# Expenses Management version 1.0 designed for PALM OS operating system

**Stud. Daniela Ilea, Assoc.Prof. PhD Eng. Dan L. Lacrămă**
"Tibiscus" University of Timişoara, Romania

ABSTRACT. This paper is focused on a Java application destined for keeping the expense evidence on a PALM OS operating system. The program was designed in order to be used by average customer and doesn't require any special computers' science expertise.

## 1. Generalities

Since the introduction of the first Palm Pilot in 1996, the Palm OS platform has defined the trends and expectations for mobile computing - from the way people use handheld as personal organizers to the use of mobile information devices as essential business tools, and even the ability to access the Internet or a central corporate database via a wireless connection. Today, after ten years, PALM OS operating system that powers 40 million mobile devices. The popularity of Palm OS has given rise to a huge community of users, enterprises, developers (Palm Os inspired 2000 developers to write programs for it) , manufacturers, wireless operators and infrastructure and middleware providers who together make up the Palm Economy.

The product wants to be a simple and easy to use application, concerning the daily expenses evidence. Its target is the Palm OS average users, not necessary computers' science skilled, who want to keep a clear evidence of day to day expenses.

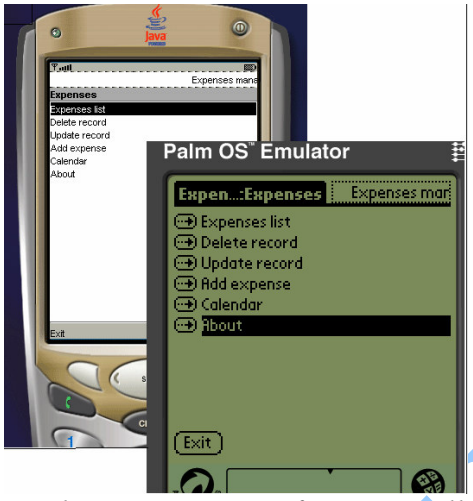


***Fig. 1.1:*** *The application start interface on a cell phone screen and on an Palm OS Emulator*

For developing this application, there are some observations to make:

- The application is developed in Java Micro edition, and it requires the installation of Midp.prc on Palm, the Java virtual machine.
- The Palm allows only prc files (for application) and pdb files (for database file)
- To transfom the .java file in .prc file, I used some tools , downloaded from Sun and Palm : Ktoolbar (to transform .java file into .jad file) and Converter ( to transform .jad file into .prc file)
- The program was installed on PAL OS PDA using its built in synchronization application
- During the application development, it was extremely necessary using the emulators' applications: Palm Emulator (from Palm OS) and Phone (from Sun).
- The application is designed for the model ZIRE 31.

Storing the data is an entire different concept for the programmers used to Microsoft environnement (MS Access, Sql Server). Unlike Microsoft database concepts: databases containing tables with attributes and recorsds, Midlet Records Management Concepts contains the concepts of Recordstore and Records.

Subsequently there are a few general considerations about Midlet Records Management.

The contents of java.microedition.rms MIDlet : Record Management System.

Contents of javax.microedition.rms:

- RecordStore – A collection of records
- RecordComparator – Compares two records in an implementation dependent manner to see if they match
- RecordEnumeration – A birectional RecordStore enumerator
- RecordFilter – Defines filter to specify matching criteria
- RecordListener – Listener for Record changed/deleted/added events

The first concept is Records: records represent the data user inputs throw the application. There are a few observations to make regarding records:

- Records are Array of Bytes
- Each record has unique identifier (integer value)
- Following classes can be used to pack/unpack different data types into and out of Records
  - DataInputStream
  - DataOutputStream
  - ByteArrayInputStream
  - ByteArrayOutputStream

These methods are part of java.io package in CLDC

All records are stored in Recordstore. Their characteristics are:

- Consists of a collection of records
- Persistent across multiple invocations of a MIDlet
- Created in platform-independent locations
- Naming space controlled by MIDlet Suite
- When a MIDlet suite is removed, the associated RecordStore must be removed by the platform
- MIDlet responsible for handling multiple threads accessing a RecodStore
- Time stamped with the last time it was modified
- Maintains a version, that is incremented for each operation that modifies RecordStore contents - Useful in synchronization

RecordStore Operations are:

- Open a record store, optionally create if it doesn't exist – openRecordStore(name, create)

- Remove a record store, deleting all its records – DeleteRecordStore(name)
- Get the names of all record stores known to the MIDlet suite - listRecordStores()

Record Access

- Add a record to the record store – addRecord (byte[] data, int offset,int numBytes)
- Replace the contents of a record – setRecord (int recordId, byte[]newData, int offset, int numBytes)
- Delete a record – deleteRecord(int recordId)
- Retrieve a record – getRecord (int recordId) – getRecord (int recordId, byte[]buffer, int offset)

## 2. The content of the file Expenses.java

Because the application was designed for Zire 31 model, which has only 14 MB capacity, the optimization was a very important consideration when building the application. I have to mention that the application could run on a advanced model of phone too. This application takes only 2 KB from the Palm's hard disk, a very good achievement.

To make the development of this application possible, I had to import the next classes of java.microedition:

- javax.microedition.lcdui.*(contains AlertType, Ticker, Display, List, Textbox),
- javax.microedition.midlet.*( MIDlet, Exception, MiDletStateChangeException),
- javax.microedition.rms.* (contains described already)

and java.io:

- import java.io.ByteArrayOutputStream;
- import java.io.DataOutputStream;
- import java.io.IOException.

The application is build using in most cases the list control. The main menu is a list containing the main options of the user, options available accessing the list's elements.


***Fig. 2.1:*** *The main options*

- Expenses list: a list of all expenses with their total value

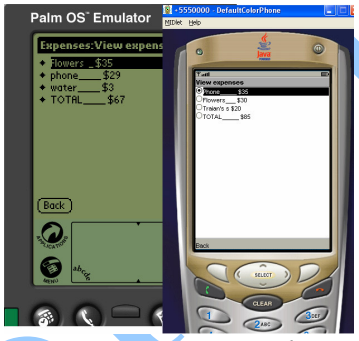


***Fig. 2.2:*** *Expenses list*

- Delete expense: the user has to choose from the list of expenses the record he wants to delete

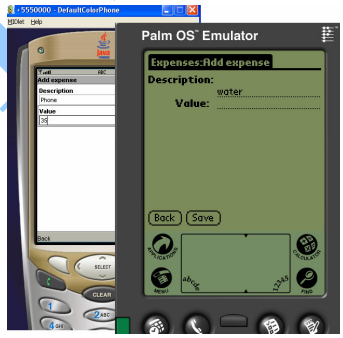


***Fig. 2.3:*** *Add new item to expenses list*


***Fig. 2.5:*** *Insert interface*

- Update record: the user chooses from the list a record he wants to update , the update being made from there in a update form
- Add new expense: the user can add a new expense throw an add form
- Calendar, an option not necessary for the purpose of the application, but considered useful
- About form, an form informing about the software creator and the version of the application

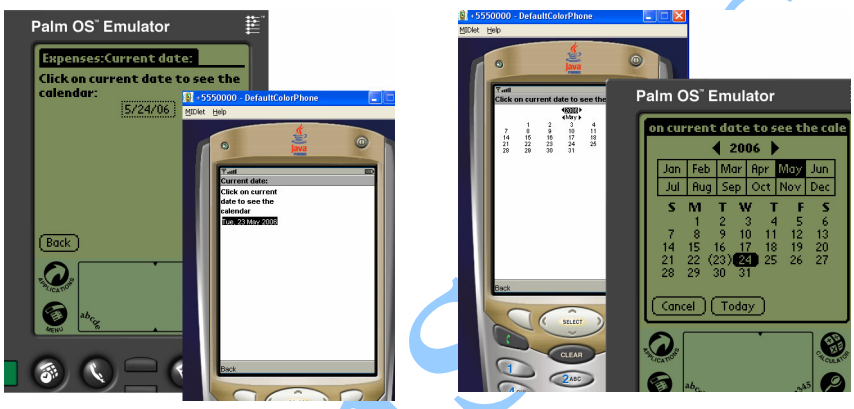


***Fig. 2.5:*** *Calendar*

Quitting every list return the user into the main list, from there the application has an Exit button which allows the user to quit the application. The top of the screen is filled with a ticker control, with animation, not just for esthetical purposes, but also as a title for application. For performance reasons, the application is using only three forms, for About, Add and Update (the same form for both of them) and Calendar. The application uses the Alert control, used to notify the users that not all fields are filled.

The rest of controls are classics, from java.microedition all of them: textboxes and buttons.

## Conclusion

As a conclusion, Palm Os operating system and java make a very good pair. The operating system is a very satisfactory and java.microedition is a perfect choice for programming it.

This small application can be considered as a model for developing a various other programs able to solve practical problems for a large range of customers.

## References

[Bed88] M. A. Bednarczyk - *Categories of asynchronous systems*, PhD thesis, University of Sussex, 1988.

[LD00] L. Luca and I. Despi – *Modele matematice pentru concurență*, Analele Universității "Tibiscus", vol. X: 101-106, Timişoara, 2000

[SNW94] V.Sassone, M. Nielsen and G. Winskel – *Relationships between models of concurrency*, in Proceedings of the REX 3 school and symposium, 1994.