Memory leak repeatedly adding elements with Image in List item, "refreshing List" (J2ME, S60)

Overview

Applications using a MIDP 2.0 List item using elements with Images to fill the List repeatedly will cause a Memory Leak and thus will force the application to shut off.

Update

After some testing on the Nokia 6233 (Series 40) I noticed that even there the memory usage keeps incrementing, the difference though is that the Garbage COllector does it's job just before going OutOfMemory, which might indicate a problem with the S60's Garbace Collector's implementation ???

Description

When using a List, with several elements (25+), each element using the SAME Image, and the List gets repeatedly refreshed the used memory of the application will keep increasing in S60 devices.

How to reproduce

- Create a data Source (for example a Vector)
- Fill it with some data, preferably something like 25+ items (Strings for example)
- Afterwards create one instance of an Image
- Fill the List with the Vectors data and the Image (i.e : List.append(String, Image);)
- Repeatedly Fill the List by looping through last point.
- Memory will keep increasing only in S60 devices (tested on a Nokia 6233 Series 40 and will run indefinitely just fine)

Source code :

testList.java

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

/**
 * @author Tiger79
 */
public class testList extends MIDlet implements CommandListener
{
    private Display mDisplay;
    private Form myForm;
    private Command exitCmd, showCmd;
    private theList myList;
    private StringItem myString, myFreeMemory, myUsedMemory;

    public void startApp()
    {
        mDisplay = Display.getDisplay(this);

        myList = new theList(this);

        myForm = new Form("Little test");

        exitCmd = new Command("Exit", Command.EXIT, 0);
        showCmd = new Command("Show List", Command.SCREEN, 0);

        myForm.addCommand(exitCmd);
        myForm.addCommand(showCmd);

        myForm.setCommandListener(this);

        myString = new StringItem("Loops : ", "");
        myFreeMemory = new StringItem("Free Memory : ", "");
        myUsedMemory = new StringItem("Used Memory : ", "");

        myForm.append(myString);
        myForm.append(myFreeMemory);
        myForm.append(myUsedMemory);
```


(c) Copyright Nokia 2010. All rights reserved.

```
        showMidlet();
    }

    public void pauseApp()
    {
    }

    public void destroyApp(boolean unconditional)
    {
    }

    public void commandAction(Command c, Displayable d)
    {
        if(c == exitCmd)
            this.notifyDestroyed();
        else if(c == showCmd)
            mDisplay.setCurrent(myList);

    }

    public void setValue(int passedValue)
    {
        myString.setText(String.valueOf(passedValue));
        myFreeMemory.setText(String.valueOf(Runtime.getRuntime().freeMemory()/1024));
        long temp = Runtime.getRuntime().totalMemory() - Runtime.getRuntime().freeMemory();
        myUsedMemory.setText(String.valueOf(temp/1024));

    }

    public void showMidlet()
    {
        mDisplay.setCurrent(myForm);
    }
}
```

theList.java

```
import java.util.Timer;
import java.util.TimerTask;
import java.util.Vector;
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.Image;
import javax.microedition.lcdui.List;


/**
 *
 * @author Tiger79
 */
public class theList extends List implements CommandListener
{
    private Vector myVector;    //Vector with Strings to be shown in List
    private Image myImage;      //the culprit : an Image, just instanced once
    private Timer fillTimer = new Timer();  //Timer to perform a refresh of the List every x seconds
    private fillTask fillItTask = new fillTask();   //the task Class which performs the refresh
    public int counter = 0;     //counter that counts how many refreshes have been done
    private testList myMidlet;  //callback to Midlet
    private Command backCmd;

    public theList(testList mMidlet)
    {
       super("testing", List.IMPLICIT);
       myMidlet = mMidlet;

       myVector = new Vector();

       try
       {
           myImage = Image.createImage("/littleImage.png");
       }
       catch(Exception ex)
       {
           System.out.print("Problem creating Image : " + ex.toString());
       }

       backCmd = new Command("Back", Command.BACK, 0);
       this.addCommand(backCmd);
       setCommandListener(this);

       for(int i = 0 ; i < 25; i ++)
           myVector.addElement( i + " - TestData, just some testData to test the testing thingie");

       fillTimer.scheduleAtFixedRate(fillItTask, 0, 1000);

    }

    private void fillList()
    {
        this.deleteAll();
        for(int i = 0; i < myVector.size(); i ++)
            append((String)myVector.elementAt(i), myImage);
        counter++;
        myMidlet.setValue(counter);
    }

    private class fillTask extends TimerTask
    {
        public final void run()
        {
            fillList();
        }
    }

    public void commandAction(Command c, Displayable d)
    {
        if(c == backCmd)
        {
            myMidlet.showMidlet();
        }
    }
}
```


(C) Copyright © Nokia 2013. All rights reserved.

- This Image can be used :

Solution

just don't refresh that List item very often ;)

Test Results

- Nokia N95 : around 615 refreshes
- Nokia N91 : around 197 refreshes
- Nokia 6233 (Series 40) : I stopped etsting after 3700 refreshes, no problems here

Reference

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.