Taking a picture in Java ME

Overview

This code snippet describes how to capture an image from camera on Nokia devices which support Mobile Media API (JSR-135) and using the API for image capture. It shows how to take control over camera for a Java ME application by checking first if a device has a camera and then displaying the viewfinder of camera.

In addition, the code snippet shows how a camera snapshot application can be implemented on Series 40, S60 and Symbian devices. Since Series 40, S60 and Symbian support different locators for image capture (capture://image for Series 40, capture://video for S60 and Symbian), the code snippet provides a method for determining the device platform first before setting the platform specific locator for image snapshot. The method is based on checking the supported image encodings for snapshot: png and image/bmp encodings are both supported by S60 and Symbian devices, whereas Series 40 devices do not support these encodings.

Device Requirements

This example can be tested on Series 40 and Symbian devices which support Mobile Media API (JSR-135).

For more information about Java ME support on Nokia device models, please visit the device specifications page on Nokia Developer web site here: Device specifications on Nokia Developer web site . The Java ME support including the supported Java APIs can be checked for each device by clicking the 'APIs' tab when viewing a device specification.

Source file: CaptureMidlet.java

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

import javax.microedition.media.Manager;
import javax.microedition.media.Player;
import javax.microedition.media.MediaException;
import javax.microedition.media.control.VideoControl;

import java.io.IOException;

public class CaptureMIDlet extends MIDlet implements CommandListener {

    private Display display;
    // Command for capturing image by camera. Placed in cameraForm.
    private Command cmdCapture;
    // Command for exiting from midlet. Placed in cameraForm.
    private Command cmdExit;
    // Command for returning from imageForm to cameraForm.
    // Placed in imageForm.
    private Command cmdBack;
    // Player for camera
    private Player player;
    // Video control of camera
    private VideoControl videoControl;
    // Form where captured image is placed
    private Form imageForm;
    // Image captured by camera.
    private Image capturedImage;
    // Alert to be displayed if error occurs.
    private Alert alert;
    // String for capture locator
    private String captureLocator;

    /**
     * Constructor.
     */
    public CaptureMIDlet() {

        display = Display.getDisplay(this);

        if (checkCameraSupport() == false) {
            showAlert("Error", "Camera is not supported!", null);
            return;
        }

        /**
         * If encodings of bmp and image/bmp are not supported for image capture
         * with camera, then the image capture will be done with the locator for
         * Series 40: capture://image
      *
         */
        if (checkEncodingSupport() == false) {

            captureLocator = "capture://image";

        }

        /**
         * If encodings of bmp and image/bmp are supported for image capture
         * with camera, then the capture will be done with the locator for S60
         * and Symbian: capture://video
```


(C) Copyright Nokia 2013. All rights reserved.

```
     *
      */
        if (checkEncodingSupport() == true) {

            captureLocator = "capture://video";
        }

                createCamera();
                createImageForm();
    }

    /**
     * Check camera support.
     * @return true if camera is supported, false otherwise.
     */
    private boolean checkCameraSupport() {
        String propValue = System.getProperty("supports.video.capture");
        return (propValue != null) && propValue.equals("true");
    }

    /**
     * Checking encodings for camera snapshot.
     * S60 and Symbian devices support png and image/bmp encodings whereas Series 40 devices don't support them.
     * The encoding support is used for determining the platform specific capture locator for camera snapshot.
     * @return
     */
    private boolean checkEncodingSupport() {
        String encodings = System.getProperty("video.snapshot.encodings");
        return (encodings != null) && (encodings.indexOf("png") != -1) && (encodings.indexOf("image/bmp") != -1);
    }

    private void createCamera() {

        // Create camera form
        Form cameraForm = new Form("Camera");

        // Create commands for this form
        cmdCapture = new Command("Capture", Command.OK, 0);
        cmdExit = new Command("Exit", Command.EXIT, 0);
        // Add commands to form
        cameraForm.addCommand(cmdCapture);
        cameraForm.addCommand(cmdExit);
        // Set midlet as command listener for this form
        cameraForm.setCommandListener(this);

        try {

            player = Manager.createPlayer(captureLocator);
            player.realize();
            player.prefetch();

            videoControl = (VideoControl) player.getControl("VideoControl");

            Item item = (Item) videoControl.initDisplayMode(VideoControl.USE_GUI_PRIMITIVE, null);
            cameraForm.append(item);
            player.start();

        } catch (IOException ex) {

            showAlert("IOException", ex.getMessage(), null);

        } catch (MediaException ex) {

            showAlert("MediaException", ex.getMessage(), null);
        }
        display.setCurrent(cameraForm);

    }

     /**
     *  Creates form where the captured image will be placed.
     */
    private void createImageForm() {
        // Create camera form
        imageForm = new Form("Captured image");
        // Create commands for this form
        cmdBack = new Command("Back", Command.BACK, 0);
        // Add commands to form
        imageForm.addCommand(cmdBack);
        // Set midlet as command listener for this form
        imageForm.setCommandListener(this);
    }

    /**
     * Captures image data from camera in the default format
     * and creates image from it.
     */
    private boolean captureImage() {
        try {
            byte[] imageData;

            imageData = videoControl.getSnapshot(null);

            capturedImage = Image.createImage(imageData, 0, imageData.length);

        } catch (MediaException exc) {

            showAlert("MediaException", exc.getMessage(), null);

            return false;

        } catch (SecurityException secExc) {

            showAlert("SecurityException", secExc.getMessage(), null);

            return false;

        }
        player.close();
        player = null;
        videoControl = null;

        return true;
    }

    /**
     * Places captured image as item on the image form
     * and displays form.
     */
    private void showCapturedImage() {
```


(C) Copyright Nokia 2013. All rights reserved.

```
        //imageForm.deleteAll();
        if (capturedImage != null) {
            imageForm.append(capturedImage);
        }

        display.setCurrent(imageForm);
    }

    /**
     * From MIDlet.
     * Signals the MIDlet that it has entered the Active state.
     */
    public void startApp() {


    }

    /**
     * From MIDlet.
     * Signals the MIDlet to enter the Paused state.
     */
    public void pauseApp() {
        // TODO: pause player if it is running.
    }

    /**
     * From MIDlet.
     * Performs exit from midlet.
     */
    public void exitMIDlet() {
        notifyDestroyed();
    }

    /**
     * From MIDlet.
     * Signals the MIDlet to terminate and enter the Destroyed state.
     */
    public void destroyApp(boolean unconditional) {
        if (player != null) {
            player.deallocate();
            player.close();
        }
    }

    /**
     * Shows alert with specified title and text. If next displayable is not
     * specified then application will be closed after alert closing.
     * @param title - Title of alert.
     * @param message - text of alert.
     * @param nextDisp - next displayable. Can be null.
     */
    private void showAlert(String title, String message, Displayable nextDisp) {
        alert = new Alert(title);
        alert.setString(message);
        alert.setTimeout(Alert.FOREVER);

        if (nextDisp != null) {
            display.setCurrent(alert, nextDisp);
        } else {
            display.setCurrent(alert);
            alert.setCommandListener(this);
        }
    }

    /**
     * From CommandListener.
     * Indicates that a command event has occurred on Displayable displayable.
     * @param command - a Command object identifying the command.
     * @param displayable - the Displayable on which this event has occurred.
     */
    public void commandAction(Command command, Displayable displayable) {
        // Handles "Capture image" command from cameraForm
        if (command == cmdCapture) {
            if (captureImage() == true) {
                showCapturedImage();
            }
        }
        // Handles "back" command from imageForm
        if (command == cmdBack) {
            createCamera();

        }
        // Handles "exit" command from forms
        if (command == cmdExit) {
            exitMIDlet();
        }
        // Handle "ok" command from alert
        if (displayable == alert) {
            exitMIDlet();
        }
    }
}
```

Postconditions

When the MIDlet starts, the viewfinder of the camera is displayed on a form.

To make capture, press "Capture image". The captured image will be displayed on a form. To return to viewfinder, press "Back".

Supplementary material

Executables and source files are available in File:TakingPictureV2.zip.

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.