Creating CustomItem in Java ME

This article needs to be updated: If you found this article useful, please fix the problems below then delete the {{ArticleNeedsUpdate}} template from the article to remove this warning.

Reasons: trashedDev
(10 Aug 2012)
The full potential of this article isn't available on full-touch devices since this article needs to have a physical number pad.

Overview

This code snippet demonstrates how to create and display custom control. It consist of two files:

1. **CustomControl.java** - Inherited from `MIDlet`, displays the `MyItem` control.
2. **MyItem.java** - Inherited from `CustomItem`.

Custom contol class `MyItem` implements the method `keyPressed()`, which is responsible for storing the last key-pressed code. It also implements the `paint()` method, which displays the last key-pressed code inside the control.

Source file: CustomControl.java

```
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.Form;
import javax.microedition.midlet.MIDlet;


public class CustomControl extends MIDlet implements CommandListener{
    //Main form.

    private Form form;
    private Display display;
    private Command exitCommand;
    private MyItem myItem;

    /**
     * Constructor. Constructs the object and initializes displayables.
     */
    public CustomControl() {
        form = new Form("Custom Control MIDlet.");

        // Add command buttons
        exitCommand = new Command("Exit", Command.EXIT, 1);
        form.addCommand(exitCommand);
        form.setCommandListener(this);

        display = Display.getDisplay(this);
        display.setCurrent(form);

        myItem = new MyItem("Custom control");
        form.append(myItem);
    }

    /**
     * From CommandListener.
     * Called by the system to indicate that a command has been invoked on a
     * particular displayable.
     * @param cmd the command that was invoked
     * @param displayable the displayable where the command was invoked
     */
    public void commandAction(Command cmd, Displayable displayable) {
        if (cmd == exitCommand) {
            notifyDestroyed();
        }
    }

    /**
     * From MIDlet.
     * Called when the MIDlet is started.
     */
    public void startApp() {
        // No implementation required.
    }

    /**
     * From MIDlet.
     * Called to signal the MIDlet to enter the Paused state.
     */
    public void pauseApp() {
        // No implementation required.
    }

    /**
     * From MIDlet.
     * Called to signal the MIDlet to terminate.
     * @param unconditional whether the MIDlet has to be unconditionally
     * terminated
     */
    public void destroyApp(boolean unconditional) {
        // No implementation required.
    }
}
```


(C) Copyright Nokia 2013. All rights reserved.

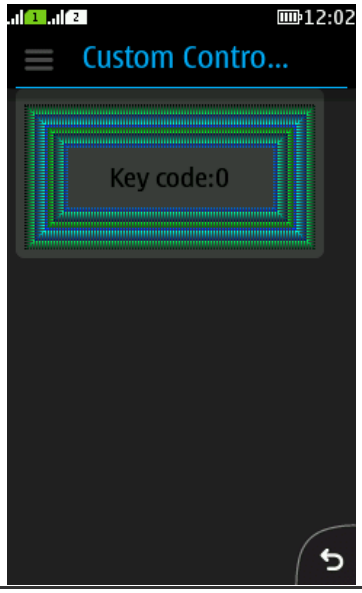
Source file: MyItem.java

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

class MyItem extends CustomItem {
    // Last key pressed code.
    private int key;

    /**
     * Constructor.
     */
    public MyItem( String label ){
        super( "" );
    }

    /**
     * From CustomItem.
     * Called by the system to redraw canvas.
     * @param graphics used for drawing operations.
     */
    public void paint(Graphics g, int width, int height) {
        //Display last key pressed.
        String output = "Key code:"+String.valueOf(key);

        g.drawString(output,(width-g.getFont().stringWidth(output))/2,
                            (height-g.getFont().getHeight())/2, 0);

        g.setStrokeStyle(Graphics.DOTTED);
        for (int i= 0;i<(height+width)/10;i++)
        {
            g.setColor(i*10000);
            g.drawRect( 0+i, 0+i, (width-i*2), (height-i*2));
        }
    }

    /**
     * From CustomItem.
     * Called by the system to retrieve minimum width required for this control.
     */
    protected int getMinContentWidth() {
        return 150;
    }

    /**
     * From CustomItem.
     * Called by the system to retrieve minimum height required for this control.
     */
    protected int getMinContentHeight() {
        return 100;
    }

    /**
     * From CustomItem.
     * Called by the system to retrieve preferred width for this control.
     */
    protected int getPrefContentWidth(int arg0) {
        return 200;
    }

    /**
     * From CustomItem.
     * Called by the system to retrieve preferred height for this control.
     */
    protected int getPrefContentHeight(int arg0) {
        return 100;
    }

    /**
     * From CustomItem.
     * Called by the system to redraw canvas.
     * @param graphics used for drawing operations.
     */
    protected void keyPressed(int keyCode) {
        key = keyCode;
        this.repaint();
    }
}
```

Postconditions

After starting the MIDlet, the user will see custom control with the last key-pressed code.

(C) Copyright Nokia 2013. All rights reserved.



Supplementary material

You can view the source file and executable application in the attached zip archive. The archive is available for download at Media:Creating custom control in J2ME.zip

Related articles:

- Archived:LCDUI CustomItem content remains blank when appended to a Form before setting the Form as current Displayable (Known Issue)
- CustomItem (Java Developers' Library)

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.