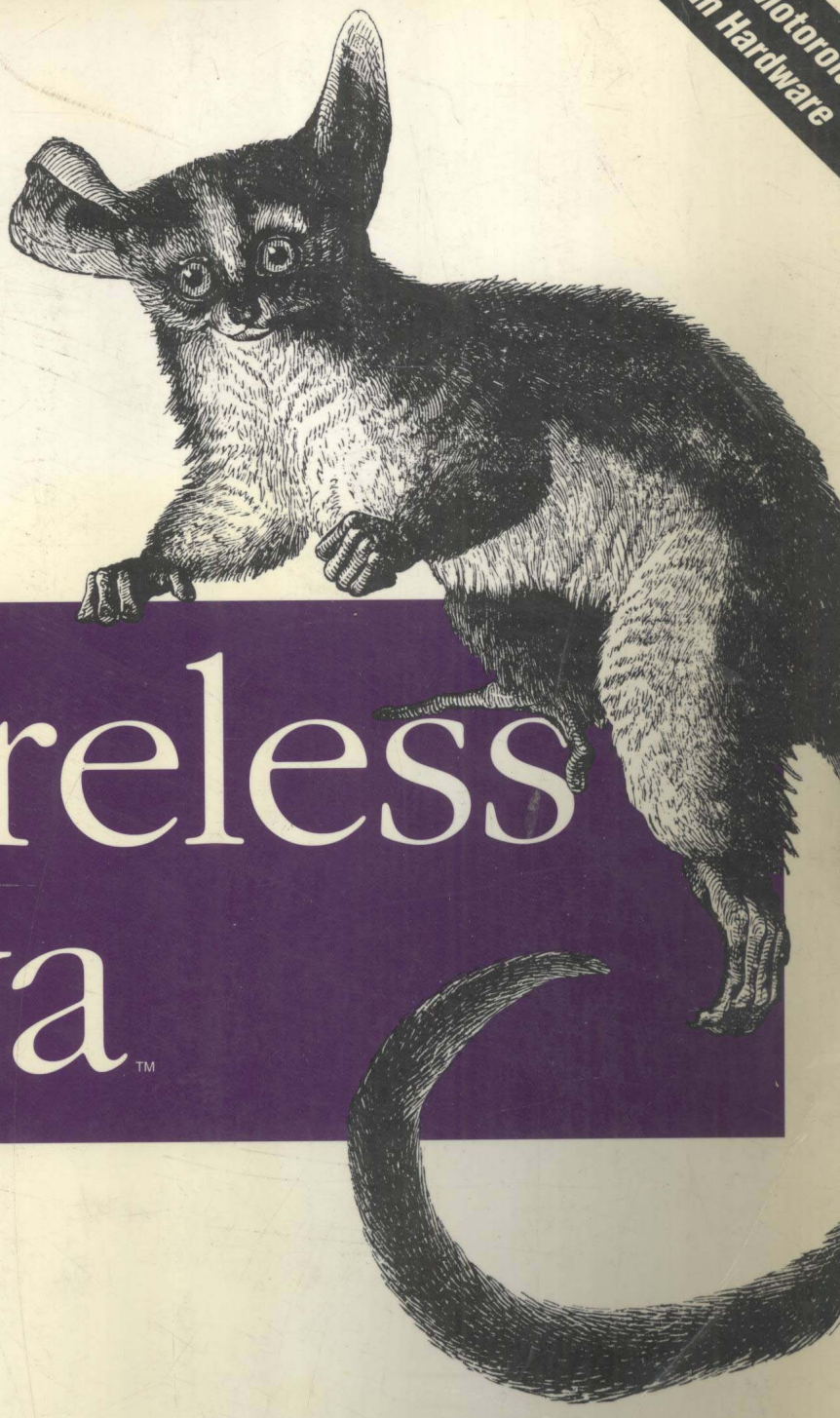
无线Java™入门（影印版）

Covers Motoro
& Palm Hardware



Learning

Wireless Java™

Qusay H. Mahmoud 著

EILLY®

清华大学出版社

无线Java™入门（影印版）

Learning Wireless Java™

Qusay H. Mahmoud

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Paris • Sebastopol • Taipei • Tokyo

O'Reilly & Associates, Inc. 授权清华大学出版社出版

清华大学出版社

图书在版编目（CIP）数据

无线 Java™ 入门：/（美）马哈默德（Mahmoud, H. Q.）著 — 影印版．－北京：清华大学出版社，2002.11
书名原文：Learning Wireless Java
ISBN 7-302-05953-5

Ⅰ.无… Ⅱ.马… Ⅲ.移动通信－Java 语言－程序设计－英文 Ⅳ.TN929.5
中国版本图书馆 CIP 数据核字（2002）第 077173 号

北京市版权局著作权合同登记
图字：01-2002-4632 号

书　　名／ 无线 Java™ 入门（影印版）
书　　号／ ISBN 7－302－05953－5／TP・3542
责任编辑／ 冯志强
封面设计／ Ellie Volckhausen，张健
出版发行／ 清华大学出版社（www. tup. tsinghua. edu. cn）
地　　址／ 北京清华大学学研大厦（邮政编码 100084）
经　　销／ 各地新华书店
印　　刷／ 北京市艺辉印刷有限公司
开　　本／ 787 毫米×980 毫米　16 开本　16. 5 印张
版　　次／ 2002 年 11 月第一版　2003 年 6 月第二次印刷
印　　数／ 1501－2000 册
定　　价／ 99. 00 元（册）

Preface

Most Internet technologies are designed for desktop computers or enterprise servers running on reliable networks with relatively high bandwidth. Handheld wireless devices, on the other hand, have a more constrained computing environment. They tend to have less memory, less powerful CPUs, different input devices, and smaller displays.

Since the mid-1990s, various architectures and protocols have been introduced to deal with these constraints. The Wireless Application Protocol (or WAP), which is a specification developed by the WAP Forum (*http://www.wapforum.org*), takes advantage of several data-handling approaches already in use. Developing wireless applications using WAP technologies is similar to developing Web pages with a markup language (e.g., HTML or XML) because WAP technologies are browser-based.

Another approach to developing wireless applications is to use the Java 2 Platform, Micro Edition (J2ME™). The Java™ programming language already plays an important role in modern programming. With WAP, you can use Java servlets and JavaServer Pages™ to generate Wireless Markup Language (WML) pages dynamically. However, with J2ME, you can now write applications in Java and store them directly on a cell phone. This adds a whole new dimension to wireless programming.

Audience

This book is about programming with J2ME on wireless devices. If you're already familiar with the architecture, you probably noticed that the Connected Limited Device Configuration (CLDC) and the Mobile Information Device Profile (MIDP) classes are not large. Therefore, this book is correspondingly compact in size. The book acts as a quick guide for programmers who are familiar with the Java 2 Standard Edition (J2SE™) and want to get up to speed quickly with the J2ME. We assume that you are familiar with Java programming and have worked with the J2SE

classes. In addition, we assume that you are familiar with setting up Java to work under various environments (Windows or Unix platforms), as well as compiling and running Java applications.

The book also serves as a quick reference for Java programmers who are interested in developing wireless software applications. The examples presented throughout the book are a good starting point for working with all the MIDP features, including user interface, networking, and databases. However, we should point out that this book is *not* a rehash of the entire J2SE class library. Several of the classes of `java.io`, `java.lang`, and `java.net` are included in the CLDC and MIDP libraries, but are less bulky than their J2SE counterparts. We assume that you already know how to use these classes, although we have included them in the API reference for completeness.

Contents of This Book

This book is divided into three parts. Part I, *Introducing Java 2 Platform, Micro Edition (J2ME)*, gives an overview of the J2ME and includes information about its architectural components: namely, configurations and profiles. Part I also presents detailed coverage of the CLDC and the MIDP.

Chapter 1, *Overview of J2ME*
: This chapter introduces the J2ME environment and also explains configurations and profiles. In addition, it shows you how to set up the J2ME Wireless Toolkit to compile, preverify, and run a simple MIDlet using the command line with the Wireless Toolkit emulator.

Chapter 2, *The Connected Limited Device Configuration (CLDC)*
: This chapter discusses the CLDC, including its requirements, limitations, and the differences between its classes and the classes of the J2SE. In addition, it looks briefly at the standalone CLDC and KVM distribution.

Chapter 3, *The Mobile Information Device Profile (MIDP)*
: This chapter introduces the requirements, limitations, and classes of the MIDP, as well as introducing MIDlets and their associated Java Application Descriptor (JAD) files.

Part II, *Programming with the CLDC and the MIDP*, contains programming details of the MIDP. It shows you how to program the phone interface, handle events, make network connections, and work with databases.

Chapter 4, *Working with MIDlets*
: This chapter picks up where Chapter 3 left off, explaining the MIDlet lifecycle methods, the Java application manager, and showing how to use the KToolbar application inside the J2ME Wireless Toolkit to simplify MIDlet development. We also discuss how to deploy MIDlets and include step-by-step instructions on how to download a MIDlet into a Motorola i85s or i50x J2ME-enabled phone.

Chapter 5, *MIDP GUI Programming*
: This chapter introduces the MIDP GUI model and its associated classes. In addition, it gives detailed coverage of both the high-level and low-level MIDP GUI APIs.

Chapter 6, *MIDP Events*
: This chapter continues the discussion of the MIDP GUI APIs by describing how various events take place surrounding the graphical components and commands. In addition, we cover the `CommandListener` and `ItemStateListener` interfaces, as well as low-level event handling.

Chapter 7, *Networking*
: This chapter discusses the Generic Connection Framework provided by the CLDC and shows how to implement an HTTP connection across the Internet, using a MIDlet. The chapter also includes examples of how to send data to CGI scripts and Java servlets across a network. Finally, the chapter briefly discusses wireless session tracking and security for MIDlet data traveling across the airwaves.

Chapter 8, *Database Programming*
: This chapter introduces the concept of data stores, which are simple databases that MIDP applications can use to store persistent data beyond the lifetime of the MIDlet that created them. In addition, the chapter includes a MIDlet that can be used to download stock information from a remote web site.

Chapter 9, *The MIDP for Palm OS*
: This chapter gives a quick introduction to the MIDP implementation on the Palm Connected Organizers, including step-by-step instructions on how to deploy MIDlets to a PalmPilot.

Part III, *API Quick Reference*, contains several chapters that are quick references for the J2ME CLDC and MIDP APIs. There is also an appendix that contains bibliographic information and URLs to J2ME specifications, white papers, wireless software development kits, and other information that is important to developers.

Conventions Used in This Book

This book uses the following typographical conventions:

A `Constant Width` font is used for:

- Anything that might appear in a Java program, including keywords, data types, constants, method names, objects, variables, class names, and interface names
- All Java code examples
- Attributes that might appear in a manifest or JAD file

An *italic* font is used for:

- New terms where they are defined
- Pathnames, filenames, directory names, and program names (unless the program name is the name of a Java class; then it appears in constant width, like other class names)
- Internet addresses, such as domain names, URLs, and email addresses

A **boldface** font is used for:

- Example lines of Java code to which we wish to draw attention

Comments and Questions

The information in this book has been tested and verified, but you may find that features or libraries have changed, or you may even find mistakes. You can send any errors you find, as well as suggestions for future editions, to:

O'Reilly and Associates, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
(800) 998-9938 (in the United States or Canada)
(707) 829-0515 (international/local)
(707) 829-0104 (fax)

You can also send electronic messages. To be put on the mailing list or to request a catalog, send email to:

info@oreilly.com

To ask technical questions or comment on the book, send email to:

bookquestions@oreilly.com

I would be pleased to receive feedback on this book. You can contact me by email at:

qmahmoud@javacourses.com

The O'Reilly web site for this book is located at *http://www.oreilly.com/catalog/wirelessjava* and contains all the source examples for this book.

In addition, we have created another web site, *http://www.javacourses.com/wireless*, that includes links to material that supports the use of this book for training and personal study. This web site provides the following supplements:

- Additional source code for new applications
- Links to online J2ME material, and information on other related books
- J2ME tips and tricks

- A set of overhead projector transparencies for instructors interested in using the book in their training courses
- Up-to-date information on topics presented in the book

Acknowledgments

I am deeply grateful to my editor, Robert Eckstein, for all his comments, suggestions, and guidelines throughout the development of this book. I did not know about all the contributions an editor can make to a book until I worked with Bob. Thanks, Bob! Thanks also to the production team at O'Reilly for their hard work on this book.

Special thanks also to Monica Pawlan, Jenny Pratt, Dana Nouri, and Laureen Hudson of the Java Developer Connection (JDC), who either provided comments or edited some of the examples used in this book when they first appeared on the JDC. Also, thanks to the thousands of JDC members who sent in comments and suggestions regarding my articles. Thanks also to the following people who reviewed the contents of this book for accuracy: Ben Griffin, Marc Loy, and Jeff Cunningham.

I would also like to thank my family for their support during my studies, especially my brother, Dr. Mohammad H. Hamdan, for teaching me the value of hard work.

Finally, thanks to my wife, Reema, for her love, support, tolerance, and coffee, and my baby son Yusef, who was born on October 14, 2001, for providing a fun home environment while I finished this book.

Table of Contents

PART I

Introducing Java 2 Platform, Micro Edition (J2ME)

Part I is an introduction to the Java 2 Micro Edition (J2ME) and J2ME programming. These chapters will give you an overview of the J2ME, and quickly teach you everything you need to know to get started with J2ME programming.

Chapter 1, *Overview of J2ME*

Chapter 2, *The Connected Limited Device Configuration (CLDC)*

Chapter 3, *The Mobile Information Device Profile (MIDP)*

CHAPTER 1

Overview of J2ME

This book is about wireless Java programming with the Java 2 Platform, Micro Edition (J2ME). Sun Microsystems, Inc. introduced J2ME at the JavaOne conference in June 1999 as the younger sibling of both the Java 2 Standard Edition (J2SE) and the Java 2 Enterprise Edition (J2EE). At the time, distributed programming was taking the Java developer community by storm, so most of the participants at the show were more interested in what J2EE had to offer. However, over the next two years, developers also realized that there was tremendous value in having small components running Java. Two years later, at the 2001 JavaOne conference, Sun devoted an entire track for individuals seeking to master the once arcane J2ME. Luckily, you don't need to attend JavaOne to learn about J2ME. Instead, this book will help you through the myriad details of understanding J2ME architecture and programming J2ME applications.

In this chapter, we will present an overview of J2ME's primary components, including virtual machines, configurations, and profiles. We'll then present a few short examples of J2ME-enabled applications to whet your appetite and to show you how easy it is to get started with J2ME.

What Is J2ME?

J2ME is a version of Sun Microsystems' Java that is aimed at the consumer and embedded devices market, which includes electronic commodities such as cellular telephones, pagers, Personal Digital Assistants (PDAs), set-top boxes, and other small devices. Since its release, over 600 companies have joined the development effort, including large corporations such as Palm, Nokia, Motorola, and RIM. However, the direction that J2ME travels is not shrouded in secrecy behind closed corporate doors. Instead, development of J2ME is handled through the Java Community Process (JCP), which allows anyone with an Internet connection to get involved.

J2ME provides a complete set of solutions for creating state-of-the-art networked applications for small devices. It also promises to enable device manufacturers, service providers, and application developers to deploy new applications and services to

their customers. However, in doing so, it does not sacrifice some of the founding guidelines of Java, which have become increasingly important these days, namely cross-platform compatibility and security.

A High-Level View

From a high-level view, J2ME defines the following components:

- A series of Java virtual machines, each for use on different types of small devices, each with different requirements
- A group of libraries and APIs that can be run under each of the virtual machines; these are known as *configurations* and *profiles*
- Various tools for deployment and device configuration

The first two components make up the *J2ME runtime environment*. Figure 1-1 provides a relational view of the runtime environment. At its heart is a Java virtual machine, which runs on top of a device's host operating system. Above that is a specific J2ME configuration, which consists of programming libraries that provide basic functionality based on the resource requirements of the device. On top of the configuration are one or more J2ME profiles, which are additional programming libraries that take advantage of kindred functionalities on similar devices.

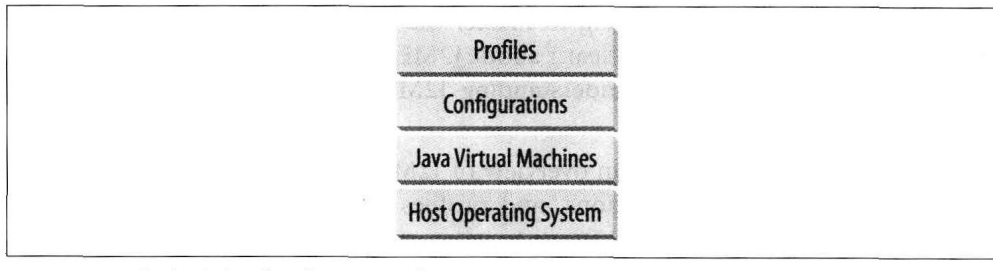


Figure 1-1. The high-level architecture of J2ME runtime environment

If you haven't worked with J2ME before, you're probably wondering about the top two layers. It's important to distinguish between a configuration and a profile in the J2ME world, so let's introduce them now.

Configurations

Cellular telephones, pagers, organizers, and other small devices are diverse in form, functionality, and feature. However, they often use similar processors and have similar amounts of memory. For these reasons, the J2ME designers created *configurations*. Configurations define a horizontal grouping of products based on the available memory budget and processing power of each device. Once this information is known, the configuration then outlines the following:

- The Java programming language features supported
- The Java virtual machine features supported
- The basic Java libraries and APIs supported

Currently, there are two standard configurations in the J2ME world: the *Connected Limited Device Configuration* (CLDC) and the *Connected Device Configuration* (CDC). Let's look at the CDC first.

The CDC

The CDC is targeted toward powerful devices that are intermittently connected to a network, including set-top boxes, Internet TVs, home appliances, and car navigation systems. The CDC contains a full-featured Java virtual machine, similar to that in use today with the J2SE. The difference lies in the respective devices' memory and display capabilities.

Here are the resource requirements for CDC devices, as given by the official J2ME specifications:*

- The device is powered by a 32-bit processor.
- The device has 2 megabytes or more of total memory available for Java. This includes both RAM and flash memory or ROM.
- The device requires the full functionality of the Java 2 "Blue Book" virtual machine.
- The device has connectivity to some kind of network, often with a wireless, intermittent connection and with limited (often 9600 bps or less) bandwidth.
- The device may have a user interface with some degree of sophistication, but a user interface is not mandatory.

The CLDC

The second type of configuration is more prevalent in the J2ME world: the CLDC. This configuration specifies a much smaller footprint for consumer and embedded devices than the CDC. The CLDC was first distributed in October 1999 with the idea of creating a "lowest common denominator" Java platform for embedded devices, specifically in terms of networking, I/O, security, and core libraries. Today, some of the devices that you might find powered by the CLDC include mobile cell phones, two-way pagers, personal digital assistants (PDAs), and personal organizers.

Here are the requirements for the J2ME CLDC, again from the official J2ME specifications:*

* The J2ME CDC specifications are located on the Java Community Process web site as JSR-36, which can be found at *http://www.jcp.org/jsr/detail/36.jsp*.

- The device can have between 160 and 512 kilobytes of total memory available for the Java platform, including both RAM and flash memory or ROM.
- The device can have limited power, such as battery-powered operation.
- The device has connectivity to some kind of network, often with a wireless, intermittent connection and with limited (often 9600 bps or less) bandwidth.*
- In addition, the device may have a user interface with some degree of sophistication, but a user interface is not mandatory.

The two products' configurations, along with some of their respective products, are shown in Figure 1-2.

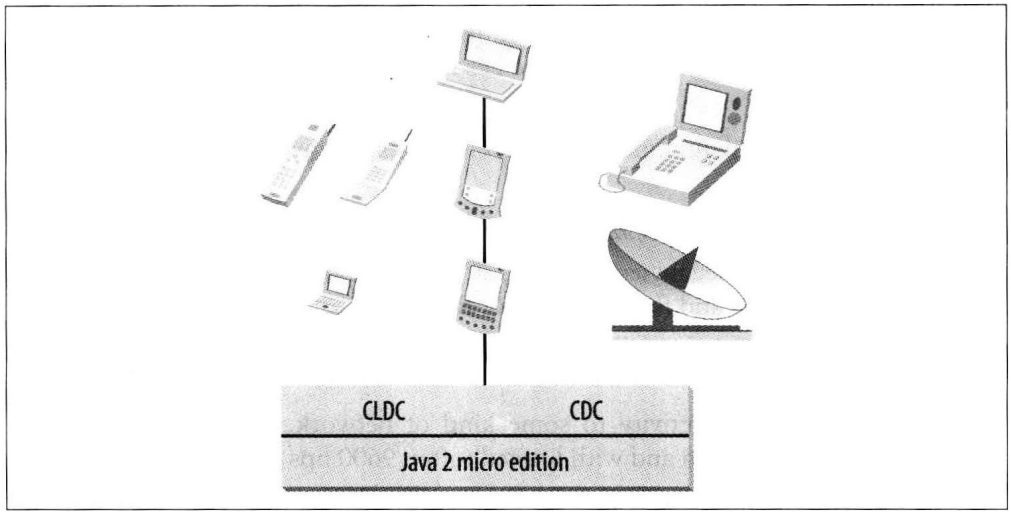


Figure 1-2. J2ME architecture

Note that although the two product groups are supported by different configurations, the line between the two configurations is somewhat blurred. In the future, technological advances will likely make this boundary more and more cloudy. However, for the moment, the important thing to remember is that the boundary between the CLDC and the CDC is defined in terms of the target device's memory budget, battery usage, and the presence or absence of a user interface.

Virtual Machines

As mentioned above, the CLDC and CDC configurations each define their own set of supported features from the Java virtual machine. Consequently, each requires its own Java virtual machine. The CLDC virtual machine is far smaller than the virtual

* Note that CLDC stands for *Connected Limited Device Configuration*, not Connectivity-Limited Device Configuration. The difference between the CLDC and the CDC is not in the type or speed of the network connection.