



工业和信息化部普通高等教育“十二五”规划教材立项项目

21世纪高等教育计算机规划教材



Java ME 程序设计

Java ME Programming

■ 孙更新 宾晟 编著

— 讲解详细，实例通俗易懂

— 内容不流于表面，而是从最佳实践角度入手

— 章节最后配备课程设计案例和上机习题



人民邮电出版社
POSTS & TELECOM PRESS



工业和信息化部普通高等教育“十二五”规划教材立项项目

21世纪高等教育计算机规划教材

COMPUTER

Java ME 程序设计

Java ME Programming

■ 孙更新 宾晟 编著



人民邮电出版社

北京

图书在版编目 (C I P) 数据

Java ME程序设计 / 孙更新, 宾晟编著. -- 北京 :
人民邮电出版社, 2012.10
21世纪高等教育计算机规划教材
ISBN 978-7-115-28981-0

I. ①J… II. ①孙… ②宾… III. ①
JAVA语言—程序设计—高等学校—教材 IV. ①TP312

中国版本图书馆CIP数据核字(2012)第211147号

内 容 提 要

本书循序渐进地介绍了 Java ME 开发技术。全书深入分析了 Java ME 的核心知识, 包括配置、简表、Java ME 的图形用户界面、游戏编程、记录存储、无线数据传输等。此外, 通过丰富典型的案例, 从实践的角度展示了如何更好地使用 Java ME 开发手机应用程序。本书最后的两个综合游戏开发案例更是将全书的内容进行总结, 使读者对 Java ME 技术融会贯通。

本书内容全面、实例丰富、易于理解, 每章的内容都不流于表面, 而是从最佳实践的角度入手, 为读者更好地使用 Java ME 开发手机应用程序提供了很好的指导。本书适合高等院校计算机科学、软件工程、数字媒体技术、通信及相关专业的本科和专科教师及学员作为 Java ME 移动开发相关课程教材使用, 也是学习和正在从事无线应用系统开发人员的优秀教材和参考书籍。

21 世纪高等教育计算机规划教材

Java ME 程序设计

-
- ◆ 编 著 孙更新 宾 晟
责任编辑 刘 博
 - ◆ 人民邮电出版社出版发行 北京市崇文区夕照寺街 14 号
邮编 100061 电子邮件 315@ptpress.com.cn
网址 <http://www.ptpress.com.cn>
中国铁道出版社印刷厂印刷
 - ◆ 开本: 787×1092 1/16
印张: 22 2012 年 10 月第 1 版
字数: 580 千字 2012 年 10 月北京第 1 次印刷

ISBN 978-7-115-28981-0

定价: 45.00 元

读者服务热线: (010)67170985 印装质量热线: (010)67129223

反盗版热线: (010)67171154

[illegible]

本书共 12 章。

第1章 Java ME 概述, 主要介绍 Java ME 编程中的一些基本概念, 包括 Java ME 的定义和体系结构以及 KVM、CLDC、MIDP 等概念。并在此基础上详细介绍了 MIDlet 程序的相关概念和其编译运行的基本步骤。第2章 Java ME 开发环境配置, 主要介绍了各种常用 Java ME 开发环境的安装和配置过程, 以及利用这些开发工具创建和运行 Java ME 应用程序的详细步骤。第3章 高级用户界面, 主要介绍了开发简单的高级人机交互界面所使用的各种常用组件的基本使用方法。第4章 低级用户界面, 主要介绍了开发低级人机交互界面所使用的各种常用组件的基本使用方法。第5章 记录存储管理系统 (RMS), 主要涉及 MIDlet 如何使用记录管理系统对数据进行存储和管理。第6章 Java ME 中的网络应用开发, 主要介绍如何开发基于 HTTP 和 Socket 编程的网络应用程序。第7章 Java ME 中的无线数据传输应用开发, 主要介绍红外和蓝牙数据传输这两种目前应用最广泛的无线传输技术, 详细介绍了使用这两种无线开发技术进行通信的具体编程方法。第8章 Java ME 中的短消息应用开发, 主要介绍如何使用 Java ME 编程实现短消息的发送和接收, 并在此基础上进一步介绍发送、接收二进制消息和多媒体消息的方法。第9章 Java ME 中的多媒体应用开发, 主要介绍 Java ME 中的多媒体编程, 通过对播放音频和视频文件等实际开发中经常用到的基本功能的介绍, 使读者对多媒体开发有一定的了解。第10章 Java ME 中的游戏开发, 主要介绍 MIDP 2.0 中 Game API 开发包的使用, 通过实例让读者轻松掌握如何使用 MIDP 2.0 进行手机游戏开发。第11章 Java ME 中的 3D 程序开发, 介绍如何在手机中使用 Mobile 3D Graphics API (JSR184) 显示 3D 图形并进行控制。第12章 角色扮演类游戏开发, 以一款基于 Java ME 最新 MIDP 2.0 标准开发的手机角色扮演类游戏为例, 详细介绍了标准的 Java ME 商业化应用程序的开发过程及技术规范。

由于时间仓促和作者水平有限,书中难免会有纰漏和不足之处,恳请各位专家同仁和读者批评指正。

2012年6月

2012年6月

目 录

第 1 章 Java ME 概述.....1

1.1 Java ME 背景介绍.....1

1.1.1 Java ME 简介.....1

1.1.2 Java ME 体系结构.....2

1.1.3 Java ME 中的虚拟机.....3

1.1.4 Java ME 配置.....3

1.1.5 Java ME 简表.....4

1.2 Java ME 配置简介.....4

1.2.1 CDC 概述.....4

1.2.2 CLDC 概述.....5

1.3 MIDP 简介.....6

1.3.1 MIDP 设备.....7

1.3.2 MIDP 体系结构.....7

1.3.3 MIDP 2.0 新特性.....8

1.4 MIDlet 简介.....9

1.4.1 MIDlet 生命周期.....9

1.4.2 MIDlet 程序结构.....10

1.4.3 MIDlet 程序中的事件处理.....11

1.4.4 MIDlet 套件.....12

课后小结.....14

习题.....14

第 2 章 Java ME 开发

环境配置.....15

2.1 Java 无线开发工具包.....15

2.1.1 Java WTK 的安装.....15

2.1.2 Java WTK 目录结构.....17

2.1.3 Java WTK 项目开发.....17

2.1.4 Java WTK 项目管理.....21

2.1.5 Java WTK 项目发布.....21

2.2 Eclipse+EclipseME 集成

开发工具.....27

2.2.1 搭建 Eclipse+EclipseME 开发

平台.....27

2.2.2 使用 Eclipse+EclipseME 开发

MIDlet 程序.....31

课后小结.....33

习题.....33

第 3 章 高级用户界面.....34

3.1 MIDP 高级用户界面设计概述.....34

3.2 Display 类和 Displayable 类.....35

3.3 高级屏幕对象.....38

3.3.1 TextBox 文本框.....38

3.3.2 List 列表.....41

3.3.3 Alert 消息框.....44

3.3.4 Form 表单.....51

3.3.5 Ticker 滚动条.....53

3.4 表单中的 Item 组件.....55

3.4.1 StringItem.....55

3.4.2 TextField.....58

3.4.3 ImageItem.....60

3.4.4 DateField.....63

3.4.5 Gauge.....65

3.4.6 ChoiceGroup.....70

3.4.7 CustomItem.....72

3.4.8 Spacer.....75

3.5 高级事件处理.....77

3.5.1 Command 软键.....77

3.5.2 CommandListener 软键事件.....79

3.5.3 ItemStateListener 组件事件.....81

课后小结.....83

习题.....83

第 4 章 低级用户界面.....84

4.1 Canvas 画布屏幕设计.....84

4.1.1 Canvas 概述.....84

4.1.2 画布屏幕大小.....85

4.1.3 画布屏幕的绘制和重绘.....87

4.1.4 Canvas 屏幕可视性通知	90
4.2 Canvas 事件处理	92
4.2.1 按键事件	92
4.2.2 指针事件	95
4.2.3 游戏动作	98
4.3 图形绘制与 Graphics 类	101
4.3.1 Graphics 类概述	101
4.3.2 颜色模型	101
4.3.3 坐标系统	102
4.3.4 图形剪裁	105
4.3.5 绘制几何图形	106
4.4 文字绘制与 Font 类	109
4.4.1 Font 类概述	109
4.4.2 绘制文本	109
4.4.3 字体属性	110
4.5 图像处理与 Image 类	113
4.5.1 Image 类概述	113
4.5.2 绘制不变图像	113
4.5.3 绘制可变图像	116
课后小结	118
习题	118

第 5 章 记录存储管理系统 RMS

5.1 记录存储系统概述	119
5.2 记录存储系统类库	120
5.3 记录存储系统基本操作	121
5.3.1 创建和打开记录存储	121
5.3.2 关闭和删除记录存储	121
5.3.3 增加记录	123
5.3.4 获取记录	124
5.3.5 修改记录	124
5.3.6 删除记录	125
5.4 记录存储系统高级操作	128
5.4.1 记录存储枚举接口	128
5.4.2 记录存储过滤接口	128
5.4.3 记录存储比较接口	132
5.4.4 记录存储监听接口	133
课后小结	134
习题	134

第 6 章 Java ME 中的网络应用

开发

6.1 基于 HTTP 的网络开发	135
6.1.1 HTTP 概述	135
6.1.2 建立 HTTP 连接	136
6.1.3 使用 HTTP 连接	138
6.2 基于 Socket 和数据报的网络开发	148
6.2.1 Socket 和数据报编程概述	148
6.2.2 Socket 编程	149
6.2.3 数据报编程	158
课后小结	160
习题	160

第 7 章 Java ME 中的无线数据

传输应用开发

7.1 无线数据传输概述	161
7.1.1 红外数据传输概述	161
7.1.2 蓝牙数据传输概述	162
7.2 通过 IrOBEX 进行红外通信	162
7.2.1 IrOBEX 协议概述	162
7.2.2 IrOBEX 开发步骤	163
7.3 通过 JABWT 进行蓝牙通信	171
7.3.1 蓝牙协议栈	171
7.3.2 通过 RFCOMM 实现蓝牙应用	172
7.4 课程设计：基于蓝牙通信的五子棋	
游戏	190
7.4.1 游戏功能分析	190
7.4.2 游戏具体实现	191
课后小结	203
习题	203

第 8 章 Java ME 中的短消息

应用开发

8.1 无线消息概述	204
8.1.1 GSM 短消息服务	204
8.1.2 GSM 小区广播服务	205
8.2 WMA 概述	205
8.2.1 WMA 简介	205

8.2.2 WMA 类库简介.....	206	10.4.1 帧动画.....	260
8.3 使用 WTK 中的 WMA 控制台.....	206	10.4.2 帧序列.....	260
8.3.1 配置和启动 WTK 中的 WMA 控制台.....	207	10.4.3 精灵翻转.....	264
8.3.2 使用 WMA 控制台发送文本 消息.....	208	10.4.4 参考像素点.....	269
8.3.3 WMA 控制台发送小区广播.....	209	10.4.5 碰撞检测.....	270
8.3.4 WMA 控制台发送多媒体消息.....	210	10.5 游戏背景 TiledLayer 类.....	270
8.4 使用 WMA 编写收发短消息程序.....	211	10.5.1 图像贴图和单元格.....	271
8.4.1 发送和接收 SMS 消息.....	211	10.5.2 制作静态地图背景.....	272
8.4.2 发送和接收二进制消息.....	217	10.5.3 制作动态地图背景.....	273
8.4.3 发送和接收 MMS 消息.....	222	10.6 图层管理 LayerManager 类.....	278
课后小结.....	228	10.6.1 图层的索引.....	278
习题.....	228	10.6.2 图层的位置与绘制.....	279
		10.6.3 设置视图窗口.....	279
		10.7 课程设计: 疯狂坦克游戏.....	280
		10.7.1 游戏程序结构和设计思想.....	280
		10.7.2 游戏程序分析和具体实现.....	281
		课后小结.....	296
		习题.....	296
第 9 章 Java ME 中的多媒体 应用开发.....	229	第 11 章 Java ME 中的 3D 程序 开发.....	297
9.1 MM API 概述.....	229	11.1 手机 3D 开发概述.....	297
9.2 媒体管理类 Manager.....	229	11.2 3D 模型的建立.....	298
9.3 播放器接口 Player.....	230	11.2.1 顶点数组.....	299
9.3.1 播放器状态.....	231	11.2.2 顶点缓冲和索引缓冲.....	299
9.3.2 播放器监听接口.....	234	11.2.3 外观属性.....	299
9.4 媒体控制接口 Control.....	235	11.2.4 多面体模型的建立.....	300
9.4.1 获取控制器.....	235	11.3 3D 场景的创建.....	300
9.4.2 音调控制接口.....	235	11.3.1 世界 World 类.....	301
9.4.3 音量控制接口.....	237	11.3.2 摄像机 Camera 类.....	301
9.4.4 视频控制接口.....	238	11.3.3 光线 Light 类.....	305
9.5 实现手机拍照功能.....	243	11.3.4 背景 Background 类.....	305
课后小结.....	249	11.3.5 场景中的坐标转换.....	306
习题.....	249	11.3.6 场景的绘制.....	306
		11.4 外部建模导入 3D 模型.....	310
		课后小结.....	312
		习题.....	312
第 10 章 Java ME 中的游戏 开发.....	250	第 12 章 角色扮演类游戏开发.....	313
10.1 Game API 概述.....	250	12.1 游戏创意.....	313
10.2 游戏容器 GameCanvas 类.....	251		
10.2.1 屏幕缓冲.....	251		
10.2.2 获取键盘输入状态.....	252		
10.2.3 实现游戏主循环.....	253		
10.3 图层 Layer 类.....	259		
10.4 游戏精灵 Sprite 类.....	259		

12.2 游戏的流程和模块结构	314	12.3.2 游戏的主角类实现	318
12.2.1 游戏基本运行流程	314	12.3.3 游戏的地图类实现	319
12.2.2 游戏主要模块	314	12.3.4 游戏的画布屏幕类实现	320
12.3 游戏功能的实现	315	12.3.5 游戏 MIDlet 类	341
12.3.1 游戏的闪屏类实现	315	12.4 游戏运行展示	344
		课后小结	344

第 1 章

Java ME 概述

学习目标

Java ME 作为目前手机应用程序开发的主要编程语言，受到越来越多开发人员的重视。本章将详细讲解 Java ME 编程中的一些基本概念，包括 Java ME 的定义、体系结构以及 KVM、CLDC、MIDP 等内容。并在此基础上详细介绍了 Java ME 的 MIDlet 应用程序的相关概念和其编译运行的基本步骤。

内容摘要

- Java ME 的体系结构
- CLDC 配置的概念和功能
- MIDP 简表的的概念和功能
- MIDlet 生命周期
- MIDlet 程序基本结构

1.1 Java ME 背景介绍

随着无线网络的普及以及移动信息设备的发展，我们已经进入普及计算时代。当手机可以随时随地无线上网，PDA 可以和个人 PC 同步，车载 PC 可以获得网上 GPS 地图数据时，这些孤立的设备才能够发挥最大的作用。而平台无关性和网络功能正是普及计算的设计目标和基本要求。

Java 语言的“一次编写，到处运行”的平台无关性，以及对网络 and 多重设备间联网的机制，显然对这个领域是至关重要的软件条件。这个领域内的 Java 标准就是 Java ME。Java ME 以往称作 J2ME (Java Platform, Micro Edition)，是为机顶盒、移动电话和 PDA 之类嵌入式消费电子设备提供的 Java 语言平台，包括虚拟机和一系列标准化的 Java API。它和 Java SE、Java EE 一起构成 Java 技术的三大版本，并且同样是通过 JCP (Java Community Process) 制订的。

1.1.1 Java ME 简介

根据 Sun 公司的定义：Java ME 是一种高度优化的 Java 运行环境，主要针对消费类电子设备的，例如蜂窝电话和可视电话、数字机顶盒、汽车导航系统等。JAVA ME 技术在 1999 年的 Java One Developer Conference 大会上正式推出，它将 Java 语言的与平台无关的特性移植到小型电子设备

上, 允许移动无线设备之间共享应用程序。

在消费电子和嵌入式设备领域, 不同设备和领域之间的硬件配置差别极大, 内存从几百 KB 到几十 MB, 从没有屏幕到 Web-TV, CPU 从低功耗的嵌入式处理器到多核的几 GHz 的 RISC 处理器。这就使得 Java ME 的标准需要有不同的层次和类别来适应这些具有极大差异的设备和领域。因此, 相比之下 Java ME 就比 Java SE 和 Java

EE 的标准要复杂得多。Java SE、Java EE 和 Java ME 3 个版本的不同应用场合如图 1-1 所示。

Java ME 在适用于这些设备的同时也保留了 Java 的传统特性, 即代码具有可移植性、部署灵活性、安全的网络传输性和代码的稳定性等。因此, Java ME 目前已经被广泛地应用于消费类嵌入式设备中, 成为了嵌入式开发中的一种标准。

1.1.2 Java ME 体系结构

与 Java SE 和 Java EE 相比, Java ME 总体的运行环境和目标更加多样化, 但其中每一种产品的用途却更为单一, 而且资源限制也更加严格。为了在达到标准化和兼容性的同时尽量满足不同方面的需求, Java ME 的架构分为 Java 虚拟机 (Java VM)、配置 (Configuration)、简表 (Profile) 和可选包 (Optional Packages)。它们的不同组合形成了不同的运行环境。

- Java 虚拟机, 针对不同的嵌入式设备以及为了满足不同的需要, Java ME 提供了 KVM 和 CVM 两大类虚拟机。
- 配置, 它是将某一类特定应用范畴的设备的共性提取出来形成的规范。它为开发人员提供了一个基础的、能在所有平台上通用的核心 API 类库。
- 简表, 它是将某一个特殊行业或领域内的设备的特性提取出来, 形成对特定领域的特性的总结, 简表是作为配置的一个补充。
- 可选包, 独立于前面两者, 提供附加的、模块化的和更为多样化的功能。目前标准化的可选包主要包括数据库访问、多媒体、蓝牙等。

其中前三项组成了 Java ME 的基本运行环境, 它们的关系如图 1-2 所示。

其中 Java 虚拟机是运行环境的核心, 其运行于设备中的操作系统之上, 再往上是具体的 Java ME 配置, 包括根据设备的资源需要而提供的基本核心 API 类库, 配置的上层是一个或者多个针对不同应用领域而设置的简表。



图 1-1 Java 3 个版本的不同应用场合

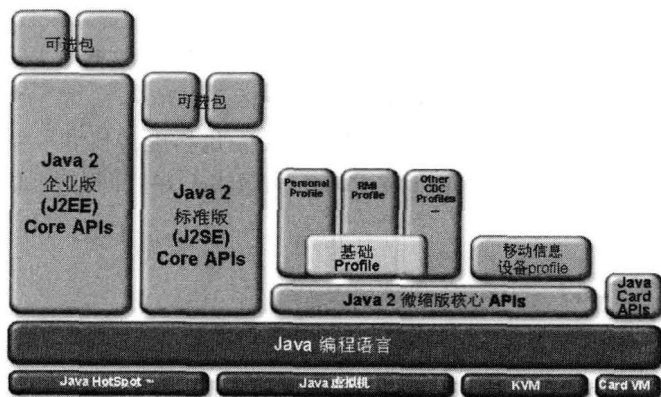


图 1-2 Java ME 平台的体系结构

1.1.3 Java ME 中的虚拟机

JVM (Java Virtual Machine, Java 虚拟机) 是可运行 Java 代码的虚拟计算机。只要根据 JVM 规格描述将解释器移植到特定的计算机上, 就能保证经过编译的任何 Java 代码能够在该系统上运行。它也是实现 Java 语言“一次编写, 到处运行”的关键技术。

那么在支持 Java ME 的移动信息设备中是否需要 JVM 呢? 答案是肯定的, 因为移动信息设备和普通 PC 一样, 也需要 Java 虚拟机作为操作系统和 Java 应用程序交互的接口层。但是由于移动信息设备内存空间大小的限制, 在 Java ME 中需要针对不同的应用场合提供特殊的虚拟机。在 JAVA ME 中一般还有 KVM (K Virtual Machine, K 虚拟机) 和 CVM (C Virtual Machine, C 虚拟机) 两类虚拟机, 这两类虚拟机在功能上都是 JVM 的缩减版, 这两类虚拟机的适用范围并不相同, KVM 的功能比 CVM 的功能稍弱。

最初 CVM 是“Compact Virtual Machine”的缩写, 后来 Sun 公司意识到开发者可能混淆“Compact”的发音和 KVM 中的“K”的发音, 因此现在 C 不代表任何含义了。该虚拟机主要用于存储容量较大的嵌入式设备, 例如, 可视电话、POS 收款机等。

KVM 是最小开发环境的一部分, 包含在 Java ME 软件中; KVM 使用 C 语言编写, 适用于那些内存和 CPU 能力有限的设备。这里的“K”表示千字节, 说明 KVM 适合运行在 (几) 千字节的内存中。例如, 移动电话、寻呼机和个人数字助理 (PDA) 中通常会使用 KVM, 用以提供常见的计算功能。

KVM 和 JVM 相似, 因为它也是执行使用 Java 技术撰写的应用程序与 Applet 的引擎。只不过 KVM 在移动电话和移动设备上使用, 而 JVM 在普通 PC 上使用。

Java ME 技术使用包括 KVM 和 CVM 在内的不同大小的虚拟机。其中 KVM 适用于有限连接设备, 而 CVM 和 JVM 适用于连接设备。

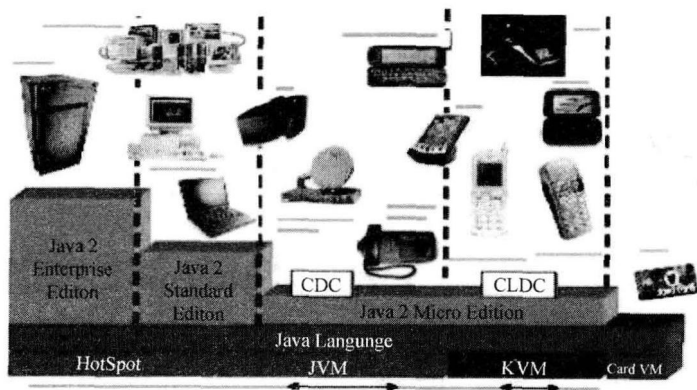


图 1-3 Java 3 个版本的虚拟机

Java 语言不同版本所使用的虚拟机及其适用设备如图 1-3 所示。

1.1.4 Java ME 配置

配置是将基本运行的环境定义为一套核心类和一个运行在特定类型设备上的特定的虚拟机。Java ME 中提供了两种配置: CLDC (Connected Limited Device Configuration) 和 CDC (Connected Device Configuration)。这两种配置所针对的设备的 CPU 和内存大小是不一样的, 因此它们所使用的虚拟机和核心类库也不相同。

Java ME 根据设备的硬件配置把设备分成两类: 连接设备 (Connected Device) 和有限连接设备 (Connected Limited Device)。

连接设备主要是指 32 位及 75MHz 以上的处理器、具有 2 MB 以上内存的嵌入式设备; 有限连接设备主要是指 16/32 位 RISC/CISC 微处理器或控制器, 并且其可用内存为 160 KB~512 KB

的嵌入式设备。

CDC 针对的设备主要是数字电视、机顶盒、网络电话、车载计算设备等连接设备的配置，这些设备的特点是有线连接，稳定而持续的电源供应，设备资源比较受限。CDC 使用 CVM 虚拟机，具有较强的处理能力和较大的核心类库。

CLDC 支持的设备主要包括无线电话、手机、个人数字助手（PDA）以及小型零售支付终端等。CLDC 使用 KVM 虚拟机，其处理能力较弱，并且只包含程序运行所需要的最小类库。

1.1.5 Java ME 简表

简表是建立在配置基础之上，并和配置一起构成了完整的运行环境。它对设备横向分类，针对特定领域细分市场，内容主要包括特定用途的类库和 API。CLDC 上已经标准化的简表有 MIDP（Mobile Information Device Profile）和 IMP（Information Module Profile），而 CDC 上标准化的简表有 FP（Foundation Profile）、PBP（Personal Basis Profile）和 PP（Personal Profile）。

简表为相同消费电子设备的不同生产厂商提供了标准化的 Java 类库。简表的实现是 Java 应用程序接口的一个集合，用于适应被定义配置的应用程序接口提供的服务，简表是一个完整的运行环境，一个在简表上执行的应用程序不需要额外的支持类。

配置和简表的层次关系如图 1-4 所示。

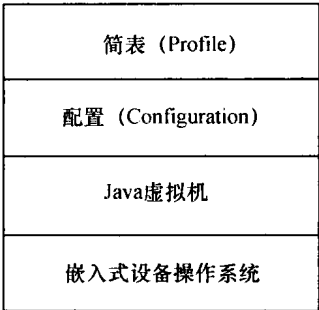


图 1-4 配置和简表层次图



目前 Java ME 领域里使用最广泛的是移动信息设备简表（MIDP），它主要是针对手机和其他双向移动通信设备而设计的，本书在后面章节中将会重点讲解移动信息设备简表。

1.2 Java ME 配置简介

1.2.1 CDC 概述

CDC 是应用在具有较大内存的设备中的，CDC 针对有限和间歇的无线连接设备，CDC 与 CVM 和 Foundation 简表共同定义了完整的 Java ME 环境。

CDC 应用的目标设备通常具有以下特征：

- 使用 32 位处理器。
- 2 MB 或更多可供 Java 平台使用的总内存。
- 设备要求的 Java 2 “蓝皮书”虚拟机的全部功能。
- 网络连通性，常常是无线的、不一致的连接并且带宽有限。
- 用户接口混乱，程度参差不齐，甚至有时根本就没有接口。

CDC 是整个 Java SE API 的一个较为完整的子集。它包含了 CLDC 中定义的所有的类和接口。但是 CDC 中的一些更值得注意的功能是 CLDC 中所没有的，例如：

- 支持浮点数（包括 java.lang.Float、java.lang.Double 和 java.lang.StrictMath 类）。
- 支持 java.lang.ClassLoader 类。

- 支持本地进程 (`java.lang.Process`)。
- 高级多线程支持 (包括支持线程组和更多线程)。
- 串行化的类 (`java.io.Serializable` 和 `java.io.Externalizable`)。
- 映象 API (包括 `java.lang.reflect` 软件包)。
- 文件系统支持。
- 支持 Java SE 类型网络 (`java.net`)。
- 对 Java SE Collections API 更完全的支持。
- 为 `javax.microedition.io` 包增加一个 `HttpConnection` 接口。这样可为 HTTP 连接提供必要的方法和常量。
- 支持 Java SE 的 `java.lang.ref`、`java.math`、`java.security`、`java.security.cert`、`java.text`、`java.util.jar` 和 `java.util.zip` 等包。

1.2.2 CLDC 概述

2000 年 5 月, Java Community Process (JCP) 公布了 CLDC 1.0 规范 (即 JSR30)。作为第一个面向小型嵌入式信息设备的 Java 应用开发规范, CLDC 是由包括 Nokia、Motorola 和 Siemens 在内的 18 家全球知名公司共同协商完成的。

CLDC 的目标是: 为小型的、资源受限的连接设备定义一个 Java 平台标准; 允许向上述设备动态地传递 Java 应用和内容; 使 Java 开发人员能够轻松地在这些设备上应用开发。

CLDC 是针对少于 512 KB 内存的嵌入式设备的配置。CLDC 采用的是 KVM 虚拟机, 它是高度优化的 JVM 方式, 它针对的是 16 位或 32 位的微处理器、160 KB~512 KB 内存为基础的低端、网络连接、电池驱动的设备。典型的内存要求是 128 KB 来存储 KVM 和相关的核心运行类库。

CLDC 应用的目标设备通常具有以下硬件特征:

- 提供 160 KB~512 KB 的内存。
- 低速处理器。
- 有限电力 (常常为电池)。
- 与网络的连接是有限的、也许是发射性的连接 (常常是无线连接)。
- 非常有限的用户界面, 小屏幕。

和硬件类似, CLDC 上运行的软件也是多种多样的。例如, 有些设备支持多进程操作系统或者支持文件系统; 而有些功能极其有限的设备并不需要文件系统。对于这些不确定性, CLDC 只定义了软件所必须的最小集合。CLDC 规范中要求操作系统不需要支持多进程或是分址空间寻址, 也不用考虑运行时的协调和延迟; 但是必须提供至少一个可控制的实体来运行虚拟机。

此外, 就网络化而言, CLDC 针对的是有限的和间歇性的无线连接设备, 所有在 Java ME CLDC 之上运行的应用软件都应能在其他任何高端的 Java 虚拟机上运行。当然这需要 CLDC 库的支持。CLDC 配置关注于 I/O、网络化、安全问题、国际化问题和核心 Java 库等相关的领域。

1.2.2.1 CLDC 的功能范围

CLDC 包含了一个基本的 Java ME 运行环境, 其中包括虚拟机和核心的 Java API 类库。作为专门针对于小型嵌入式信息设备的配置, CLDC 对 Java SE 类库进行了大量的简化, 其类库只保留了 Java 规范中定义的最核心的 3 个包, 即 `java.io`、`java.lang` 和 `java.util`, 并重新定义了一个新的包 `javax.microedition`。这里可以通过前缀来区别: `java.`前缀表示核心的 java 包, `javax.`前缀表示 Java ME 标准 java 扩展包。

CLDC 中的包和其所对应的功能如下所示。

- java.io: 标准的输入/输出功能, Java SE 中 java.io 包的子集。
- java.lang: 核心语言包, Java SE 的子集。
- java.util: 实用工具类。
- javax.microedition.io: 通用连接框架类及接口。

javax.microedition 包中其他的子包定义了 CLDC 中没有定义的功能, 如对应用程序生命周期的管理、用户界面 (UI)、事件处理模式、永久性存储和用户与应用程序的交互等。这些功能的定义是由移动信息设备简表 (MIDP) 来完成的。

1.2.2.2 CLDC 类库简介

为了确保与其他 Java 平台的兼容性, 绝大多数的 CLDC 类库是从 Java SE 和 Java EE 中继承的, 是 Java SE 和 Java EE 的子集。由于目标设备的特殊性, CLDC 类库在安全、输入/输出、用户界面、网络和存储管理等方面没有全部使用 Java SE 来实现; 其中的部分类库 CLDC 进行了重写, 如用户界面、网络连接等。

CLDC 的类库可以分为两大类: 一类是从 Java SE 标准类库中继承的; 另一类是为 CLDC 专门设计的 (这部分类其实也可以被映射到 Java SE 中)。

对于第一类 CLDC 类库, 包括了 Java SE 的 3 个核心的包 java.io、java.lang 和 java.util。但是 CLDC 中的这 3 个包和 Java SE 中的相应包相比, 也只是 Java SE 相应包的一个很小的子集。



例如 java.util 的类与接口由 Java SE 中的 53 个减少到 10 个。

对于后一类 CLDC 类库, 只有描述标准连接框架的 javax.microedition.io 包, 它和 MIDP 中定义的包一起被放置于 javax.microedition 包中。

1.2.2.3 CLDC 1.1 新特性

CLDC 共有 1.0 和 1.1 两个版本。CLDC 1.1 版 (JSR139) 相对于 CLDC 1.0 版本并没有本质上的变化。随着硬件水平的不断提高, CLDC 1.1 版在兼容性和可用性上有了一些改进, 并增加了一些 1.0 版本中没有的新特性:

- 增加对浮点数据的支持, 核心类库中增加 java.lang.Float 类和 java.lang.Double 类。
- Calendar、Date 和 TimeZone 类被重新设计, 与 Java SE 中的类更加类似。
- 对错误处理有了更加明确的定义, 并增加了 No、Class、Def、Found、Error 类。
- 对于 Thread 类, CLDC 1.1 允许为线程命名, 并通过 getName() 方法得知线程的名字, 同时增加 interrupt() 方法, 允许中断线程; 增加了新的构造方法。
- 对一些类库进行了小的修改, 一些方法被添加或是修正。
- 由于允许使用浮点运算, 所支持设备的内存被提高到 160 KB~192 KB。

1.3 MIDP 简介

移动信息设备简表 (MIDP, Mobile Information Device Profile) 针对的是移动信息设备或者体积较小的能力有限的手持设备 (MID), 这些设备的共性是具有用户接口, 即屏幕和输入机制。

MIDP 目标设备的典型例子就是手机和 PDA。

1.3.1 MIDP 设备

移动信息设备的硬件要求非常简单，即单色或者彩色显示屏幕，最小尺寸为 96 像素 × 54 像素；小型键盘或者触摸屏输入方式；双向无线网络连接能力；最小 128 KB 存储空间用于 MIDP 系统软件，8 KB 用于应用程序定义的存储空间，32 KB 用于 Java 运行时堆栈的存储空间。

除了硬件要求，MIDP 规范还定义了其实现的软件需求，即支持 MIDP 的设备应该能够满足以下几个条件：

- 能够管理软件并提供运行 KVM 虚拟机的线程。
- 提供读写永久存储空间，可以访问无线网络，能够显示输出。
- 能够接受并处理按钮或者触摸屏触发的事件。
- 通过应用程序管理器实现 MIDP 应用程序的生命周期的管理。

在 MIDP 规范中没有定义应用程序管理器（Application Management Software，AMS）是如何工作和编写的，仅仅是定义了其应该实现的功能，因此在不同的设备上应用程序管理器都是设备厂商自行研发的，其差别可能会比较大，应用程序管理器甚至可以用非 Java 语言来编写。

1.3.2 MIDP 体系结构

MIDP 中主要包含 7 个软件包，其中有 4 个核心 CLDC 软件包，另外加上 3 个特定于 MIDP 的软件包，其名称和功能如下：

- 核心包：java.lang。
- 工具包：java.util。
- I/O 包：java.io。
- 网络 I/O 包：javax.microedition.io。
- 用户界面包：javax.microedition.lcdui。
- MIDP 主类包：javax.microedition.midlet。
- 持久化存储包：javax.microedition.rms。

MIDP 中特定软件包的主要功能如下：

javax.microedition.lcdui 定义用来控制 UI（User Interface）的类。这个软件包既包含高级 UI 类，又包含低级 UI 类。

MIDP 包括一个低级的 UI API 集合和一个高级的 UI API 集合。低级的 API 集合允许开发者完全访问一个设备的显示屏，也允许访问原始键和指针事件。然而，使用低级 API 时，没有可用的用户界面控件，而且应用程序必须精确地绘制出按钮和其他所有的控件。

与之相反，高级 API 集合提供简单的用户界面控件但不能直接访问原始的输入事件或显示屏。由于显示屏的尺寸和 MIDP 设备输入方法的差异，控件的外观显示显得很抽象。MIDP 的实现确定了绘制控件的方法，也确定了如何管理用户输入。

javax.microedition.midlet 包含 MIDP 应用程序中的主类 MIDlet，为 MIDP 应用程序提供访问关于其运行所在环境信息的权限。

javax.microedition.rms 定义一些类的集合，这些类为 MIDlet 提供了永久存储并随后重新得到数据的机制。

1.3.3 MIDP 2.0 新特性

2000 年 9 月，SUN 公司发布了 MIDP 的第一个正式版本 MIDP 1.0。随着 Java ME 技术的不断发展和用户需求的不断提高，SUN 公司又于 2002 年 11 月发布了 MIDP 2.0 版本。它对设备的内存资源和处理能力的要求较 1.0 版本要高，但 MIDP 2.0 版本也为应用开发者提供了更方便、更丰富多彩的组件包，主要增加了游戏接口的实现、声音输出接口的实现、安全网络机制的实现。MIDP 2.0 版本的这些特性将使基于移动设备的 Java 应用具有更加广阔的前景，也必将使新一代的移动设备发生革命性的变化并领导时尚潮流。

MIDP 2.0 版本中软件包及功能描述如表 1-1 所示。

表 1-1 MIDP 2.0 版本软件包功能描述

属 性	描 述
javax.microedition.lcdui	提供一系列用户界面接口
javax.microedition.lcdui.game	专门用于游戏设计的接口
javax.microedition.rms	数据管理，用于保存数据记录
javax.microedition.midlet	应用管理接口
javax.microedition.io	基本网络连接接口
javax.microedition.media	媒体接口规范（JSR-135）的实现包
javax.microedition.media.control	媒体播放器的控制类
javax.microedition.pki	数字签名规范的实现接口，用于安全网络
java.io	基本输入输出接口
java.lang	基本数据类型接口
java.util	基本应用工具接口

根据与 MIDP 1.0 版本中软件包的对照，MIDP 2.0 版本的新增特性如下：

- 用户界面扩展，例如弹出窗口、可激活的字符串和图像等。
- 对屏幕布局的更好控制，包括可以对 Form 上的 Item 的大小进行控制。
- 可以自定义 Form Item，应用不再受限于 MIDP 1.0 中的几种基本的 Form，而可以自己使用 Java 实现新的类型。
- 支持简单的声音回放，MIDP 2.0 设备必须支持 MMAPI 的一个功能子集。当然厂商也可以支持完整的 MMAPI。
- 新引入的 Sprite API 简化了图形应用，特别适合游戏的开发。
- 支持 HTTPS、TCP/IP、UDP 以及串口通信。
- MIDlet 可以被设置成可由外部事件激活，该技术被称为“push architecture”。
- 所有的 MIDP 2.0 设备必须支持使用 OTA 获取 MIDlet 套件，实际上，大部分 MIDP 1.0 设备就支持 OTA 下载了，只不过在 MIDP 1.0 中，此项特性并不是强制必须的。

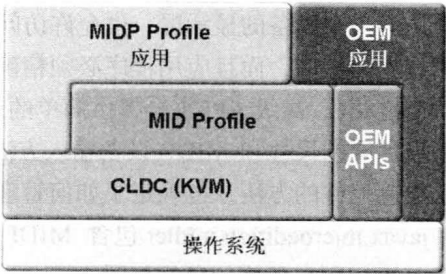


图 1-5 基于 CLDC 和 MIDP 的应用程序的体系结构

- 支持对 MIDlet 进行数字签名。此签名作为其内容的加密摘要被一起提供。此特性提供了一个有效的方案,使得验证 MIDlet 的发布者成为可能。设备的使用者可以根据这个信息来选择是否赋予 MIDlet 对运行环境的更多访问权限。

根据之前关于 CLDC 和 MIDP 的介绍,可以归纳出基于 CLDC 和 MIDP 的应用程序的体系结构如图 1-5 所示。

1.4 MIDlet 简介

移动信息设备小程序 (Mobile Information Devices let, MIDlet) 是可以在 KVM 上运行的 Java ME 应用程序。它是一个普通的 Java 类,但是在定义时必须继承 javax.microedition.midlet 包中的 MIDlet 类。

1.4.1 MIDlet 生命周期

与 Java SE 中的 Applet 或者 Java EE 中的 Servlet 类似, MIDlet 应用程序在移动设备上运行时也具有应用程序生命周期。所谓生命周期,是指程序从开始执行到退出的生命过程,例如,一般的 Java SE 应用程序的生命周期就是从 main () 方法开发,当 main () 方法结束而且没有线程运行时则程序结束。由于 Java ME 手机程序在执行过程中,需要为手机环境的系统事件预留出处理的接口,所以使 MIDlet 的生命周期显得比较复杂。

MIDlet 的生命周期共有 3 个状态:活动 (Active) 状态、暂停 (Paused) 状态和销毁 (Destroyed) 状态。

- 活动状态:程序处于正常执行状态。当一个 MIDlet 进入活动状态时,它将获得用于执行任务所需的所有资源。转移到活动状态之后,所需的线程应该被启动。

- 暂停状态:程序处于后台运行状态,当一个 MIDlet 进入暂停状态时,它应该释放所有所持有的资源并停止活动的线程。

- 销毁状态:程序进入退出前的处理,该状态结束则程序退出。当一个 MIDlet 进入销毁状态时,它应该释放所有资源,停止正在执行的线程,并保存持久化的数据。

MIDlet 中必须实现一些特定的生命周期方法,应用程序管理器通过调用 MIDlet 生命周期中的各个方法来管理 MIDlet 的行为。

MIDlet 定义如下生命周期方法来控制 MIDlet 的状态:

- 默认构造器:默认构造器用来执行基本的初始化操作,此时 MIDlet 还处在暂停状态,它并没有获得任何所需的资源。

- startApp () 方法: startApp () 方法用来获取 MIDlet 所需的资源,然后 MIDlet 将会处于活跃状态。

- pauseApp () 方法: pauseApp () 方法用来释放 MIDlet 所持有的全部资源。

- destroyApp () 方法: destroyApp () 方法用来保存 MIDlet 的状态并释放 MIDlet 所持有的全部资源。通常 MIDlet 应该在被关闭之前进入此状态。

整个 MIDlet 程序的执行流程如下:

1. 系统调用继承自 MIDlet 类的默认构造方法,创建对象。
2. 系统进入 MIDlet 生命周期的暂停状态,这个是 MIDlet 生命周期的第一个状态。