Lab Exercise 3

Please follow the instruction in Workshop Note 3 to complete this exercise.

1. Turn on your computer and startup **Windows XP** (*English version*), download and install **J2ME Wireless Toolkit 2.2** to your computer. Then download the laboratory resource files from http://www.peter-lo.com/Teaching/I123-1-A/Source3.zip.

2. Create a MIDlet to display a string item on the Form. Then modify it to allow user input "Username" and "Password", and display it on screen. (*Page 1 – 4*)

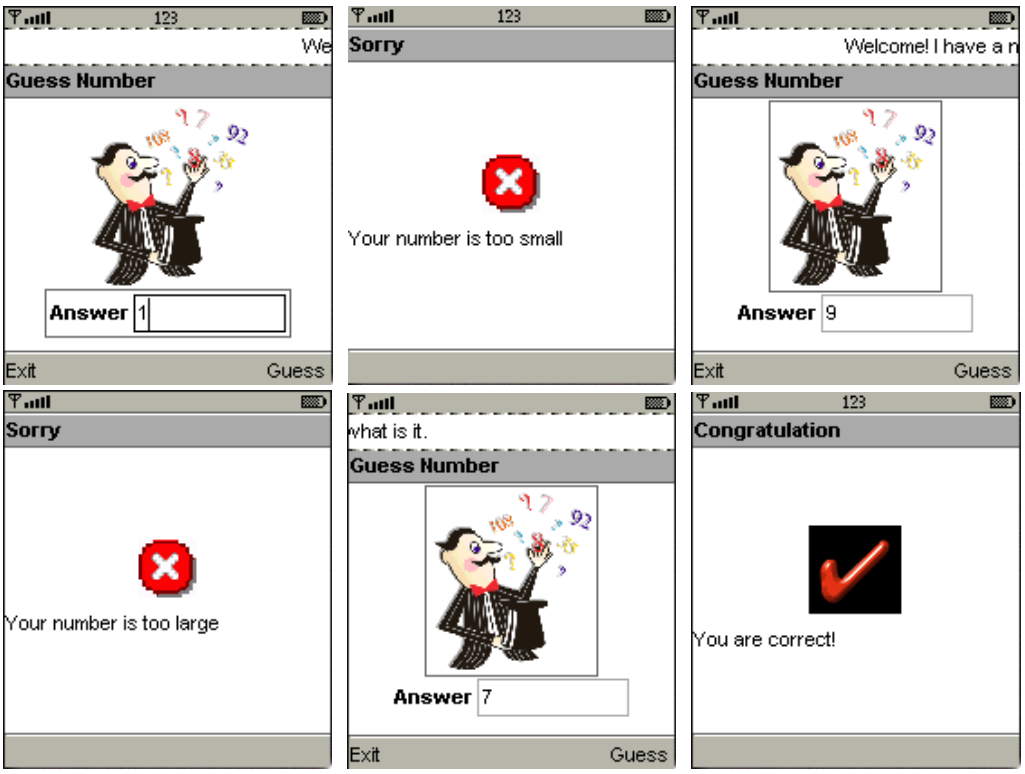
3. Create a Date Field in a Form, and adjust the calendar. (*Page 5 – 6*)

4. Create an Interactive Gauge in a Form. Then modify the maximum value and the initial value, and press the [🡐] [🡒] button to change the value. (*Page 7 – 8*)

5. Create an immutable image in a Form, and then modify the layout position from LAYOUT_CENTER to LAYOUT_LEFT or LAYOUT_RIGHT. Can you see the different between these parameters? Moreover, try to modify your code to support grayscale mobile device. (*Page 9 – 12*)

6. Create a choice group for checkbox, radio button and popup for a Form, then modify it to support [Select All] feature and embed image in the group. (*Page 13 – 22*)

7. Modify your "Guess Number" game as follow.



1. Create a String Item in a Form

1. In your "***J2ME Wireless Toolkit 2.2***", use **[New Project…]** to create a new project. Let's name the project at "***MyForm***" and the class at "***MyClass***". Then create a java source file called "***MyClass.java***" in the "***src***" folder under your project folder (C:\WTK22\apps\MyForm\src).

```
// include the MIDlet supper class
import javax.microedition.midlet.MIDlet;
// include the GUI libraries of MIDP
import javax.microedition.lcdui.*;

/*************************************************************************/
/* All MIDlet applications must extend the MIDlet class.                 */
/*************************************************************************/
public class MyClass extends MIDlet implements CommandListener {
  private Display midletDisplay;    // Reference to Display object
  private Command cmdExit;          // Command to exit the MIDlet
  private Form frmMain;             // Main form
  private StringItem strMessage;    // String Item

  // Define the no-argument constructor
  public MyClass() {
    // Retrieve the display from the static display object
    midletDisplay = Display.getDisplay(this);

    // Create the String Message
    strMessage = new StringItem("Game Center:", "Please Login");

    // Textfield for Username and Password
    cmdExit = new Command("Exit", Command.EXIT, 0);

    // Create Form, add Commands and textfield, listen for events
    frmMain = new Form("User Login");
    frmMain.append(strMessage);
    frmMain.addCommand(cmdExit);
    frmMain.setCommandListener(this);
  }

  // Called by application manager to start the MIDlet.
  public void startApp() {
    // Set the current display of the midlet to the Form
    midletDisplay.setCurrent(frmMain);
  }

  // PauseApp is used to suspend background activities and release resources
  // on the device when the midlet is not active.
  public void pauseApp() {
  }

  // DestroyApp is used to stop background activities and release
  // resources on the device when the midlet is at the end of its life cycle.
  public void destroyApp(boolean unconditional) {
  }

  // Implement the event handling method defined in the CommandListener interface.
  public void commandAction(Command c, Displayable s) {
    if (c == cmdExit) {
      destroyApp(true);
      notifyDestroyed();
    }
  }
}
```

2. Use the **[Build]** button to build the project, then press **[Run]** button to test the program.

2. Create a Text Field in a Form

1. Open the project "***MyForm***" you created before and modify the source code for your java file "***MyClass.java***" as follow:

```
// include the MIDlet supper class
import javax.microedition.midlet.MIDlet;
// include the GUI libraries of MIDP
import javax.microedition.lcdui.*;


/*****************************************************************************/
/* All MIDlet applications must extend the MIDlet class.                     */
/*****************************************************************************/
public class MyClass extends MIDlet implements CommandListener {
  private Display midletDisplay;    // Reference to Display object
  private Command cmdSubmit;        // Get contents of text field
  private Command cmdExit;          // Command to exit the MIDlet
  private Form frmMain;             // Main form
  private StringItem strMessage;    // String Item
  private TextField txtUserName;    // Text field to store User name
  private TextField txtPassword;    // Text field to store Password

  // Define the no-argument constructor
  public MyClass() {
    // Retrieve the display from the static display object
    midletDisplay = Display.getDisplay(this);

    // Create the String Message
    strMessage = new StringItem("Game Center:", "Please Login");

    // Textfield for Username and Password
    cmdSubmit = new Command("Submit", Command.SCREEN, 1);
    cmdExit = new Command("Exit", Command.EXIT, 0);

    // Textfield for Username and Password
    txtUserName = new TextField("User Name:", "", 8, TextField.ANY);
    txtPassword = new TextField("Password:", "", 8, TextField.PASSWORD);

    // Create Form, add Commands and textfield, listen for events
    frmMain = new Form("User Login");
    frmMain.append(strMessage);
    frmMain.append(txtUserName);
    frmMain.append(txtPassword);
    frmMain.addCommand(cmdExit);
    frmMain.addCommand(cmdSubmit);
    frmMain.setCommandListener(this);
  }

  // Called by application manager to start the MIDlet.
  public void startApp() {
    // Set the current display of the midlet to the Form
    midletDisplay.setCurrent(frmMain);
  }

  // PauseApp is used to suspend background activities and release resources
  // on the device when the midlet is not active.
  public void pauseApp() {
  }

  // DestroyApp is used to stop background activities and release
  // resources on the device when the midlet is at the end of its life cycle.
  public void destroyApp(boolean unconditional) {
  }
```

```
  // Implement the event handling method defined in the CommandListener interface.
  public void commandAction(Command c, Displayable s) {
    if (c == cmdSubmit) {
      Alert al = null;      // Output Alert
      String str = null;    // Output String

      // Concatenate the Username and Password to a string for output
      str = "Username=" + txtUserName.getString()
           + "\nPassword=" + txtPassword.getString();

      // Output the username and password as a alert
      al = new Alert("Login Info", str, null, AlertType.INFO);

      // Display the Alert on screen
      midletDisplay.setCurrent(al);
    } else if (c == cmdExit) {
      destroyApp(true);
      notifyDestroyed();
    }
  }
}
```

2. Use the **[Build]** button to build the project, then press **[Run]** button to test the program. Input the data in the text field of "User Name" and "Password", and then press the **[Submit]** to see the result.

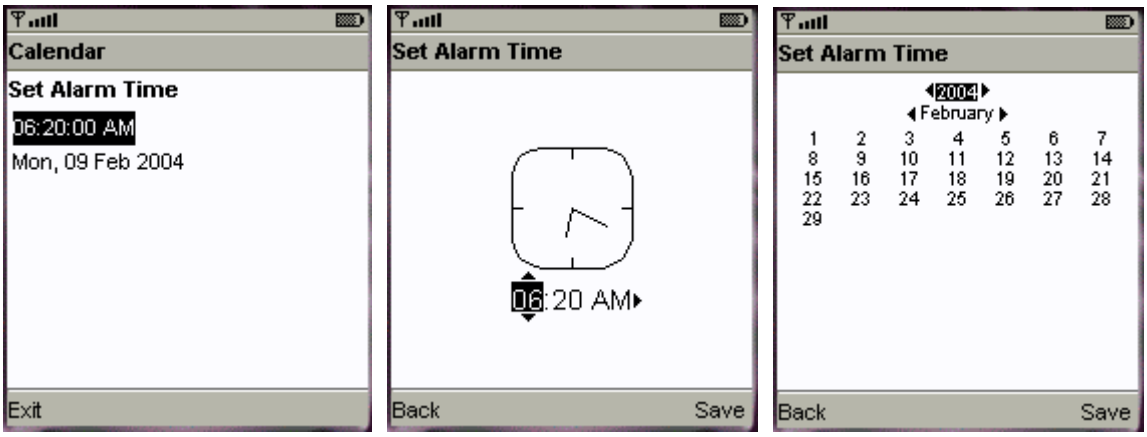
3. Create an Date Field

1. Create a new project, name the project and the class at "***MyDateField***" and "***MyClass***". Then create a java source file called "***MyClass.java***" in the "***src***" folder under your project folder (C:\WTK22\apps\MyDateField\src).

```
// include the MIDlet supper class
import javax.microedition.midlet.MIDlet;
// include the GUI libraries of MIDP
import javax.microedition.lcdui.*;
// include the MID Profile Utility Classes
import java.util.*;


/*****************************************************************************/
/* All MIDlet applications must extend the MIDlet class.                     */
/*****************************************************************************/
public class MyClass extends MIDlet implements CommandListener {
  private Display midletDisplay;        // Reference to Display object
  private Form frmMain;                 // Main form
  private Command cmdExit;              // Command Button to exit the MIDlet
  private DateField dfCalendar;         // DateField component

  // Define the no-argument constructor
  public MyClass() {
    // Retrieve the display from the static display object
    midletDisplay = Display.getDisplay(this);

    // The main form
    frmMain = new Form("Calendar");

    // DateField with todays date as a default
    dfCalendar = new DateField("Set Alarm Time", DateField.DATE_TIME);
    dfCalendar.setDate(new Date());

    // Add the Exit buttons
    cmdExit = new Command("Exit", Command.EXIT, 0);

    // Create Form, add Commands and DateField, listen for events
    frmMain.append(dfCalendar);
    frmMain.addCommand(cmdExit);
    frmMain.setCommandListener(this);
  }

  // Called by application manager to start the MIDlet.
  public void startApp() {
    // Set the current display of the midlet to the Form
    midletDisplay.setCurrent(frmMain);
  }

  // PauseApp is used to suspend background activities and release resources
  // on the device when the midlet is not active.
  public void pauseApp() {
  }

  // DestroyApp is used to stop background activities and release
  // resources on the device when the midlet is at the end of its life cycle.
  public void destroyApp(boolean unconditional) {
  }

  // Implement the event handling method defined in the CommandListener interface.
  public void commandAction(Command c, Displayable s) {
    if (c == cmdExit) {
      destroyApp(true);
```

```
      notifyDestroyed();
    }
  }
}
```

2. Use the **[Build]** button to build the project, then press **[Run]** button to test the program. Place your cursor to the time field, then press **[Select]** button to adjust the time. Or you can place your cursor on the date field and press **[Select]** to edit the date. Don't forget to press the **[Save]** button when you finish the adjustment.

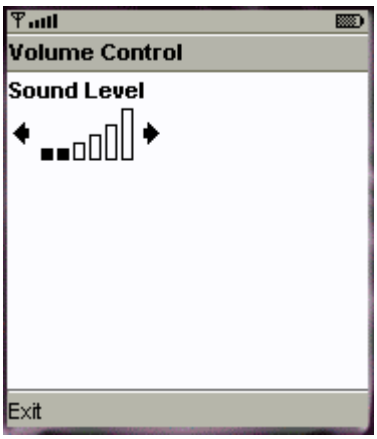


4. Create an Interactive Gauge in a Form

1. Create a new project, name the project and the class at "***InteractiveGauge***" and "***MyClass***". Then create a java source file called "***MyClass.java***" in the "***src***" folder under your project folder (C:\WTK22\apps\InteractiveGauge\src).

```
// include the MIDlet supper class
import javax.microedition.midlet.MIDlet;
// include the GUI libraries of MIDP
import javax.microedition.lcdui.*;

/*****************************************************************************/
/* All MIDlet applications must extend the MIDlet class.                     */
/*****************************************************************************/
public class MyClass extends MIDlet implements CommandListener {
  private Display midletDisplay;        // Reference to Display object
  private Form frmMain;                 // Main form
  private Command cmdExit;              // Command Button to exit the MIDlet
  private Gauge gaVolume;               // Volume adjustment

  // Define the no-argument constructor
  public MyClass() {
    // Retrieve the display from the static display object
    midletDisplay = Display.getDisplay(this);

    // The main form
    frmMain = new Form("Volume Control");

    // Create the interactive gauge with max value = 6; initial value= 2
    gaVolume = new Gauge("Sound Level", true, 6, 2);

    // Add the Exit buttons
    cmdExit = new Command("Exit", Command.EXIT, 0);

    // Create Form, add Commands and textfield, listen for events
    frmMain.append(gaVolume);
    frmMain.addCommand(cmdExit);
    frmMain.setCommandListener(this);
  }

  // Called by application manager to start the MIDlet.
  public void startApp() {
    // Set the current display of the midlet to the Form
    midletDisplay.setCurrent(frmMain);
  }

  // PauseApp is used to suspend background activities and release resources
  // on the device when the midlet is not active.
  public void pauseApp() {
  }

  // DestroyApp is used to stop background activities and release
  // resources on the device when the midlet is at the end of its life cycle.
  public void destroyApp(boolean unconditional) {
  }

  // Implement the event handling method defined in the CommandListener interface.
  public void commandAction(Command c, Displayable s) {
    if (c == cmdExit) {
      destroyApp(true);
      notifyDestroyed();
    }
  }
}
```

2. Use the **[Build]** button to build the project, then press **[Run]** button to test the program. You can use the [🡐] and [🡒] buttons to adjust the volume.



7. Create an Immutable Image in a Form

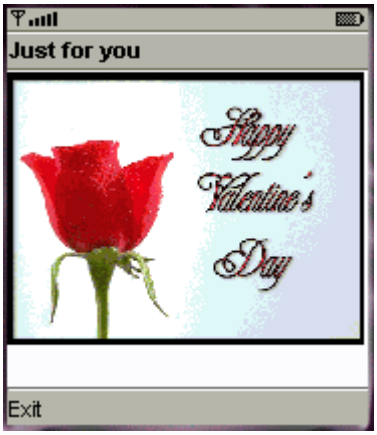
1. Create a new project, name the project and the class at "***MyFormImage***" and "***MyClass***", and then copy the PNG image to the resource folder (C:\WTK22\apps\MyFormImage\res). *If you are using Eclipse, you are required to put the images to the folder: workspace\ProjectName\verified\classes.*

2. Create a java source file called "***MyClass.java***" in the "***src***" folder under your project folder (C:\WTK22\apps\MyFormImage\src).

```
// include the MIDlet supper class
import javax.microedition.midlet.MIDlet;
// include the GUI libraries of MIDP
import javax.microedition.lcdui.*;


/**************************************************************************/
/* All MIDlet applications must extend the MIDlet class.                  */
/**************************************************************************/
public class MyClass extends MIDlet implements CommandListener {
  private Display midletDisplay;        // Reference to Display object
  private Form frmMain;                 // Main form
  private Command cmdExit;              // Command Button to exit the MIDlet

  // Define the no-argument constructor
  public MyClass() {
    // Retrieve the display from the static display object
    midletDisplay = Display.getDisplay(this);

    // The main form
    frmMain = new Form("Just for you");

    // All the Exit buttons
    cmdExit = new Command("Exit", Command.EXIT, 0);

    // Create Form, add Commands, listen for events
    frmMain.addCommand(cmdExit);
    frmMain.setCommandListener(this);

    // Read the appropriate image
    try {
      Image im = Image.createImage("/picture.png");
      frmMain.append(new ImageItem(null, im, ImageItem.LAYOUT_CENTER, null));
    } catch (java.io.IOException e) {
      System.err.println("Unable to locate or read the PNG file");
    }
  }
```

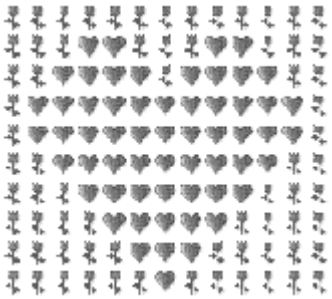
```
  // Called by application manager to start the MIDlet.
  public void startApp() {
    // Set the current display of the midlet to the Form
    midletDisplay.setCurrent(frmMain);
  }

  // PauseApp is used to suspend background activities and release resources
  // on the device when the midlet is not active.
  public void pauseApp() {
  }

  // DestroyApp is used to stop background activities and release
  // resources on the device when the midlet is at the end of its life cycle.
  public void destroyApp(boolean unconditional) {
  }

  // Implement the event handling method defined in the CommandListener interface.
  public void commandAction(Command c, Displayable s) {
    if (c == cmdExit) {
      destroyApp(true);
      notifyDestroyed();
    }
  }
}
```

3. Use the **[Build]** button to build the project, then press **[Run]** button to test the program. I hope you like my Saint Valentine Card. layout position from LAYOUT_CENTER to LAYOUT_LEFT or LAYOUT_RIGHT. Can you see the different between these parameters?



8. Detect the color support for a mobile device

1. Open the project "***MyFormImage***" you just created, and then copy two images (one in color and one in grayscale) to the resource folder (C:\WTK22\apps\MyFormImage\res).

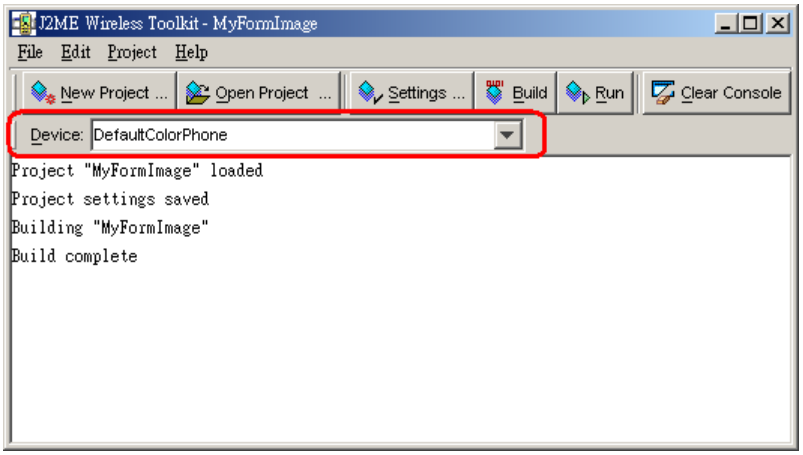


2. Create a java source file called "***MyClass.java***" in the "***src***" folder under your project folder (C:\WTK22\apps\MyFormImage\src).

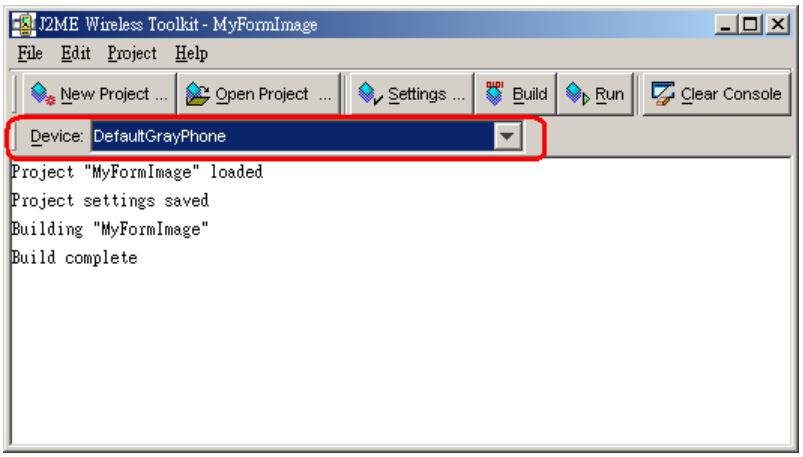
```
// include the MIDlet supper class
import javax.microedition.midlet.MIDlet;
// include the GUI libraries of MIDP
import javax.microedition.lcdui.*;



/*************************************************************************/
/* All MIDlet applications must extend the MIDlet class.                 */
/*************************************************************************/
public class MyClass extends MIDlet implements CommandListener {
  private Display midletDisplay;        // Reference to Display object
  private Form frmMain;                 // Main form
  private Command cmdExit;              // Command Button to exit the MIDlet

  // Define the no-argument constructor
  public MyClass() {
    // Retrieve the display from the static display object
    midletDisplay = Display.getDisplay(this);

    // The main form
    frmMain = new Form("Just for you");

    // All the Exit buttons
    cmdExit = new Command("Exit", Command.EXIT, 0);

    // Create Form, add Commands, listen for events
    frmMain.addCommand(cmdExit);
    frmMain.setCommandListener(this);

    // Read the appropriate image
    try {
      // Read the appropriate image based on color support
      Image im = Image.createImage((midletDisplay.isColor()) ? "/color.png":
"/grayscale.png");
      frmMain.append(new ImageItem(null, im, ImageItem.LAYOUT_CENTER, null));
    } catch (java.io.IOException e) {
      System.err.println("Unable to locate or read the PNG file");
    }
  }

  // Called by application manager to start the MIDlet.
```

```
  public void startApp() {
    // Set the current display of the midlet to the Form
    midletDisplay.setCurrent(frmMain);
  }

  // PauseApp is used to suspend background activities and release resources
  // on the device when the midlet is not active.
  public void pauseApp() {
  }

  // DestroyApp is used to stop background activities and release
  // resources on the device when the midlet is at the end of its life cycle.
  public void destroyApp(boolean unconditional) {
  }

  // Implement the event handling method defined in the CommandListener interface.
  public void commandAction(Command c, Displayable s) {
    if (c == cmdExit) {
      destroyApp(true);
      notifyDestroyed();
    }
  }
}
```

3. Use the **[Build]** button to build the project. Then select "DefaultColorPhone" and press the [Run] button to execute the MIDlet, a color picture will display on your screen.



4. Select "DefaultGrayPhone" on the "Device" list, and press the [Run] button again. Can you see the grayscale photo this time?



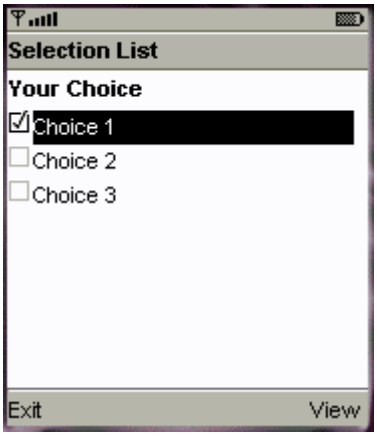
9. Create a choice group for a Form (Checkbox)

1. Create a new project, name the project and the class at "***MyChoice***" and "***MyClass***". Then create a java source file called "***MyClass.java***" in the "***src***" folder under your project folder (C:\WTK22\apps\MyChoice\src).

```
// include the MIDlet supper class
import javax.microedition.midlet.MIDlet;
// include the GUI libraries of MIDP
import javax.microedition.lcdui.*;


/*****************************************************************************/
/* All MIDlet applications must extend the MIDlet class.                     */
/*****************************************************************************/
public class MyClass extends MIDlet implements CommandListener {
  private Display midletDisplay;    // Reference to Display object
  private Form frmMain;             // Main form
  private Command cmdExit;          // Command to exit the MIDlet
  private Command cmdView;          // View the choice selected
  private ChoiceGroup cgPrefs;      // Choice Group of preferences
  private int choiceGroupIndex;     // Index of choice group on form

  // Define the no-argument constructor
  public MyClass() {
    // Retrieve the display from the static display object
    midletDisplay = Display.getDisplay(this);

    // Create a multiple choice group
    cgPrefs = new ChoiceGroup("Your Choice", Choice.MULTIPLE);

    // Append options, with no associated images
    cgPrefs.append("Choice 1 ", null);
    cgPrefs.append("Choice 2 ", null);
    cgPrefs.append("Choice 3 ", null);

    // Create the View and Exit button
    cmdExit = new Command("Exit", Command.EXIT, 0);
    cmdView = new Command("View", Command.SCREEN, 2);

    // Create Form, add components, listen for events
    frmMain = new Form("Selection List");
    choiceGroupIndex = frmMain.append(cgPrefs);
    frmMain.addCommand(cmdView);
    frmMain.addCommand(cmdExit);
    frmMain.setCommandListener(this);
  }

  // Called by application manager to start the MIDlet.
  public void startApp() {
    // Set the current display of the midlet to the Form
    midletDisplay.setCurrent(frmMain);
  }

  // PauseApp is used to suspend background activities and release resources
  // on the device when the midlet is not active.
  public void pauseApp() {
  }

  // DestroyApp is used to stop background activities and release
  // resources on the device when the midlet is at the end of its life cycle.
  public void destroyApp(boolean unconditional) {
  }
```

```
  // Implement the event handling method defined in the CommandListener interface.
  public void commandAction(Command c, Displayable s) {
    if (c == cmdView) {
      boolean selected[] = new boolean[cgPrefs.size()];
      // Fill array indicating whether each element is checked
      cgPrefs.getSelectedFlags(selected);
      for (int i = 0; i < cgPrefs.size(); i++) {
        System.out.println(cgPrefs.getString(i) + selected[i]);
      }
    } else if (c == cmdExit) {
      destroyApp(true);
      notifyDestroyed();
    }
  }
}
```

2. Compile and execute the MIDlet. You can use your up and down button to control the cursor and press **[Select]** to tick the check box. Then you can press **[View]** to know which box you have selected.



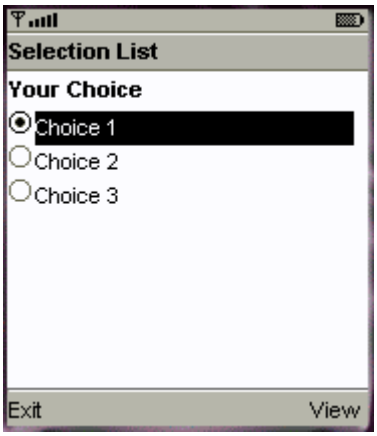
10. Create a choice group for a Form (Radio Button)

1. Open the project "***MyChoice***" you created, and then modify the source code of "***MyClass.java***" as follow:

```
// include the MIDlet supper class
import javax.microedition.midlet.MIDlet;
// include the GUI libraries of MIDP
import javax.microedition.lcdui.*;


/*************************************************************************/
/* All MIDlet applications must extend the MIDlet class.                 */
/*************************************************************************/
public class MyClass extends MIDlet implements CommandListener {
  private Display midletDisplay;    // Reference to Display object
  private Form frmMain;             // Main form
  private Command cmdExit;          // Command to exit the MIDlet
  private Command cmdView;          // View the choice selected
  private ChoiceGroup cgPrefs;      // Choice Group of preferences
  private int choiceGroupIndex;     // Index of choice group on form


  // Define the no-argument constructor
  public MyClass() {
    // Retrieve the display from the static display object
    midletDisplay = Display.getDisplay(this);

    // Create a multiple choice group
    cgPrefs = new ChoiceGroup("Your Choice", Choice.EXCLUSIVE);

    // Append options, with no associated images
    cgPrefs.append("Choice 1 ", null);
    cgPrefs.append("Choice 2 ", null);
    cgPrefs.append("Choice 3 ", null);

    // Create the View and Exit button
    cmdExit = new Command("Exit", Command.EXIT, 0);
    cmdView = new Command("View", Command.SCREEN, 2);

    // Create Form, add components, listen for events
    frmMain = new Form("Selection List");
    choiceGroupIndex = frmMain.append(cgPrefs);
    frmMain.addCommand(cmdView);
    frmMain.addCommand(cmdExit);
    frmMain.setCommandListener(this);
  }


  // Called by application manager to start the MIDlet.
  public void startApp() {
    // Set the current display of the midlet to the Form
    midletDisplay.setCurrent(frmMain);
  }

  // PauseApp is used to suspend background activities and release resources
  // on the device when the midlet is not active.
  public void pauseApp() {
  }


  // DestroyApp is used to stop background activities and release
  // resources on the device when the midlet is at the end of its life cycle.
  public void destroyApp(boolean unconditional) {
  }
```

```
  // Implement the event handling method defined in the CommandListener interface.
  public void commandAction(Command c, Displayable s) {
    if (c == cmdView) {
      boolean selected[] = new boolean[cgPrefs.size()];
      // Fill array indicating whether each element is checked
      cgPrefs.getSelectedFlags(selected);
      for (int i = 0; i < cgPrefs.size(); i++) {
        System.out.println(cgPrefs.getString(i) + selected[i]);
      }
    } else if (c == cmdExit) {
      destroyApp(true);
      notifyDestroyed();
    }
  }
}
```

2. Compile and execute the MIDlet. You can use your up and down button to control the cursor and press **[Select]** to tick the check box. Then you can press **[View]** to know which button you have selected.



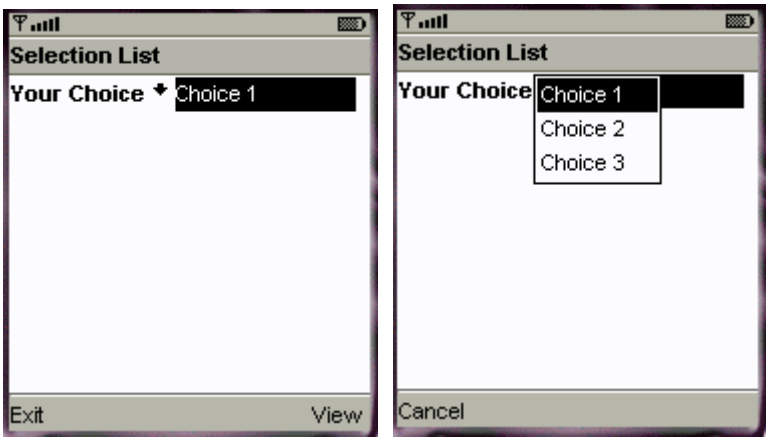
11. Create a choice group for a Form (Pop-up)

1. Open the project "***MyChoice***" you created, and then modify the source code of "***MyClass.java***" as follow:

```
// include the MIDlet supper class
import javax.microedition.midlet.MIDlet;
// include the GUI libraries of MIDP
import javax.microedition.lcdui.*;


/*****************************************************************************/
/* All MIDlet applications must extend the MIDlet class.                     */
/*****************************************************************************/
public class MyClass extends MIDlet implements CommandListener {
  private Display midletDisplay;    // Reference to Display object
  private Form frmMain;             // Main form
  private Command cmdExit;          // Command to exit the MIDlet
  private Command cmdView;          // View the choice selected
  private ChoiceGroup cgPrefs;      // Choice Group of preferences
  private int choiceGroupIndex;     // Index of choice group on form


  // Define the no-argument constructor
  public MyClass() {
    // Retrieve the display from the static display object
    midletDisplay = Display.getDisplay(this);

    // Create a multiple choice group
    cgPrefs = new ChoiceGroup("Your Choice", Choice.POPUP);

    // Append options, with no associated images
    cgPrefs.append("Choice 1 ", null);
    cgPrefs.append("Choice 2 ", null);
    cgPrefs.append("Choice 3 ", null);

    // Create the View and Exit button
    cmdExit = new Command("Exit", Command.EXIT, 0);
    cmdView = new Command("View", Command.SCREEN, 2);

    // Create Form, add components, listen for events
    frmMain = new Form("Selection List");
    choiceGroupIndex = frmMain.append(cgPrefs);
    frmMain.addCommand(cmdView);
    frmMain.addCommand(cmdExit);
    frmMain.setCommandListener(this);
  }


  // Called by application manager to start the MIDlet.
  public void startApp() {
    // Set the current display of the midlet to the Form
    midletDisplay.setCurrent(frmMain);
  }

  // PauseApp is used to suspend background activities and release resources
  // on the device when the midlet is not active.
  public void pauseApp() {
  }


  // DestroyApp is used to stop background activities and release
  // resources on the device when the midlet is at the end of its life cycle.
  public void destroyApp(boolean unconditional) {
  }
```

```
  // Implement the event handling method defined in the CommandListener interface.
  public void commandAction(Command c, Displayable s) {
    if (c == cmdView) {
      boolean selected[] = new boolean[cgPrefs.size()];
      // Fill array indicating whether each element is checked
      cgPrefs.getSelectedFlags(selected);
      for (int i = 0; i < cgPrefs.size(); i++) {
        System.out.println(cgPrefs.getString(i) + selected[i]);
      }
    } else if (c == cmdExit) {
      destroyApp(true);
      notifyDestroyed();
    }
  }
}
```

2. Compile and execute the MIDlet. You can press **[Select]** on the choice group to call up the pop up menu. Then you can use the cursor to select your choice.



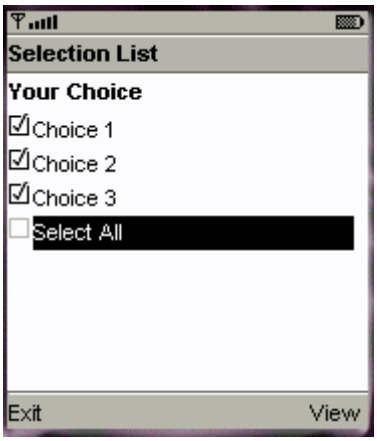
12. Handling [Select All] event in a choice group

1. Open the project "***MyChoice***" you created, and then modify the source code of "***MyClass.java***" as follow:

```
// include the MIDlet supper class
import javax.microedition.midlet.MIDlet;
// include the GUI libraries of MIDP
import javax.microedition.lcdui.*;


/*************************************************************************/
/* All MIDlet applications must extend the MIDlet class.                 */
/*************************************************************************/
public class MyClass extends MIDlet implements CommandListener, ItemStateListener {
  private Display midletDisplay;    // Reference to Display object
  private Form frmMain;             // Main form
  private Command cmdExit;          // Command to exit the MIDlet
  private Command cmdView;          // View the choice selected
  private ChoiceGroup cgPrefs;      // Choice Group of preferences
  private int choiceGroupIndex;     // Index of choice group on form
  private int selectAllIndex;       // Index of the "Select All" option

  // Define the no-argument constructor
  public MyClass() {
    // Retrieve the display from the static display object
    midletDisplay = Display.getDisplay(this);

    // Create a multiple choice group
    cgPrefs = new ChoiceGroup("Your Choice", Choice.MULTIPLE);

    // Append options, with no associated images
    cgPrefs.append("Choice 1 ", null);
    cgPrefs.append("Choice 2 ", null);
    cgPrefs.append("Choice 3 ", null);
    selectAllIndex = cgPrefs.append("Select All", null);

    // Create the View and Exit button
    cmdExit = new Command("Exit", Command.EXIT, 0);
    cmdView = new Command("View", Command.SCREEN, 2);

    // Create Form, add components, listen for events
    frmMain = new Form("Selection List");
    choiceGroupIndex = frmMain.append(cgPrefs);
    frmMain.addCommand(cmdView);
    frmMain.addCommand(cmdExit);
    frmMain.setCommandListener(this);
    frmMain.setItemStateListener(this);
  }

  // Called by application manager to start the MIDlet.
  public void startApp() {
    // Set the current display of the midlet to the Form
    midletDisplay.setCurrent(frmMain);
  }

  // PauseApp is used to suspend background activities and release resources
  // on the device when the midlet is not active.
  public void pauseApp() {
  }

  // DestroyApp is used to stop background activities and release
  // resources on the device when the midlet is at the end of its life cycle.
  public void destroyApp(boolean unconditional) {
  }
```

```
  // Implement the event handling method defined in the CommandListener interface.
  public void commandAction(Command c, Displayable s) {
    if (c == cmdView) {
      boolean selected[] = new boolean[cgPrefs.size()];
      // Fill array indicating whether each element is checked
      cgPrefs.getSelectedFlags(selected);
      for (int i = 0; i < cgPrefs.size(); i++) {
        System.out.println(cgPrefs.getString(i) + selected[i]);
      }
    } else if (c == cmdExit) {
      destroyApp(true);
      notifyDestroyed();
    }
  }

  // Handle the [Select All] option
  public void itemStateChanged(Item item) {
    if (item == cgPrefs) {
      // Is "Select all" option checked ?
      if (cgPrefs.isSelected(selectAllIndex)) {
        // Set all checkboxes to true
        for (int i = 0; i < cgPrefs.size(); i++) {
          cgPrefs.setSelectedIndex(i, true);
        }
        // Remove the check by "Select All"
        cgPrefs.setSelectedIndex(selectAllIndex, false);
      }
    }
  }
}
```

2. Compile and execute the MIDlet. Move the cursor to “Select All” item and press the **[Select]** button to notify the change.



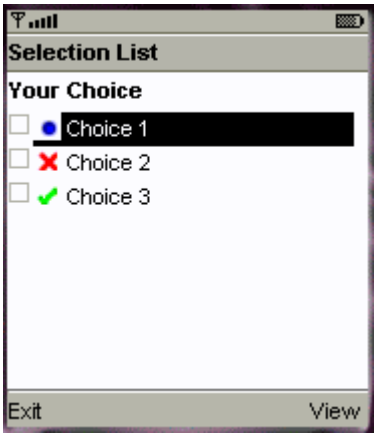
13. Embed Image to a choice group

1. Open the project "***MyChoice***" you created, and then modify the source code of "***MyClass.java***" as follow:

```
// include the MIDlet supper class
import javax.microedition.midlet.MIDlet;
// include the GUI libraries of MIDP
import javax.microedition.lcdui.*;



/*****************************************************************************/
/* All MIDlet applications must extend the MIDlet class.                     */
/*****************************************************************************/
public class MyClass extends MIDlet implements CommandListener {
  private Display midletDisplay;    // Reference to Display object
  private Form frmMain;             // Main form
  private Command cmdExit;          // Command to exit the MIDlet
  private Command cmdView;          // View the choice selected
  private ChoiceGroup cgPrefs;      // Choice Group of preferences
  private int choiceGroupIndex;     // Index of choice group on form

  // Define the no-argument constructor
  public MyClass() {
    // Retrieve the display from the static display object
    midletDisplay = Display.getDisplay(this);

    // Create a multiple choice group
    cgPrefs = new ChoiceGroup("Your Choice", Choice.MULTIPLE);

    // Append options, with associated images
    try {
      cgPrefs.append(" Choice 1 ", Image.createImage("/choice1.png"));
      cgPrefs.append(" Choice 2 ", Image.createImage("/choice2.png"));
      cgPrefs.append(" Choice 3 ", Image.createImage("/choice3.png"));
    } catch (java.io.IOException e) {
      System.err.println("Unable to locate or read the PNG file");
    }

    // Create the View and Exit button
    cmdExit = new Command("Exit", Command.EXIT, 0);
    cmdView = new Command("View", Command.SCREEN, 2);

    // Create Form, add components, listen for events
    frmMain = new Form("Selection List");
    choiceGroupIndex = frmMain.append(cgPrefs);
    frmMain.addCommand(cmdView);
    frmMain.addCommand(cmdExit);
    frmMain.setCommandListener(this);
  }

  // Called by application manager to start the MIDlet.
  public void startApp() {
    // Set the current display of the midlet to the Form
    midletDisplay.setCurrent(frmMain);
  }

  // PauseApp is used to suspend background activities and release resources
  // on the device when the midlet is not active.
  public void pauseApp() {
  }

  // DestroyApp is used to stop background activities and release
  // resources on the device when the midlet is at the end of its life cycle.
```

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

  // Implement the event handling method defined in the CommandListener interface.
  public void commandAction(Command c, Displayable s) {
    if (c == cmdView) {
      boolean selected[] = new boolean[cgPrefs.size()];
      // Fill array indicating whether each element is checked
      cgPrefs.getSelectedFlags(selected);
      for (int i = 0; i < cgPrefs.size(); i++) {
        System.out.println(cgPrefs.getString(i) + selected[i]);
      }
    } else if (c == cmdExit) {
      destroyApp(true);
      notifyDestroyed();
    }
  }
}
```

2. Compile and execute the MIDlet. You can see the icon on the left of your list. Besides, you can still use your up and down button to control the cursor and press **[Select]** to tick the check box, and press **[View]** to know which box you have selected.



Appendix: Guess Number

1. Create a new project, name the project and the class at "***GuessNumber***" and "***MyClass***". Then create a java source file called "***MyClass.java***" in the "***src***" folder under your project folder (C:\WTK22\apps\GuessNumber\src).

```
// include the MIDlet supper class
import javax.microedition.midlet.MIDlet;
// include the GUI libraries of MIDP
import javax.microedition.lcdui.*;
// include Java Random utility
import java.util.Random;

/*************************************************************************/
/* All MIDlet applications must extend the MIDlet class.                 */
/*************************************************************************/
public class MyClass extends MIDlet implements CommandListener {
  private Display midletDisplay;        // Reference to Display object
  private Command cmdGuess;             // Command Button to Guess Number
  private Command cmdExit;              // Command Button to exit
  private TextBox mainScreen;           // Text Box to store Answer
  private Random rand = new Random();   // Initialize Random Seek
  private int RandomNumber = 0;         // Random Number

  // Define the no-argument constructor
  public MyClass() {
    // Retrieve the display from the static display object
    midletDisplay = Display.getDisplay(this);

    // Create the View and Exit button
    cmdExit  = new Command("Exit", Command.EXIT, 0);
    cmdGuess = new Command("Guess", Command.SCREEN, 1);

    // Create Textbox to store answer
    mainScreen = new TextBox ("Answer", "", 8, TextField.NUMERIC);

    // Create the View and Exit Button
    mainScreen.addCommand(cmdExit);
    mainScreen.addCommand(cmdGuess);

    // Add the Command Listener
    mainScreen.setCommandListener(this);

    // Set the current display of the midlet to the textBox screen
    midletDisplay.setCurrent(mainScreen);
  }

  // Called by application manager to start the MIDlet.
  public void startApp() {
    // Random a number between 0 to 9
    RandomNumber = Math.abs(rand.nextInt()) % 9;

    // Set the current display of the midlet to the Form
    midletDisplay.setCurrent(mainScreen);

    // Print out the answer for debug only
     System.out.println(RandomNumber);
  }

  // PauseApp is used to suspend background activities and release resources
  // on the device when the midlet is not active.
  public void pauseApp() {
  }
```

```
  // DestroyApp is used to stop background activities and release
  // resources on the device when the midlet is at the end of its life cycle.
  public void destroyApp(boolean unconditional) {
  }

  // Implement the event handling method defined in the CommandListener interface.
  public void commandAction(Command c, Displayable s) {

    if (c == cmdGuess) {
      Alert al = null;      // Output Alert

      try {
        if (Integer.parseInt(mainScreen.getString()) == RandomNumber) {
          al = new Alert("Congratulation", "You are correct!",
               Image.createImage("/correct.png"), AlertType.CONFIRMATION);
        } else if (Integer.parseInt(mainScreen.getString()) > RandomNumber) {
          al = new Alert("Sorry", "Your number is too large",
               Image.createImage("/error.png"), AlertType.ERROR);
        } else if (Integer.parseInt(mainScreen.getString()) < RandomNumber) {
          al = new Alert("Sorry", "Your number is too small",
               Image.createImage("/error.png"), AlertType.ERROR);
        }
      } catch (java.io.IOException e) {
        System.err.println("Unable to locate or read the PNG file");
      }

      // Display the Alert on screen
      midletDisplay.setCurrent(al);
    } else if (c == cmdExit) {
      destroyApp(true);
      notifyDestroyed();
    }
  }
}
```

2. Use the **[Build]** button to build the project, then press **[Run]** button to test the program. You can input the number and press **[Guess]** to guess the number or press **[Exit]** to quit the game.