Receiving key events in Java ME

This code snippet demonstrates how to handle key events using the MIDP 2.0 API.

Overview

Only some displayables can handle key events. In MIDP 2.0 those are:

- `CustomItem`
- `Canvas`
- `GameCanvas`

This example implements some `Canvas` class methods. In order to handle a key press, the `Canvas.keyPressed` method is implemented. This method is called by the framework when a 'key press' event occurs. The method takes the key code integer value as a parameter, which can be used for special handling.

The second implemented method is `Canvas.paint`. It is used for displaying the code and name of the pressed key.

Canvas can also handle 'key release', 'key repeat', 'pointer drag', 'pointer press', and 'pointer release' events. The methods for handling these events are:

- `keyReleased`
- `keyRepeated`
- `pointerDragged`
- `pointerPressed`
- `pointerReleased`

For more info, see API documentation.

Source file: MyCanvas.java

```
import javax.microedition.lcdui.Canvas;
import javax.microedition.lcdui.Font;
import javax.microedition.lcdui.Graphics;

/**
 * Canvas demostrates key handling procedure example.
 */
public class MyCanvas extends Canvas {
    /**
     * Array holds string displayed with key code and key name.
     */
    private String[] displayStrings;
    /**
     * Holds last pressed key code.
     */
    private int pressedKey;

    /**
     *
     */
    public MyCanvas() {
        pressedKey = 0;
        displayStrings = new String[2];
        displayStrings[0] = "Key code: ";
        displayStrings[1] = "Key: ";
    }
```


(C) Copyright Nokia 2014. All rights reserved.

```
    /**
     * From Canvas.
     * Method fills canvas draw area with grey color, draws a rectangle border
     * for a canvas and displays last presend key code and it's name.
     * @param g
     */
    protected void paint( Graphics g ) {
        // Setting up text font to default one.
        g.setFont( Font.getDefaultFont() );
        // Setting up Grey color as the current one.
        g.setColor( 0xC0C0C0 );
        // Filling the screen with the current color.
        g.fillRect( 0, 0, getWidth(), getHeight() );
        // Setting text and canvas border color.
        g.setColor( 0x000000 );
        // Draw a simple border for our canvas.
        g.drawRect( 0 + 2, 0 + 2, getWidth() - 4, getHeight() - 4 );
        // Changing pen position to where our key codes will be displayed
        g.translate( 4, getHeight() / 2 );
        // drawing a key code
        // If pressedKey variable holds '0' as a value no key code ot it's name
        // will be displayed then.
        if( pressedKey != 0) {
           // Drawing a string containing pressed key code
           g.drawString( displayStrings[0] + pressedKey,
                   0, 0, Graphics.TOP | Graphics.LEFT );
           // Drawing a string containing pressed key code name
           g.drawString( displayStrings[1] +
                   "'" + getKeyName( pressedKey ) + "'",
                   0, 20, Graphics.TOP | Graphics.LEFT );
        } else {
           g.drawString( "Press a key ... ", 0, 20, Graphics.TOP | Graphics.LEFT );
        }
        // moving cursor back to where it was
        g.translate( 0, - getHeight() / 2 );
    }
    /**
     * From Canvas.
     * Method saves code of a pressed key to the pressedKey variable.
     * @param keyCode
     */
    protected void keyPressed( int keyCode ) {
        pressedKey = keyCode;
        repaint();
    }
}
```

Source file: ReceivingKeyEvents.java

```
    /**
     * Reference to our Canvas implementation object.
     */
    private MyCanvas canvas;

    public ReceivingKeyEvents() {
```


(C) Copyright Nokia 2014. All rights reserved.

```
        display = Display.getDisplay( this );
        setupMainForm();
        setupCanvas();
    }
    /**
     * Instantiates a canvas variable with MyCanvas instance.
     */
    private void setupCanvas() {
        canvas = new MyCanvas();
        // Adding 'Back' softkey to be able to return to app's main form.
        canvas.addCommand( BACK_COMMAND );
        // Setting up our MIDlet class as canvas command listener.
        canvas.setCommandListener( this );
    }
    /**
     * From CommandListener.
     * Called by the system to indicate that a command has been invoked on a
     * particular displayable.
     * @param command the command that was invoked
     * @param displayable the displayable where the command was invoked
     */
    public void commandAction(Command command, Displayable displayable) {
        if (command == EXIT_COMMAND) {
            // Exit the MIDlet
            exit();
        } else if (command == EXECUTE_COMMAND) {
            // Bringing canvas to the foreground.
            display.setCurrent( canvas );
        } else if ( command == BACK_COMMAND ) {
            // Switching back from canvas to mainForm.
            display.setCurrent( mainForm );
        }
    }
```

Postconditions

When the MIDlet is started, the main form with a text field will be displayed. Press the 'Start' softkey and a canvas will be displayed, waiting for key inputs from the user. If you press one of the numpad buttons, the appropriate key code and name will be displayed on the canvas.

To get back to the main form, press the 'Back' softkey.

Supplementary material

This code snippet is part of the stub concept, which means that it has been patched on top of a template application in order to be more useful for developers. The version of the Java ME stub application used as a template in this snippet is v1.1.

- The patched, executable application that can be used to test the features described in this snippet is available for download at Media:ReceivingKeyEvents.zip.
- You can view all the changes that are required to implement the above-mentioned features. The changes are provided in unified diff and colour-coded diff (HTML) formats in Media:ReceivingKeyEvents.diff.zip.
- For general information on applying the patch, see Using Diffs.
- For unpatched stub applications, see Example app stubs with logging framework.

(C) Copyright Nokia 2014. All rights reserved.

(C) Copyright Nokia 2014. All rights reserved.