Objective

The objective to this lab is to:

1. Take a deeper look into Java ME
2. Deploy Java ME MIDlet applications on the Blackberry device

Experiment 1 Working with the Record Management System (RMS)

In this experiment, you will use RMS APIs to build a phone book for your mobile device. RMS which is a simple-record oriented database enables MIDlets to persistently store data and retrieve it later.

To start the experiment, do the following tasks:

1. The following methods demonstrate how to use RMS APIs to manage and manipulate data. Read them carefully.

Listing 1: ReadWrite.java

```
/* Make ReadWrite.java a public static class if you want to try the methods out */
/* To open a record store*/
public void openRecStore(String recordStore_name)
{
    try{
        RecordStore rs =
            RecordStore.openRecordStore(recordStore_name,true);
        //rs.addRecordListener(new Lab2RecordListner());
    }
    catch(Exception e)
    {
        System.err.println(e.toString());
    }
}

  /* To close a record store*/
public void closeRecStore()
{
    try{
        rs.closeRecordStore();
    }
    catch(Exception e)
    {
```

```
            System.err.println(e.toString());

        }
    }

    /* To add a new record to a record store*/
    public void writeRecord(String str)
    {
        byte[] rec = str.getBytes();
        try
        {
            rs.addRecord(rec, 0, rec.length);
        }
        catch (Exception e)
        {
            System.err.println(e.toString());

        }
    }


    /* To return a copy of the data stored in the given record*/
    public String readRecords(int i)
    {
        byte[] recData= null;
        try
        {
            recData = rs.getRecord(i);
            return new String(recData);
        }
        catch (Exception e)
        {
            System.err.println(e.toString());
            return null;
        }
    }

    /* Todelete a record from the record store*/
    public void deleteRecord(int recId)
    {
        try{
            rs.deleteRecord(recId);
        }
        catch(Exception e)
        {
            System.err.println(e.toString());
        }
    }

    /* To update a record in the record store*/
    public void updateRecords(int recordindex, String str)
```

```
 {
   try
    {
       rs.setRecord(recordindex, str.getBytes(), 0, str.length());
    }
catch(Exception e)
  {
 System.err.println(e.toString());

 }
}

/* To delete the record store*/
public void deleteRecStore( String recordStore_name)
{
   if (RecordStore.listRecordStores() != null)
    {
       try
       {
          RecordStore.deleteRecordStore(recordStore_name);
       }
       catch (Exception e)
       {
          System.err.println(e.toString());

       }
    }
}

/* To check whether a record exists in the record store*/
public boolean isRecordExist(int i)
{
   byte[] recData=null;
   try{
      recData = rs.getRecord(i);
      if(recData!= null)
         return true;
      else
         return false;

   }catch (Exception e)
    {
      System.err.println(e.toString());

       return false;
    }
}
```

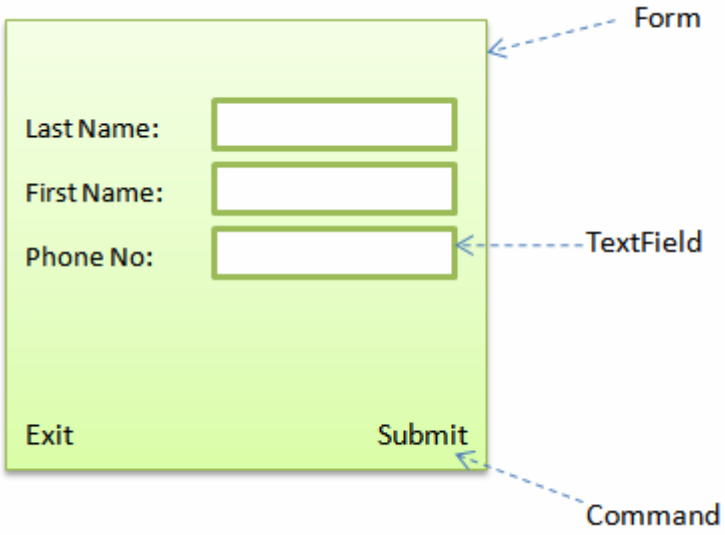
2. To get more information on RMS APIs go to the following link: http://developers.sun.com/techtopics/mobility/midp/articles/persist

Exercise 1 Write a Phone Book application

In this exercise, you will use all the experiments in developing MIDlet applications gathered in Lab 1 and Lab 2 to develop a Phone Book application. In this application you are able to store new contact names in a Record Store.

To get started, do the following steps:

1. Create a Form with the following items displayed in Figure 1:



In order to create a Form, read the following explanation carefully.

```
 …..
Display display;
Form mainForm;
 …..
public PhoneBook () {
      mainForm = new Form ("Phone Book");
      TextField lastName = new TextField("Last Name:", null, 15,
                  TextField.ANY);
      TextField firstName = new TextField("First Name:", null, 15,
                  TextField.ANY);
      TextField phone = new TextField("Phone No:", null, 10,
                  TextField.PHONENUMBER);

      mainForm.append(lastName);
      mainForm.append(firstName);
      mainForm.append(phone);
}

 public void startApp() {
```

```
        display = Display.getDisplay (this);
        display.setCurrent (mainForm);
    }
```

2. Add two Commands to the Form. One is submit_Command and the other is exit_Command (Explained in Lab 1)
3. In the commandAction method, insert action codes for the submit_Command and exit_Command :

```
public void commandAction(Command command, Displayable displayable) {
   // Insert global pre-action code here

      if (displayable == mainForm) {
         if (command == exit_Command) {
              exitMIDlet();
         } else if (command == submit_Command) {

                    ……Insert Appropriate RMS API code….
           }
       }
}
```

4. Insert appropriate methods from the Listing 1 in the action code for Submit_Command in order to store new contact name into the Record Store. Thus, when a user submits the form, new contact information including Last Name, First Name and Phone are saved in the Record Store.

5. Open a new project by clicking on New Project…(described in Lab 1)
6. Copy the java class into the src folder of the project created. The project is created in WTK directory > apps sub-directory.
7. Run the project in WTK22 environment. [**5 marks**]

Exercise 2 Deploying the Phone Book application onto the Blackberry

In this exercise you will learn how to deploy Java ME applications onto the Blackberry device. To run an existing MIDlet on a Java-enabled Blackberry device, you need to convert the .jad and .jar files of the existing application into a .cod file suitable for the BlackBerry. This can be done using the rapc compiler that comes with RIM's JDE. The "rapc" compiler can be accessed from the "bin" directory of RIM's JDE installation directory.

To convert and deploy the application on the BlackBerry device, do the following:

1. First you need to generate the .jad and .jar file for the PhoneBook application. To do this, go to the directory which you created the application and copy the .jad and .jar files that have been generated to the "bin" directory of RIM's JDE.
2. Use the rapc command as described in this article to generate the .cod file. http://developers.sun.com/techtopics/mobility/midp/articles/blackberrydev/ Use the same syntax as in the article but change the filenames.

3. Once the .cod file has been successfully generated, it is time to load the application on the BlackBerry device. To do that:
    a. Get a BlackBerry device and a USB cable from the TA
    b. Connect the USB cable to the device and the desktop
    c. Use the "javaloader" command as described in this article to load the application on the BlackBerry:
    d. http://developers.sun.com/techtopics/mobility/midp/articles/blackberrydev
    e. Disconnect the USB cable
    f. Check that the application has been loaded on the BlackBerry
4. Test the application the BlackBerry by adding some new contact names. The information will be saved on the device.
5. Demo the application from the BlackBerry device to the TA.
6. Remove the application from the BlackBerry using the command (javaloader –usb erase FileName.cod)
7. Give the BlackBerry device back to the TA. Thank you. [**5 marks**]