

EMBEDDED
SYSTEM

嵌入式技术与应用丛书

J2ME嵌入式 开发案例精解

杨军 秦冬 王莹 编著



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

<http://www.phei.com.cn>



嵌入式技术与应用丛书

J2ME 嵌入式开发案例精解

杨 军 秦 冬 王 莹 编著

电子工业出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书以技巧和案例的方式讲解了 J2ME 程序设计的方法,包括各种应用程序的方方面面,如多媒体程序、网络程序、商务应用程序等,介绍了各种可选包的使用方法。全书可以分成两大部分,第一部分侧重于编程技巧的讲解,分为图像处理、音频处理、视频处理、网络程序、短消息收发、蓝牙通信、XML 解析和其他特殊技巧几个方面,介绍了使用 J2ME 开发的多种技巧;第二部分侧重于实践案例,每章讲解一个方面的案例,涵盖桌面应用、商务应用、个性化程序、游戏和网络实用程序。本书内容全面,由浅入深,每个案例都力求简洁和清晰,详细地说明问题,提出解决问题的方法,分析程序的性能。

本书适合于具有一定 J2ME 编程基础并对 J2ME 技术感兴趣的人员阅读。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

图书在版编目(CIP)数据

J2ME 嵌入式开发案例精解/杨军,秦冬,王莹编著. —北京:电子工业出版社,2007.7

(嵌入式技术与应用丛书)

ISBN 978-7-121-04666-7

I. J… II. ①杨…②秦…③王… III. JAVA 语言—程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(2007)第 098556 号

责任编辑:高买花 特约编辑:陈宁辉

印 刷:北京民族印刷厂

装 订:北京鼎盛东极装订有限公司

出版发行:电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本:787×1092 1/16 印张:22 字数:508 千字

印 次:2007 年 7 月第 1 次印刷

定 价:36.00 元

凡所购买电子工业出版社图书有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系,联系及邮购电话:(010) 88254888。

质量投诉请发邮件至 zlt@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线:(010) 88258888。

前 言

对于 J2ME 嵌入式软件系统的开发，我们不仅仅需要掌握使用程序语言的技巧，更重要的是掌握一个 J2ME 嵌入式软件项目在系统分析、设计及测试阶段所需要的开发技巧。本书将充分利用 J2ME 在应用程序的开发和设计上灵活、高效的特点，将 J2ME 常用的知识点汇集成实例，通过这些实例向读者讲述 J2ME 的开发方法与设计技巧。

本书充分考虑了结构的层次性，进行了循序渐进的安排。本书以 J2ME 编程技巧和系统开发为基本的出发点，讲解 J2ME 使用技巧和系统开发方面的知识。全书共分为两个部分，第一部分为 J2ME 技巧部分，分别介绍 J2ME 在音频、视频、媒体、网络等方面的使用技巧。第二部分为 J2ME 的案例精解部分，讲解 J2ME 在一些典型领域的应用，每一个案例都由系统分析、设计、编码、测试几个部分组成，通过这些案例的学习，可以使读者迅速掌握 J2ME 开发方面的知识、技巧并能将它们运用到实际工作中。

本书适用于以下读者：

- 计划或正在从事 J2ME 嵌入式软件开发的程序人员；
- 对 J2ME 嵌入式软件开发有兴趣、想了解的朋友；
- 社会上 J2ME 嵌入式软件培训学校的学员；
- 大专院校相关专业的学生。

本书主要作者为杨军、秦冬、王莹，此外，以下人员也参与了本书的资料收集及写作工作：宇宏文、杨健、杨琪昌、王晓璇、张李、王沛、董华、刘新伟、高宇、王琪、王彪、陈杨、李悦、范雅娜、马旋、高春轶、王宝哲、范朝辉、高振兴等，他们对本书的完成付出了辛勤的汗水和心血，在此一并表示衷心的感谢。

由于时间仓促，加之编著水平有限，书中的缺点和不足之处在所难免，敬请读者批评指正。

编著者
2007 年 5 月

目 录

第一部分 J2ME 编程技巧案例

第 1 章 J2ME 技巧案例	(3)
1.1 图像处理技巧	(3)
1.1.1 设计说明	(3)
1.1.2 代码实现	(4)
1.1.3 疑难解析	(10)
1.1.4 举一反三	(10)
1.2 音频处理技巧	(16)
1.2.1 设计说明	(16)
1.2.2 代码实现	(16)
1.2.3 疑难解析	(19)
1.2.4 举一反三	(20)
1.3 视频处理技巧	(24)
1.3.1 设计说明	(24)
1.3.2 代码实现	(24)
1.3.3 疑难解析	(28)
1.3.4 举一反三	(28)
1.4 HTTP 网络处理技巧	(29)
1.4.1 设计说明	(29)
1.4.2 代码实现	(29)
1.4.3 疑难解析	(33)
1.4.4 举一反三	(33)
1.5 短消息收发技巧	(39)
1.5.1 设计说明	(39)
1.5.2 代码实现	(39)
1.5.3 疑难解析	(41)
1.5.4 举一反三	(42)
1.6 蓝牙通信技术技巧	(46)
1.6.1 设计说明	(46)
1.6.2 代码实现	(47)
1.6.3 疑难解析	(51)
1.6.4 举一反三	(51)

1.7 XML 解析	(52)
1.7.1 设计说明	(52)
1.7.2 代码实现	(54)
1.7.3 疑难解析	(61)
1.7.4 举一反三	(61)
1.8 其他特殊技巧	(82)
1.8.1 使用定位服务	(82)
1.8.2 使用 PushRegister	(84)
1.8.3 使用自定义 Item	(86)
1.8.4 查看系统内存信息	(88)
1.8.5 产生随机数	(88)
1.8.6 管理个人信息	(89)
1.9 本章小结	(95)

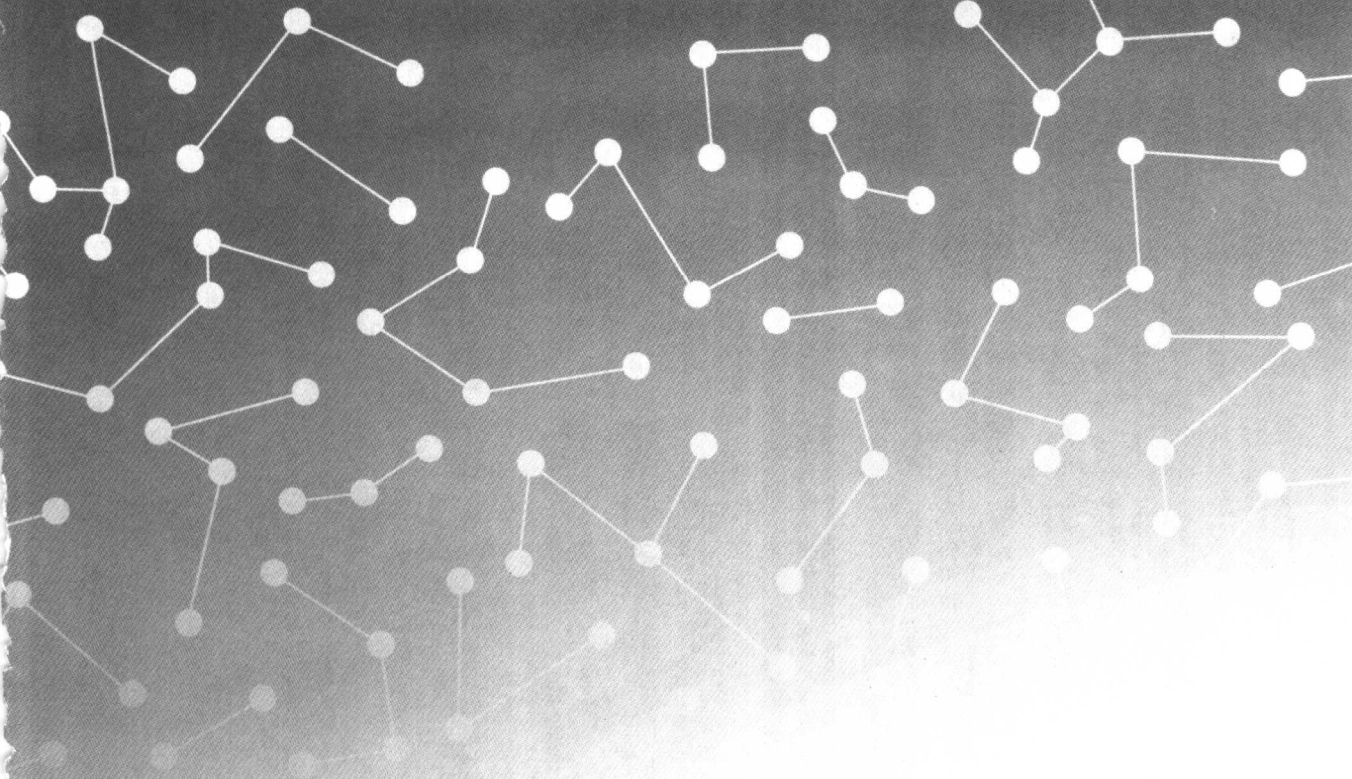
第二部分 J2ME 编程综合案例

第 2 章 电子词典	(99)
2.1 系统概述	(99)
2.1.1 技术背景	(99)
2.1.2 基本架构	(99)
2.2 结构设计	(100)
2.2.1 原理图	(100)
2.2.2 模块间的关系	(101)
2.3 模块说明	(101)
2.3.1 界面显示模块	(101)
2.3.2 词典模块	(113)
2.4 程序扩展	(118)
2.5 本章小结	(119)
第 3 章 个人证券信息管理	(120)
3.1 系统概述	(120)
3.1.1 技术背景	(120)
3.1.2 基本架构	(121)
3.2 结构设计	(121)
3.2.1 原理图	(121)
3.2.2 工作流程	(122)
3.3 模块说明	(123)
3.3.1 界面显示模块	(123)
3.3.2 数据管理模块	(144)

3.3.3	网络连接模块	(156)
3.4	程序扩展	(158)
3.5	本章小结	(159)
第 4 章	计算器系统	(160)
4.1	系统概述	(160)
4.1.1	技术背景	(160)
4.1.2	基本架构	(161)
4.2	结构设计	(161)
4.2.1	原理图	(161)
4.2.2	两种思路的比较	(162)
4.3	模块设计	(163)
4.3.1	功能模拟	(163)
4.3.2	界面模拟	(167)
4.4	程序扩展	(174)
4.5	本章小结	(175)
第 5 章	无线网络传输系统	(176)
5.1	系统概述	(176)
5.1.1	技术背景	(176)
5.1.2	基本架构	(177)
5.2	结构设计	(177)
5.2.1	原理图	(177)
5.2.2	工作流程	(179)
5.3	模块设计	(179)
5.3.1	主程序模块	(180)
5.3.2	HTTP 传输模块	(182)
5.3.3	UDP 传输模块	(193)
5.3.4	套接字传输模块	(199)
5.4	程序扩展	(202)
5.5	本章小结	(203)
第 6 章	无线 OBEX 文件传输	(204)
6.1	系统概述	(204)
6.1.1	技术背景	(204)
6.1.2	OBEX 对象模型	(205)
6.1.3	OBEX 操作	(206)
6.2	结构设计	(208)
6.2.1	原理图	(208)
6.2.2	工作流程	(209)
6.3	模块设计	(209)

6.3.1	主程序模块	(209)
6.3.2	客户端模块	(211)
6.3.3	服务器端模块	(221)
6.4	程序扩展	(230)
6.5	本章小结	(231)
第 7 章	资源管理器	(232)
7.1	系统概述	(232)
7.1.1	技术背景	(232)
7.1.2	文件操作模式	(233)
7.1.3	FileConnection 包的使用	(233)
7.2	结构设计	(236)
7.2.1	原理图	(236)
7.2.2	工作流程	(237)
7.3	模块说明	(237)
7.3.1	界面显示模块	(237)
7.3.2	文件连接管理模块	(248)
7.4	程序扩展	(252)
7.5	本章小结	(252)
第 8 章	博弈游戏	(253)
8.1	系统概述	(253)
8.1.1	技术背景	(253)
8.1.2	中国跳棋程序设计	(255)
8.2	结构设计	(258)
8.2.1	原理图	(258)
8.2.2	工作流程	(259)
8.3	模块说明	(259)
8.3.1	主程序模块	(259)
8.3.2	界面显示模块	(262)
8.3.3	可走步产生模块	(271)
8.3.4	局面评估模块	(276)
8.3.5	智能搜索模块	(279)
8.4	程序扩展	(284)
8.5	本章小结	(285)
第 9 章	利用 J2ME 蓝牙技术设计聊天程序	(286)
9.1	系统概述	(286)
9.1.1	蓝牙技术的背景	(286)
9.1.2	聊天程序的基本架构	(287)
9.2	结构设计	(287)

9.2.1	原理图	(288)
9.2.2	工作流程	(289)
9.3	模块设计	(289)
9.3.1	主程序模块	(289)
9.3.2	屏幕显示模块	(294)
9.3.3	蓝牙通信层模块	(299)
9.3.4	远程设备管理模块	(306)
9.3.5	发送消息线程模块	(308)
9.3.6	接收消息线程模块	(308)
9.4	程序扩展	(311)
9.5	本章小结	(311)
第 10 章	邮件收发系统	(312)
10.1	系统概述	(312)
10.1.1	技术背景	(312)
10.1.2	电子邮件程序设计	(313)
10.2	结构设计	(314)
10.2.1	原理图	(314)
10.2.2	工作流程	(315)
10.3	模块设计	(315)
10.3.1	发送方客户端模块	(315)
10.3.2	发送方邮件服务器模块	(323)
10.3.3	接收方客户端模块	(328)
10.3.4	接收方邮件服务器	(335)
10.4	程序扩展	(339)
10.5	本章小结	(339)
参考文献		(340)



第一部分

J2ME 编程技巧案例

第 1 章 J2ME 技巧案例



本章目标

在阅读完本章后您将了解:

- 图像处理技巧;
- 音频处理技巧;
- 视频处理技巧;
- HTTP 网络处理技巧;
- 短消息收发技巧;
- 蓝牙通信技术技巧;
- 其他特殊技巧。



本章简介

J2ME 技术是 Sun 公司针对小型的、资源有限的设备进行 Java 应用程序开发所设计的软件开发平台。本章将介绍 J2ME 程序设计中的各种技巧,并按照内容进行分类,在每一类技巧中,我们以完整实例为主线,辅以各种短小的代码,力求精要地讲解每种技巧。

1.1 图像处理技巧

通常情况下,我们所说的图像都是指静态图像,因此,本节将介绍静态图像和动画的实现方法,前者以统计学中的饼状图为例,后者以 GIF 动画为例进行说明。

1.1.1 设计说明

在统计学中,使用图形可以形象地表示数据内在的关联,包括趋势、比较和比例等。本节中将通过 `PieMIDlet` 类介绍一种绘制饼状图的方法,根据数据信息得到各组成部分所占的比例,进而绘制饼状图中的不同成分,以不同颜色组成的扇形分别代表数据的每一个组成部分。

饼状图是静态图像的例子, J2ME 平台上也支持动画的绘制。在本节中我们通过 `GIFMIDlet` 类演示 GIF 动画的实现过程。GIF 动画是由数张图片连续显示制造的一种视觉

效果，其原理与影片是一样的，关键是每张图片所停滞的时间决定着不同的动画显示速度。根据 GIF 动画的原理，我们可以将若干幅静态图像组合起来，按照固定的时间间隔显示图像序列。这个显示的过程也就是动画播放的过程。使用 MIDP 规范中的底层界面 API 可以显示静态图像，`paint` 方法可以更新屏幕显示，固定的时间间隔则利用一个定时器来计时（这个定时器一般使用另外一个线程实现），一旦定时器到时，则重新绘制屏幕，将图像序列中的下一幅绘制到屏幕上。为此，`GIFMIDlet` 类包含三个部分：构造静态图像序列、设置定时器延时并启动、绘制序列中的当前图像。

1.1.2 代码实现

本节演示 `PieMIDlet` 类和 `GIFMIDlet` 类的具体实现方法，说明静态饼状图和 GIF 动画的绘制方法，下面分别进行分析。

1. 静态饼状图

下面的程序演示静态饼状图的绘制。该程序由两个文件组成，其中 `PieMIDlet.java` 定义主程序类，`PieCanvas.java` 定义屏幕显示类。

(1) `PieMIDlet.java` 文件

`PieMIDlet` 类对屏幕界面进行初始化，添加三个命令按钮，供用户选择。若用户选择“确定”按钮，`PieCanvas` 实例就会被设置为屏幕显示的内容，“返回”或“退出”按钮则将结束程序的运行。下面的代码演示了 `PieCanvas` 类的定义。

饼状图绘制 (`PieMIDlet.java`)

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;

public class PieMIDlet extends MIDlet implements CommandListener {
    Display display;
    Command exitCommand;
    Command backCommand;
    Command okCommand;
    PieCanvas pie;
    TextBox textbox;
    Form form;

    public PieMIDlet() {
        display = Display.getDisplay(this);
        exitCommand = new Command("退出", Command.EXIT, 1);
        backCommand = new Command("返回", Command.BACK, 2);
        okCommand = new Command("确定", Command.OK, 3);
        if (pie == null) {
            pie = new PieCanvas();
            pie.addCommand(backCommand);
```

```
        pie.setCommandListener(this);
    }
}

public void startApp () {
}

public void destroyApp (boolean unconditional) {
}

public void pauseApp () {
}

public void commandAction(Command c, Displayable s) {
    if (c == exitCommand) {
        notifyDestroyed();
    } else if (s == okCommand) {
        display.setCurrent(pie);
    }
}
```

(2) PieCanvas.java 文件

该文件定义了 PieCanvas 类，它负责静态饼状图的实际绘制过程，绘制过程由 paint 方法完成，该方法中的最后部分调用 fillArc 方法，分别绘制各自的扇形区域，最后两个参数确定扇形的旋转角度。

此外，PieCanvas 类的成员变量 x, y 确定饼状图的坐标位置，font 确定文字的字体，w 确定饼状图的宽度，h 确定饼状图的高度，fh 确定字体高度。

setColor 方法允许用户设置饼状图的不同颜色，handleActions 方法响应用户对饼状图的移动操作。

代码如下：

饼状图绘制 (PieCanvas.java)

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;

class PieCanvas extends Canvas {
    int    x, y;
    Font    font;
    int    fh;
    int    w, h;
    int    titleHeight;
    int    pieSize;
    int    pad;

    PieCanvas() {
        w = getWidth();
```

```
h = getHeight();
font = Font.getFont(Font.FACE_SYSTEM,
    Font.STYLE_PLAIN, Font.SIZE_SMALL);
fh = font.getHeight();
pad = 2;
titleHeight = fh + pad * 2;
eventHeight = fh * 3;
pieSize = h - (titleHeight + pad) - (eventHeight + pad);
if (pieSize < 20)
    pieSize = 20;
if (pieSize > (w - pad) / 2)
    pieSize = (w - pad) / 2;
}

protected void keyPressed(int key) {
    handleActions(key);
    repaint();
}

void handleActions(int keyCode) {
    int action = getGameAction(keyCode);
    switch (action) {
        case LEFT:
            x -= 1;
            break;
        case RIGHT:
            x += 1;
            break;
        case UP:
            y -= 1;
            break;
        case DOWN:
            y += 1;
            break;
    }
}

protected void paint(Graphics g) {
    g.setFont(font);
    g.setGrayScale(255);
    g.fillRect(0, 0, w, h);
    x = (x < 0) ? w - 1 : x;
```

```
y = (y < 0) ? h - 1 : y;  
x = x % w;  
y = y % h;  
int swidth = pad * 2 + font.stringWidth("饼状图");  
int title_x = (w - swidth)/2;  
g.setGrayScale(128);  
g.fillRoundRect(title_x, 0, swidth, fh, 5, 5);  
g.setGrayScale(0);  
g.drawRoundRect(title_x, 0, swidth, fh, 5, 5);  
//开始绘制饼状图  
g.setColor(255, 0, 0);  
g.fillArc(0, 0, pieSize, pieSize, 45, 270);  
g.setColor(0, 255, 0);  
g.fillArc(0, 0, pieSize, pieSize, 0, 45);  
g.setColor(0, 0, 255);  
g.fillArc(0, 0, pieSize, pieSize, 0, -45);  
g.setColor(0);  
g.drawArc(0, 0, pieSize, pieSize, 0, 360);  
}  
}
```

静态饼状图的绘制结果如图 1-1 所示。



图 1-1 静态饼状图的绘制结果

2. GIF 动画

GIF 动画实例是通过两个文件（GIFMIDlet.java 和 MyCanvas.java）完成的，其中 GIFMIDlet.java 定义主程序的 MIDlet 类，MyCanvas.java 定义屏幕显示类。

(1) GIFMIDlet.java 文件

该文件定义 GIFMIDlet 类，它负责程序的主流程控制，因为 GIF 动画实例的初始化过程并不复杂，因此 MIDlet 类中并没有定义类的构造方法。

利用静态图像实现 GIF 动画 (GIFMIDlet.java)

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;

public class GIFMIDlet extends MIDlet {
    Display display;
    public void destroyApp(boolean unconditional){
    }

    public void pauseApp() {
    }

    public void startApp() {
        display = Display.getDisplay(this);
        final MyCanvas c = new MyCanvas();
        display.setCurrent(c);
    }
}
```

(2) MyCanvas.java 文件

在该文件中建立一个底层界面类 MyCanvas，它负责屏幕的显示，该类实现 Runnable 用于定时器的计时，初始化时从外部文件中装载了三幅静态图像，然后启动线程“t”，该线程的运行主体完成了图像切换和延时，增加当前图像序列中的索引值，使当前图像过渡到下一幅图像；调用 repaint 方法，更新屏幕显示；最后调用 sleep 方法使线程延时，即控制两幅图像显示的间隔时间。

利用静态图像实现 GIF 动画 (MyCanvas.java)

```
import javax.microedition.midlet.*;
import javax.microedition.lcdui.*;

public class MyCanvas extends Canvas implements Runnable {
    Image img[]=new Image[3];
    int index;
    public MyCanvas() {
        try {
            img[0]=Image.createImage("/img1.png");
            img[1]=Image.createImage("/img2.png");
            img[2]=Image.createImage("/img3.png");
        }
    }
}
```