High Level UI using ChoiceGroup

Version 0.2, Draft

INFORMATION GUIDE

COPYRIGHT

Trademarks and Service Marks

About This Document

This document gives an overview of High Level Item class ChoiceGroup and provides a sample code snippet explaining the implementation of class ChoiceGroup on class Form.

Scope:

This document is intended for Java ME developers wishing to develop mobile Java applications. It assumes good knowledge of java programming language.

Document History:

Date	Version	Comment
04/02/09	0.2	Draft

Reference:

1. MIDP 2.0 Specification:

   http://jcp.org/en/jsr/detail?id=118

Abbreviations:

MIDP	Mobile Information Device Profile
UI	User Interface

Table of Contents

Table of Figures

Introduction

Mobile Information Device Profile (MIDP) package *javax.microedition.lcdui* has a number of High Level User Interface (UI) components. These UI components are referred as Item since they extend *javax.microedition.lcdui.Item* class. One of the UI components in MIDP package is ChoiceGroup. This document describes about ChoiceGroup and its importance in developing a MIDlet application.

Overview

The *javax.microedition.lcdui.ChoiceGroup* is an Item, which allows user to choose between different elements in a group. These elements consist of simple string, user can also include image per element as well.

ChoiceGroup can be created in two different modes:

Single Choice: In this mode, only one element can be selected at a time from the group.

Multiple Choice: In this mode, one or more than one element can be selected at a time from the group.
ChoiceGroup implements *javax.microedition.lcdui.Choice* interface to differentiate between these two modes. This is done by providing the following Choice types:

EXCLUSIVE

The EXCLUSIVE choice type is used to select only one element at a time. By default, one element is selected at any given time. ChoiceGroup implementation uses Radio Buttons for visual representation to the user. Figure 1 shows the EXCLUSIVE type ChoiceGroup.

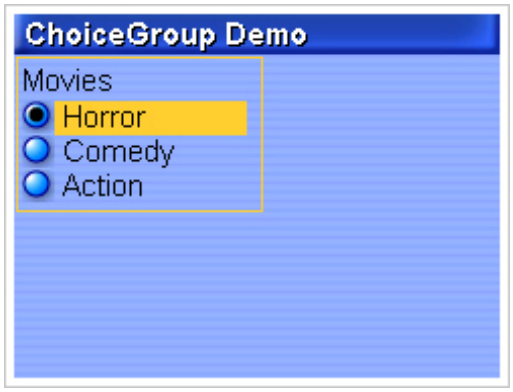

Figure 1: Exclusive ChoiceGroup

Applications for which the selected element is significant should set the selection explicitly. There is no way for the user to deselect an element within an EXCLUSIVE Choice.

MULTIPLE

The MULTIPLE choice is used to select one or more than one element in any combination. ChoiceGroup implementation uses Checkbox for visual representation to the user. Figure 2 shows MULTIPLE type ChoiceGroup.

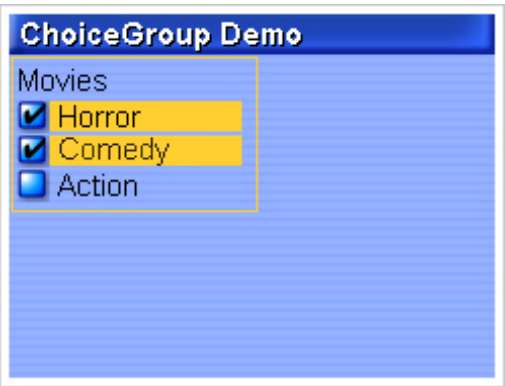

Figure 2: Multiple ChoiceGroup

POPUP

The POPUP choice is similar to the EXCLUSIVE choice. The only difference is POPUP uses different visual representation and interaction. POPUP choice is implemented using popup menu (drop down) in which only the selected element is displayed and other elements remain hidden until user interacts with this type ChoiceGroup. On user interaction, the hidden elements are shown along with the selected element. Figure 3 shows the POPUP type ChoiceGroup.

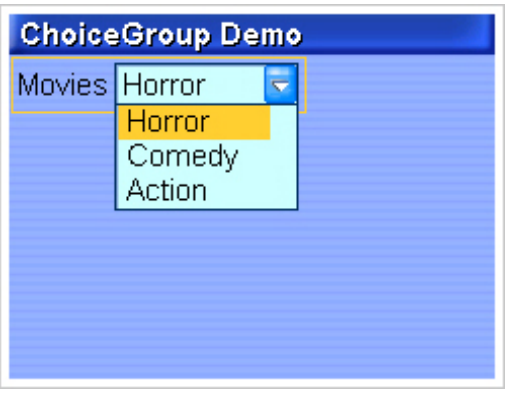

Figure 3: Popup ChoiceGroup

Creating ChoiceGroup

ChoiceGroup can be created using one of the two constructors provided by the ChoiceGroup class.

```
ChoiceGroup (String label, int choiceType)
```

The ChoiceGroup constructor requires at least a label and a type value. Additionally, a String array and an Image array containing the elements can be passed to the constructor.

```
ChoiceGroup(String label, int choiceType, String[] stringElements, Image[] imageElements)
```

where

label -- indicates the ChoiceGroup name.

choiceType -- this variable specifies the Choice for ChoiceGroup; EXCLUSIVE, MULTIPLE, or POPUP.

stringElements -- set of strings specifying the string parts of the ChoiceGroup elements.

imageElements -- set of images specifying the image parts of the ChoiceGroup elements.

ChoiceGroup Operations

Some of the important ChoiceGroup operations are as follows:

- Element can be added to the ChoiceGroup using *append(String stringpart, Image imagepart)* where stringpart is compulsory and imagepart can be null if no image is required to be added to the element.
- Element can be inserted using *insert(int index, String stringpart, Image imagepart)* where stringpart and imagepart are same as adding an element. Index signifies the index of the element where insertion should occur.
- Element can be set using *set(int index, String stringpart, Image imagepart)* where stringpart and imagepart are same as inserting an element.
- Element can de deleted using *delete(int elemNum)* where elemNum signifies the index of the element to be deleted. All elements from ChoiceGroup can be deleted using *deleteAll()* method.
- Element can be retrieved using *getString(int elemNum*) , *getImage(int elemNum)* to get String and Image respectively by passing the elemNum. Similarly index of an element can be retrieved using *getSelectedIndex().*

- Element can be selected using *setSelectedIndex(int elemNum, boolean selected)* where elemNum refers to the index. For ChoiceGroup objects of type MULTIPLE, this simply sets an individual element's selected state. For ChoiceGroup objects of type EXCLUSIVE and POPUP, this can be used only to select an element. That is, the selected parameter must be true. When an element is selected, the previously selected element is deselected. If selected one is false, this call is ignored.

- To check whether an element is selected *isSelected(int elemNum)* can be used.

- All elements present in a ChoiceGroup can be determined using *size()* method.

Sample code snippet for the ChoiceGroup

The sample code given below shows how to use ChoiceGroup class:

Class: ChoiceGroupMidlet.java

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

public class ChoiceGroupMidlet extends MIDlet implements CommandListener {

   private static final String EXIT = "Exit";
   private static final String CHOOSE = "Show";

   /*The CMD_EXIT Command to Exit the Midlet*/
   private final Command CMD_EXIT = new Command(EXIT, Command.EXIT, 1);

   /*The CMD_CHOOSE command to show the selected elements*/
   private final Command CMD_CHOOSE = new Command(CHOOSE, Command.SCREEN, 2);
   private Display display = null;
   private Form form = null;
   private ChoiceGroup choiceGroup = null;
   private final String MOVIES[] = {"Horror", "Comedy", "Action"};
   private boolean isDone;
   private int itemIndex;

   public ChoiceGroupMidlet() {
      init();
```

```
    }

    public void startApp() {
        if (!isDone) {
            form.addCommand(CMD_EXIT);
            form.addCommand(CMD_CHOOSE);
            itemIndex = form.append(choiceGroup);
            isDone = true;
        }
        form.setCommandListener(this);
        display.setCurrent(form);
    }

    public void pauseApp() {
    }

    public void destroyApp(boolean flag) {
    }

    public void commandAction(Command cmd, Displayable dis) {
        if (cmd == CMD_EXIT) {
            destroyApp(true);
            notifyDestroyed();
        } else if (cmd == CMD_CHOOSE) {
            for (int i = 0; i < choiceGroup.size(); i++) {
                if (choiceGroup.isSelected(i)) {
                    StringItem selectedItem = new StringItem(null, choiceGroup.getString(i));
                    form.append(selectedItem);
                }
            }
            form.delete(itemIndex);
            form.removeCommand(CMD_CHOOSE);
            form.setTitle("Selected Items");
        }
    }

    /*This method is used for initializing the object*/
    public void init() {
        display = Display.getDisplay(this);
        form = new Form("ChoiceGroup Demo");
        choiceGroup = new ChoiceGroup("Movies", Choice.MULTIPLE, MOVIES, null);
    }
}
```