Create a page flip effect in Java ME

This article explains how to create an attractive page flip transition effect in Java ME.

Note: This is an entry in the Asha Touch Competition 2012Q3

Introduction

Modern applications are *expected* to have a gorgeous, smooth & innovative user interface. Take for example the popular news/social reader app Flipboard, which allows users to turn pages using an attractive "page filipping" effect, similar to that of turning pages on a real book. This has allowed it to stand out from the crowd of other ebook readers, which often allow navigation only through simple scrolling or list drilling methods.

In this article I will show you how to recreate that same effect with Java ME on Nokia phones. You can see the below sample video for the results. The media player is loading...

Getting Started

Flipping is done with some 3D code to rotate the images around its Y or X axis (depending on if you flip horizontally or vertically) with some 2D cropping. A pre-requisite of this code is therefore that the phone has JSR-184.

This example uses 2 images and the two screens that are flipped between. To flip forward or backward you simply tap the right or left part of the touch screen respectively. Implementing a drag feature on the pages has been left as an exercise for the reader.

Note: A real app would typically display more than a single image, but we'd still implement the flipping using a single image as shown - we would draw the UI first to an image (`Graphics`) instead of directly to the Canvas, and then flip the image. This process is known as double buffering and results in better performance and appearance during the flipping effect.

The Code

This class is the only class (beside the MIDLET class) in the example, it inherits `GameCanvas`.

```
import javax.microedition.lcdui.Graphics;
import javax.microedition.lcdui.Image;
import javax.microedition.lcdui.game.GameCanvas;
import javax.microedition.m3g.Appearance;
import javax.microedition.m3g.Background;
import javax.microedition.m3g.Camera;
import javax.microedition.m3g.CompositingMode;
import javax.microedition.m3g.Graphics3D;
import javax.microedition.m3g.Image2D;
import javax.microedition.m3g.IndexBuffer;
import javax.microedition.m3g.Mesh;
import javax.microedition.m3g.PolygonMode;
import javax.microedition.m3g.Texture2D;
import javax.microedition.m3g.Transform;
import javax.microedition.m3g.TriangleStripArray;
import javax.microedition.m3g.VertexArray;
import javax.microedition.m3g.VertexBuffer;

/**
 *
 * @author ifrach shai
 *
 */
public class FlipboardCanvas extends GameCanvas
{
 // the 2d graphics object
 Graphics graphics;
 // the 3d graphics object
 private Graphics3D g3D;
 // the camera object
 private Camera camera;
 // members to keep track of the current page the user is viewing and if we are currently in transition mode
 int currentPage;
 boolean actionCompleted;

    // the current angle of the effect
    private int angle = 0;
    // the increment we add to angle every frame
    private int delta;

    // 3d related objects
    private Background background = new Background();
    Transform identity = new Transform();
 private Mesh firstPageMesh, secondPageMesh;
 // 2d images
 private Image firstPageImage, secondPageImage,firstPageStretch, secondPageStretch, shadow;

 public FlipboardCanvas()
 {
  super(true);
  setFullScreenMode(true);
  graphics = getGraphics();
  try {
```


(C) Copyright Nokia 2013. All rights reserved.

```
    init();
   }catch (Exception e) {
    e.printStackTrace();
    System.out.println(e.getMessage());
   }
  }

  /**
   * init a worker thread to continuely call paint
   */
  public void startRendering()
  {
   new Thread(new Runnable() {

    public void run()
    {
     while (true)
      paint(graphics);
    }
   }).start();
  }

  /**
   * helper function to perform image resize which j2me lacks
   * @param img - the images to resize
   * @param a - the width of the resize destination
   * @param b - the height of the resize destination
   * @return the resized image
   */
  public Image resize(Image img, int a, int b) // a - destination width, b - destination height
     {
         if(a == img.getWidth() && b == img.getHeight())
             return img;
         if(a <= 0 || b <= 0)
             return img;
         int ai[] = new int[img.getWidth()*img.getHeight()];
         img.getRGB(ai, 0, img.getWidth(), 0, 0, img.getWidth(), img.getHeight());
         int ai1[] = new int[a*b];
         int i = a;
         int j = img.getWidth();
         int m;
         int k1 = (int)(((double)(m = img.getHeight()) / (double)b) * 1024D);
         int l1 = (int)(((double)j / (double)a) * 1024D);
         int i2 = 0 - i;
         for(int k = 0; k < b; k++)
         {
             int l = (k1 * k >> 10) * j;
             i2 += i;
             int j1 = 0 - l1;
             for(m = 0; m < a; m++)
             {
                 int i1 = (j1 += l1) >> 10;
                 ai1[i2 + m] = ai[l + i1];
             }

         }
         return Image.createRGBImage(ai1, a, b, true);
     }

  /**
   * listen to the user tap/click interaction and decide wether to activate flip and to which direction
   */
  public void pointerPressed(int x, int y)
  {
   if (!actionCompleted)
    return;
   if (x>getWidth()/2)
   {
    if (currentPage == 1)
     return;
    angle = 0;
    actionCompleted = false;
    delta = 4;
   }
   else
   {
    if (currentPage == 0)
     return;
    angle = 180;
    actionCompleted = false;
    delta = -4;
   }
  }

  /**
   * this function loads the images from the jar and initialize the 3d related objects
   * @throws Exception
   */
     private void init() throws Exception  {

      actionCompleted = true;
      firstPageImage = Image.createImage("/first.jpg");
      secondPageImage = Image.createImage("/second.jpg");
      shadow = Image.createImage("/shadow.png");
      firstPageStretch = resize(firstPageImage,getWidth(),getHeight());
      secondPageStretch = resize(secondPageImage,getWidth(),getHeight());
      currentPage = 0;
         g3D = Graphics3D.getInstance();
         g3D.releaseTarget();
         camera = new Camera();
         camera.setPerspective( 30.0f, 1, 1.0f, 45.0f );
         background.setColor(0x00ffffff);
         firstPageMesh = createMeshPage(firstPageImage);
         secondPageMesh = createMeshPage(secondPageImage);
     }
     /**
      * turns an image to a simple rect mesh composed of 2 triables and the image as the texture
      * @param image
      * @return
      */
     private Mesh createMeshPage(Image image) {
      Texture2D texture = new Texture2D(new Image2D(Image2D.RGBA, image));
         texture.setFiltering(Texture2D.FILTER_NEAREST, Texture2D.FILTER_NEAREST);
         texture.setWrapping(Texture2D.WRAP_CLAMP, Texture2D.WRAP_CLAMP);
         texture.setBlending(Texture2D.FUNC_REPLACE);
         CompositingMode cm = new CompositingMode();
         cm.setBlending(CompositingMode.ALPHA);
         cm.setAlphaWriteEnable(true);
         cm.setDepthTestEnable(true);
         Appearance appearance = new Appearance();
```


(C) Copyright Nokia 2013. All rights reserved.

```
        PolygonMode polyMode = new PolygonMode();
        polyMode.setPerspectiveCorrectionEnable(true);
        appearance.setPolygonMode(polyMode);
        appearance.setCompositingMode(cm);
        appearance.setTexture(0, texture);
        short[] verts = { -1, -1, 0, 1, -1, 0, 1, 1, 0, -1, 1, 0 };

        VertexArray va = new VertexArray(verts.length / 3, 3, 2);
        va.set(0, verts.length / 3, verts);
        short[] tcs = { 0, 1, 1, 1, 1, 0, 0, 0 };

        VertexArray texArray = new VertexArray(tcs.length / 2, 2, 2);
        texArray.set(0, tcs.length / 2, tcs);

        VertexBuffer vb = new VertexBuffer();
        vb.setPositions(va, 1.0f, null);
        vb.setTexCoords(0, texArray, 1.0f, null);
        int[] indicies = {1,2,0,3};
        int[] stripLens = {4};
        IndexBuffer ib = new TriangleStripArray(indicies, stripLens);
        return new Mesh(vb, ib, appearance);
    }

    /**
     * The Paint method is the heart of the operation in here we check if we are in a flip effect mode
     * and decide how to paint the screen.
     * We also increment the angle of the flip effect inside the paint method
     */
    public void paint(Graphics g) {

     g.setClip(0, 0, getWidth(), getHeight());
     // painting the correct screen as the background
     switch (currentPage)
  {
  case 0:
   g.drawImage(firstPageStretch, 0, 0, Graphics.TOP | Graphics.LEFT);
   break;
  case 1:
   g.drawImage(secondPageStretch, 0, 0, Graphics.TOP | Graphics.LEFT);
   break;
  }

     if (!actionCompleted)
     {
      // in flip effect mode
      switch (currentPage)
   {
   case 0:
    if (angle >= -90)
     try {
      //clip the 2d screen in order to see just half of the page flip
      g.setClip(getWidth()/2, 0, getWidth()/2, getHeight());
      g3D.bindTarget(g, true, Graphics3D.DITHER | Graphics3D.TRUE_COLOR);
      g3D.setViewport(0, 0, getWidth(), getHeight());
      g3D.clear(background);
      // draws the other page behind the current flipping side
      g.drawImage(secondPageStretch, 0, 0, Graphics.TOP | Graphics.LEFT);
      g3D.setCamera(camera, identity);
      Transform transform = new Transform();
      transform.setIdentity();
      transform.postTranslate(0, 0, -3.8f);
      transform.postRotate(angle, 0, 1, 0);
      // rotating and rendering the current page
      g3D.render(firstPageMesh, transform);
     }
    catch (Exception e) {}
    finally {
     g3D.releaseTarget();
    }
    else
     try {
      g.setClip(0, 0, getWidth()/2, getHeight());
      g3D.bindTarget(g, true, Graphics3D.DITHER | Graphics3D.TRUE_COLOR);
      g3D.setViewport(0, 0, getWidth(), getHeight());
      g3D.clear(background);
      g.drawImage(firstPageStretch, 0, 0, Graphics.TOP | Graphics.LEFT);
      g.setClip(getWidth()/2, 0, getWidth()/2, getHeight());
      g.drawImage(secondPageStretch, 0, 0, Graphics.TOP | Graphics.LEFT);
      g.setClip(0, 0, getWidth()/2, getHeight());
      g3D.setCamera(camera, identity);
      Transform transform = new Transform();
      transform.setIdentity();
      transform.postTranslate(0, 0, -3.8f);
      transform.postRotate(angle+180, 0, 1, 0);
      g3D.render(secondPageMesh, transform);
     }
    catch (Exception e) {}
    finally {
     g3D.releaseTarget();
    }
    break;
   case 1:
    if (angle <= 270)
     try {
      g.setClip(0, 0, getWidth()/2, getHeight());
      g3D.bindTarget(g, true, Graphics3D.DITHER | Graphics3D.TRUE_COLOR);
      g3D.setViewport(0, 0, getWidth(), getHeight());
      g3D.clear(background);
      g.drawImage(firstPageStretch, 0, 0, Graphics.TOP | Graphics.LEFT);
      g3D.setCamera(camera, identity);
      Transform transform = new Transform();
      transform.setIdentity();
      transform.postTranslate(0, 0, -3.8f);
      transform.postRotate(angle-180, 0, 1, 0);
      g3D.render(secondPageMesh, transform);
     }
    catch (Exception e) {}
    finally {
     g3D.releaseTarget();
    }
    else
     try {
      g.setClip(getWidth()/2, 0, getWidth()/2, getHeight());

      g3D.bindTarget(g, true, Graphics3D.DITHER | Graphics3D.TRUE_COLOR);
      g3D.setViewport(0, 0, getWidth(), getHeight());
      g3D.clear(background);
      g.drawImage(secondPageStretch, 0, 0, Graphics.TOP | Graphics.LEFT);
      g.setClip(0, 0, getWidth()/2, getHeight());
      g.drawImage(firstPageStretch, 0, 0, Graphics.TOP | Graphics.LEFT);
```


(C) Copyright Nokia 2013. All rights reserved.

```
      g.setClip(0, 0, getWidth()/2, getHeight());
      g3D.setCamera(camera, identity);
      Transform transform = new Transform();
      transform.setIdentity();
      transform.postTranslate(0, 0, -3.8f);
      transform.postRotate(angle, 0, 1, 0);
      g3D.render(firstPageMesh, transform);
     }
    catch (Exception e) {}
    finally {
     g3D.releaseTarget();
    }
    break;
   }
      // increment the angle to continue the flipping effect
      angle -= delta;
      // checking if the flipping has reached its end
      if ((angle < -180 && delta > 0) || (angle > 360 && delta < 0))
      {
       actionCompleted = true;
       switch (currentPage)
       {
       case 0:
        currentPage = 1;
        break;
       case 1:
        currentPage = 0;
        break;
       }
      }
      // draws a cute shadow image in the center to add to the effect, this can be removed if the result isnt to your liking
      g.drawImage(shadow, getWidth()/2-shadow.getWidth()/2, 0, Graphics.TOP | Graphics.LEFT);
     }
     // flush the graphics to the screen
        flushGraphics();
    }

}
```

The code isn't that complicated and full of comments to explain each part.

I tested this effect on both Nokia SDK for Java 2.0 (beta) Emulator and a real handset (I set the Nokia JADa parameter `Nokia-MIDlet-App-Orientation: landscape` because it has more space to flip in landscape mode)

The effect on the emulator is slow but on the device it is more than acceptable (as seen in the video in the introduction). Note that you can control the delta member of the above class to change the speed/smoothness of the effect.

Summary

I hope this article help you to think of ways to spice up your UI to retain and engage users. Feel free to download the attached binary or eclipse project source to test it for yourself.

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.