CS 683 Emerging Technologies
Fall Semester, 2004
Doc 24 An Example
Contents

References

J2ME in a Nutshell, Kim Topley, O'Reilly, 2002, chapter 5

Correction for Doc 21 slide 16

```
public void paint(Graphics g)
   {
   System.out.println("paint");
   int clipX = g.getClipX();
   int clipY = g.getClipY();
   int clipWidth = g.getClipWidth();
   int clipHeight = g.getClipHeight();

   g.setColor(WHITE);
   g.fillRect(clipX, clipY, clipWidth, clipHeight);

   g.setColor(BLACK);
   g.fillRect(boxX, boxY, boxLength, boxLength);
   }

protected void moveBox()
   {
   System.out.println("First repaint");
   repaint(boxX, boxY, boxLength, boxLength);
   boxX += 5;
   if (boxX > height)
      boxX = 0;
   boxY += 2;
   if (boxY > width)
      boxY = 0;
   System.out.println("Second repaint");
   repaint(boxX, boxY, boxLength, boxLength);
   }
```

Graphics Example

This example is from code provided with J2ME in a Nutshell, Kim Topley, 2002, O'Reilly

```
package ora.ch5;

import javax.microedition.lcdui.Canvas;
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.Font;
import javax.microedition.lcdui.Graphics;
import javax.microedition.lcdui.List;
import javax.microedition.midlet.MIDlet;

public class GraphicsMIDlet extends MIDlet implements CommandListener
{

    // The MIDlet's Display object
    private Display display;

    // Flag indicating first call of startApp
    protected boolean started;

    // Exit command
    private Command exitCommand;

    // Back to examples list command
    private Command backCommand;

    // The example selection list
    private List examplesList;
```

```
// The Canvases used to demonstrate different Items
private Canvas[] canvases;

// The example names. Used to populate the list.
private String[] examples = {
   "Lines", "Rectangles", "RectangleFills",
   "Arcs", "FilledArcs", "Text"
};

protected void startApp() {
   if (!started) {
      started = true;
      display = Display.getDisplay(this);

      // Create the common commands
      createCommands();

      // Create the canvases
      createCanvases();

      // Create the list of examples
      createList();

      // Start with the List
      display.setCurrent(examplesList);
   }
}

protected void pauseApp() {
}

protected void destroyApp(boolean unconditional) {
}
```

```
public void commandAction(Command c, Displayable d) {
   if (d == examplesList) {
      // New example selected
      int index = examplesList.getSelectedIndex();
      display.setCurrent(canvases[index]);
   } else if (c == exitCommand) {
      // Exit. No need to call destroyApp
      // because it is empty.
      notifyDestroyed();
   } else if (c == backCommand) {
      // Go back to main selection list
      display.setCurrent(examplesList);
   }
}

private void createCommands() {
   exitCommand = new Command("Exit", Command.EXIT, 0);
   backCommand = new Command("Back", Command.BACK, 1);
}

private void createList() {
   examplesList = new List("Select Example", List.IMPLICIT);
   for (int i = 0; i < examples.length; i++) {
      examplesList.append(examples[i], null);
   }
   examplesList.setCommandListener(this);
}
```

```
private void createCanvases() {
   canvases = new Canvas[examples.length];
   canvases[0] = createLinesCanvas();
   canvases[1] = createRectanglesCanvas();
   canvases[2] = createRectangleFillsCanvas();
   canvases[3] = createArcsCanvas();
   canvases[4] = createFilledArcsCanvas();
   canvases[5] = createTextCanvas();

}

private void addCommands(Displayable d) {
   d.addCommand(exitCommand);
   d.addCommand(backCommand);
   d.setCommandListener(this);
}

// Create the Canvas for the line drawing example
private Canvas createLinesCanvas() {
   Canvas canvas = new LineCanvas();
   addCommands(canvas);
   return canvas;
}

// Create the Canvas for the rectangles example
private Canvas createRectanglesCanvas() {
   Canvas canvas = new RectanglesCanvas();
   addCommands(canvas);
   return canvas;
}
```

```
    // Create the Canvas for the filled rectangles example
    private Canvas createRectangleFillsCanvas() {
       Canvas canvas = new RectangleFillsCanvas();
       addCommands(canvas);
       return canvas;
    }

    // Create the Canvas for the arcs example
    private Canvas createArcsCanvas() {
       Canvas canvas = new ArcsCanvas();
       addCommands(canvas);
       return canvas;
    }

    // Create the Canvas for the filled arcs example
    private Canvas createFilledArcsCanvas() {
       Canvas canvas = new FilledArcsCanvas();
       addCommands(canvas);
       return canvas;
    }

    // Create the Canvas for the text example
    private Canvas createTextCanvas() {
       Canvas canvas = new TextCanvas();
       addCommands(canvas);
       return canvas;
    }

}
```

```
// A canvas that illustrates line drawing
class LineCanvas extends Canvas {
   public void paint(Graphics g) {
      int width = getWidth();
      int height = getHeight();

      // Fill the background using black
      g.setColor(0);
      g.fillRect(0, 0, width, height);

      // White horizontal line
      g.setColor(0xFFFFFF);
      g.drawLine(0, height/2, width - 1, height/2);

      // Yellow dotted horizontal line
      g.setStrokeStyle(Graphics.DOTTED);
      g.setColor(0xFFFF00);
      g.drawLine(0, height/4, width - 1, height/4);

      // Solid diagonal line in brightest gray
      g.setGrayScale(255);
      g.setStrokeStyle(Graphics.SOLID);
      g.drawLine(0, 0, width - 1, height - 1);
   }
}
```

Rest of the Canvas classes not shown