Chapter XII
Advanced Java ME Programming

INTRODUCTION

Chapter XI introduced the basics of Java ME programming. This chapter will build on this, focusing on advanced Java ME programming. The following two major topics will be discussed:

- **Persistent storage**: This refers to the type of storage embedded in handheld devices. Random access memory loses its contents when the power is off, but the contents of persistent storage are preserved even when the power is off. Flash memory, and very occasionally hard disks, are usually used for this kind of storage.

- **Network connection**: Many client-side applications such as weather reports and location-based applications require a network connection. The generic Connection framework of MIDP can be used for this purpose.

Although these two chapters, Chapters XI and XII, introduce Java ME programming to readers, it is not possible to fully cover Java ME programming in only two chapters. For a deeper understanding, readers can consult the section on Java ME References provided later in this chapter for more information.
**PERSISTENT STORAGE**

For some applications, including address books and appointment diaries, it is necessary for the information to be retained even after the application that created them is switched off.

*Persistent (non-volatile) storage is a storage device whose contents are preserved when its power is off.*

Without persistent storage, objects and their states are destroyed when an application closes. If objects are saved to persistent storage, their lifetime is longer than the program that created them and they can later be retrieved for users to continue to work with them. The MIDP *Record Management System (RMS)* is a persistent storage method for MIDlets. It is a simple record-oriented database, which consists of a collection of records. Record stores (binary files) are platform-dependent because they are created in platform-dependent locations, and MIDlets within a single application (a MIDlet suite) can create multiple record stores with different names. The RMS APIs provide the following functionality:

- Allow MIDlets to manipulate records within a record store.
- Allow MIDlets in the same application to share records.

They do not, however, provide a mechanism for sharing records between MIDlets in different applications. Record store names are case sensitive, and cannot be more than 32 characters long. Also, a MIDlet cannot create two record stores with the same name in the same application, although a record store can be created with the same name as a MIDlet in another application. When a new record store is created it is stored under a directory called `appdb`. For example, assume that you are using the *Wireless Toolkit* and that it is installed under the directory `C:\WTK23`. If your record store is `preferences`, the record store is located at the address of the local disk: `C:\WTK23\appdb\DefaultColorPhone\run _ by _ class _ stor- age _ preferences.db`.

**Record Management System**

The MIDP RMS implementation ensures that all individual record store operations are atomic, synchronous, and serialized, so no corruption occurs with multiple access. However, if MIDlets use multiple threads to access a record store, it is the programmer’s responsibility to synchronize this access. The RMS package consists of the following four interfaces, one class, and five exception classes:
Quality Assurance Issues for Big Data Applications in Supply Chain Management
www.igi-global.com/chapter/quality-assurance-issues-for-big-data-applications-in-supply-chain-management/219117?camid=4v1a