

    protected void showNotify() {
    }

    protected void hideNotify() {
    }
}
```

You can tap on the red and blue areas in the lower corners of the screen for testing tactile feedback. Note also, that tactile feedback settings are part of profile settings in S60 devices. For example, by default there might be not feedback at all, if you are using offline profile.

See also

- TactileFeedback JavaDoc documentation in Nokia Developer Java ME Developer's Library

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.