Introduction

The purpose of this homework is to get you acquainted with some of the development tools that we will be using this quarter and have you write your first midlet to run on a (simulated) cell phone.

We have provided you with two sample midlet suites. The first is SimpleHello, the classic HelloWorld program. Use this as a target for making sure that your compiler and build tools are installed correctly.

The second midlet suite is HW1. As provided to you, this suite contains two midlets: a property viewer and a simple database storage example. Your task is to implement an image viewer midlet like the one I showed in class and add it to the suite.

Preparation:

1. If you are working on a non-lab machine, install the course software as described on the class web site software page. If you are on a lab machine, verify that the various parts and pieces are available once you have logged on. I requested Ant/Antenna very late and so they may not be available for this assignment on the Windows lab machines. However, you can use the Wireless Toolkit directly to build and run the midlets.

2. Download the zip files for this homework from the web site. Unzip SimpleHello and take a look at the source code. Pretty simple application, isn't it? You can use KToolBar to open the project, build it and run it. Note that KToolBar expects that the files are located in the WTK installation directory in subdirectory apps, so you'll have to do a little careful file placement if you are using KToolBar. If you have Ant available, you should be able to use the build403.xml file to build the project and run it, no matter where it is.

4. If you can get a little display going that shows something like this, then you're in business. Read over the SimpleHello.java code and start to understand how a midlet is constructed and what its life cycle consists of.

Assignment

Unzip the second zip file that contains the hw1 midlet suite. There are two midlets provided to you, your task is to write a third one that implements the ImageViewer capability.

1. The SimpleHello midlet is an example of an application that is all contained in one class definition. It does not use much in the way of support from other classes. For a very small function, that is acceptable, although it is easy for a class to get large and hard to follow very quickly.

2. When an application gets larger, the design must include support packages to abstract away some of the detail clutter. In the hw1 suite, I used two packages to give you some examples of using the device capabilities.

There is a package called "menu" that provides rudimentary page management capability. This is not an "official" Sun design or something like that, it is just one implementation of a menu package.

There is a package called "store" that provides some simple access to the persistence store present on every MIDP device.

3. Read the code that implements the property viewer application. The main class is PView, supported by PropertyPage and the menu package. This is also an example of retrieving system properties and properties that are set in the "jad" file.

4. Read the code that implements the account management application. The main class is AccountViewer, supported by AccountIndexPage, AccountPage, UserAccount, and the menu and store packages. This is an example of reading and writing information to the persistence store of the device.

5. Now implement the ImageViewer application. This application is fairly simple, and does not have to be implemented using either of the support packages if you don't want to. Note that the original jar file delivered to you in the zip download contains a running version of ImageViewer, so you can experiment with it to see how it works.

a. The main class of the application is ImageViewer, as specified in the jad file. You can change this if you like, but that is what it is when delivered to you.

b. When the application runs, it presents a list of names of images. When the user selects one of the images, it is displayed on the screen. Back and Exit commands are available, and the user can keep looking as long as the images hold his interest.

c. The images are built from gif files that are stored in the jar file when the application is packaged. The sample gifs are provided to you in the res directory.

d. Look at the SimpleHello midlet to see a very basic example of building a List to show on the screen, and also how to attach Command objects to it. The various menu pages in the hw1 midlet also have examples of building a List, adding commands to it, and responding to user actions by implementing the CommandListener interface.

e. Refer to the Learning Wireless Java book, MIDP API documentation or other resource to learn how to create Image and ImageItem objects and display them. An ImageItem is displayed in a Form. A Form can also contain a StringItem which is how I labeled each screen with the path to the gif file stored in the jar file.

f. This application does not have to be fancy at all. For example, in my version, I hard coded the names and paths of the gif files. If you like, you could do something a little fancier using properties set in the jad file. Look at the property viewer code for ideas.

g. Write a brief release-notes.txt file and store that in the main directory of the midlet (the same place as the build file). In the release notes, include a brief description of the features of your midlet. Also describe any known bugs or limitations.

h. The intent of this homework is to get you some experience writing and building a midlet. As long as your midlet runs, lists the three image names, displays the selected image, and is controllable with the device buttons, your midlet will be accepted. Spend the time now to figure out how the parts relate to one another, because in a few short weeks we will be making design decisions about a larger application and you will want to have a good understanding of how it all works then.

Turn in

Build a zip file that contains all the files in your directories (and maintains the directory structure) and contains your uwnetid in the file name. If you are using Ant, the command

```
ant –buildfile build403.xml –Duwnetid=yournetid archive
```

will build the archive. Delete the zip file that doesn't have your netid on it, it doesn't have your ImageViewer code in it! Verify that the zip file does indeed have all your files in it, then follow the turnin link on the web site and turn in the archive.

This homework is due before midnight, Tuesday January 14. Don't be late, the turnin server closes automatically!