

JAVA 事件处理指南

Grant Palmer

著

沈莹 谢乐健 管知时

译

清华大学出版社

(京) 新登字 158 号

北京市版权局著作权合同登记号: 01-2001-0101

内 容 简 介

Java 语言中高效率的事件处理功能是创建功能强大的图形用户界面软件的关键。本书是第一本专门讨论 Java 事件处理的专著, 涵盖了从基础知识到中高级技巧的所有内容。本书作者 Grant Palmer 是一位著名的 Java 软件开发专家, 他不但深入剖析了 Java 事件模型, 而且讲解了如何充分利用事件处理来提供更为强大的控制能力和功能。

本书内容包括了本地和分布式事件的差别、相似性以及实用技术; 所有 Java 事件的详细参考, 包括事件类以及生存周期方法等; 高级技术包括: 事件监听器管理器类、用户定义的事件类以及事件监听器等; 许多简单实用的程序代码示例。

本书适合于各类 Java 程序员阅读, 尤其适用于准备开发图形用户界面应用程序的中高级 Java 程序员, 是他们进一步提高编程技能的实用指南。

Simplified Chinese edition copyright © 2002 by Pearson Education NORTH ASIA LIMITED and Tsinghua University Press.

Java Event Handling: first publication by Grant Palmer, Copyright © 2002.

All Rights Reserved.

Published by arrangement with Pearson Education, Inc., publishing as PH PTR.

This edition is authorized for sale only in the People's Republic of China (excluding the Special Administrative Region of Hong Kong and Macau).

本书中文简体字版由清华大学出版社和美国培生教育出版集团合作出版。未经出版者书面许可, 不得以任何方式复制或抄袭本书内容。

版权所有, 翻印必究。

本书封面贴有 Pearson Education 出版集团激光防伪标签, 无标签者不得销售。

书 名: JAVA 事件处理指南

作 者: Grant Palmer 著 沈莹、谢乐健、管知时 译

出 版 者: 清华大学出版社 (北京清华大学学研大厦, 邮编 100084)

<http://www.tup.tsinghua.edu.cn>

责任编辑: 夏兆彦

印 刷 者: 清华大学印刷厂

发 行 者: 新华书店总店北京发行所

开 本: 787×1092 1/16 印张: 28.5 字数: 729 千字

版 次: 2002 年 6 月第 1 版 2002 年 6 月第 1 次印刷

书 号: ISBN 7-900643-10-9

印 数: 0001~5000

定 价: 58.00 元

引言

欢迎来到有趣的 Java 事件天地。什么是事件？事件可谓是 Java 的信使，从事件源传递到最终目标，同时携带着有关已发生事务的信息。事件使图形用户界面(Graphical User Interface, GUI)能够进行其应该做的工作。对于实现 Java Bean 中约束和限制属性这一概念，事件是其核心部分。在一些分布式应用程序中，如使用 Jini 或者 JavaSpaces 技术的应用程序中，事件也很重要。

Java 事件处理是一项重要的课题，但令人遗憾的是，大多数 Java 参考书籍没能很好地介绍它的内容。在 Java 的广阔天地里可写的内容实在是太多了，以至于一般的参考书倾向于只用区区几十页来描述事件和事件处理。有关材料也是散布在书中各处。然而，本书却是一本完整的 Java 事件处理参考书，是市面上惟一本完全以 Java 事件处理为中心内容的书。这是一本有关这项课题的权威性的著作，内容不仅包括关于事件类和接口的完整技术参考，还详尽地介绍了 Java 事件模型本身。

0.1 本书的内容结构

本书分为三部分。第 I 部分讲述了 Java 事件处理的基础内容，以提出事件的定义和作用为开始。还介绍了 Java 事件模型的历史以及对此模型的描述。此外还描述了 Java 事件的生命周期，讲述了自事件生成那一刻到它抵达最终目标期间所发生的事情。最后，还论及其他一些主题，如事件队列和事件分派线程，而这些都是 Java 事件处理的基本内容。

本书第 II 部分是对 Java 2 标准版平台(Java Platform 2 Standard Edition, J2SE)内的事件类和接口的完整的技术参考，包括从 ActionEvent 到 WindowListener 以及它们间的所有内容。为每一个类或接口提供的信息包括语法、所属的包、当被引入到 Java API 时类或接口的层次结构，以及类或接口定义的构造函数、字段和方法。每一部分都是完全独立的，为和所有指定的类或接口有关的信息提供“一站式服务”(one stop shopping)，包括对从其他类或接口继承来的字段和方法的描述。关于类或接口的作用和怎样使用类或接口的问题，第 II 部分都会做出解答。

本书的第 I 部分和第 II 部分会使您深刻地理解 Java 事件处理。其中的基本原理应用于大多数编程工作中。在第 III 部分中，我们将探讨一些更高级的主题。您也许不会经常使用它们，但这些主题是非常有用的(使用起来也很有趣)，而且可以用来定制指定代码中事件处理的各个方面。另外，第 III 部分还详细讲述了事件监听器管理器类。这些类通常由系统使用，但是程序员可访问这些类来创建新的事件类和/或事件生成对象。同时，还描述了创建用户定义的事件类和接口的过程。

第 III 部分也演示了怎样运用所学的所有内容和怎样开发一些“真实世界中的” GUI 应用程序。关于怎样实现代码的事件处理部分，我们将不得不做出程序设计决策。第 III 部分的最后讨论了分布式事件这一概念。到此为止，本书所涉及的都是本地事件。这些事件是由使用单一 Java 虚拟机(Java Virtual Machine, JVM)，运行于单一机器上的应用程序生成的。分布式的或基于 Web 的应用程序可能运行于多个平台上并使用多个 JVM。Java 提供了一个框架，可以用于负责远程



资源产生的事件的生成、传输和处理工作。

本书还包括易于使用的附录，列出了由 J2SE API 提供的事件生成对象。列表详细地列出了每个对象能够生成的事件和怎样生成这些事件。列表不仅包括所有 GUI 组件和容器，还详细列举了各种模型、文档和其他事件生成对象的事件生成能力。例如，如果您想知道 JTextField 对象是否可以生成 TextEvent 事件，附录会告知您答案。

0.2 代码示例的注解

与其它任何编程语言一样，学习 Java 的最好方法是通过示例。而本书就提供了很多示例。这些示例是完整的独立的应用程序。书中没有代码片断示例，因为这种示例可能使人产生误解并且不能显示全貌。大多数示例(尤其是在第 I 和第 II 部分中)是简单易懂的程序，这是特意安排的。这些程序很容易理解，意在演示 Java 事件处理一个方面的一两个问题，而不使用许多额外的代码使内容散乱。

这些示例是以相似的方式进行编译和运行的。例如，为 SimpleExample.java 创建字节代码，可以键入下列程序：

```
javac SimpleExample.java
```

运行此示例，可键入

```
java SimpleExample
```

我们鼓励您试用这些示例。大多数示例都很简短，如果愿意的话，您可以自己输入它们，或者可以通过从本书附带的光盘中复制它们来使用。

我们也鼓励您灵活运用这些示例。改变某些内容，验证会产生什么效果。正如不把脚弄湿就不可能学会游泳一样，如果不编写或修改代码，就不可能真正地理解 Java 事件处理或者基于此的一般意义上的 Java 语言。

除了第 III 部分的远程事件示例之外，本书中所有示例都使用 J2SE API。您应该有 Java Development Kit (JDK) 版本 1.3，因为许多示例使用版本 1.3 中引入的类和方法。可以从 Sun 的网站 <http://java.sun.com> 下载 Java Software Development Kit，它包含大多数示例需要的所有内容。远程事件处理示例还要求在您的系统中安装 Jini。更详细的内容，请参阅本书第 III 部分。

0.3 本书不涉及的内容

既然我们已经介绍了本书的意图，现在就来简略地讲一下它所不包括的内容。本书不是一本关于 Java 的入门指导，因此要求读者具备一些(尽管不是很多)使用 Java 的经验。

您应该熟悉面向对象的基本原理，例如类、继承、构造函数等。您应该知道不同的访问类型，以及它们的意义和使用方法。您还应该知道“this”关键字的意思，了解内部类和匿名内部类。除此之外，如果对核心的 Java 包中的类和接口至少要有基本上的理解，也是很有帮助的。

本书也不是关于 AWT 或 Swing GUI 组件的参考书。这些组件用于书中的每个示例，但在使用中并没有对它们做出解释。大多数示例以简单易懂的方式运用 GUI 组件。如果您有过一些使用 GUI 组件类的经验，就会非常明白示例的意思和用它的原因了。如对任何事情有疑问，请参阅一本好的 Swing 或 AWT 参考书，以获得启示。

关于作者

自 1985 年起，Grant Palmer 就一直是 NASA(美国国家航空和宇宙航行局)Ames 研究中心的科学编程人员。在 20 世纪 90 年代中期，他明显的意识到自己的编程技能已经严重过时了。他在大多数工作中都在使用 FORTRAN，然而身边的世界却越来越转向面向对象的编程语言，如 C++和一种名为 Java 的新语言。在 1996 年冬天 Grant 首次向 Java 世界进发，从此便沉迷于其中。他最钟爱 Java 的多功能性。使用一种编程语言，他就可以把 GUI 前端处理机写成科学程序并将其整个转换成基于 Web 的应用程序。

Grant 与妻子 Lisa 及两个儿子 Jackson 和 Zachary 一起住在亚利桑那州的 Chandler。在照料两个儿子之余，他喜欢游泳，偶尔也打打高尔夫，不过打得很糟。

目 录

第 I 部分 基础知识

第 1 章	Java 事件介绍	3
1.1	什么是事件	3
1.2	本地事件和分布式事件	4
1.3	Java 事件处理的演变	4
1.4	Java 事件模型	5
1.4.1	Java 1.0 事件模型	5
1.4.2	Java 1.1 事件模型	7
第 2 章	Java 事件的生命周期	11
2.1	事件生命周期方法	12
2.1.1	Component 类的基本事件生命周期方法	12
2.1.2	事件处理方法	12
2.1.3	激发事件的方法	13
2.1.4	摒弃的 Component 类的方法	14
2.1.5	定义在 Component 子类内的事件生命周期方法	14
第 3 章	事件类	15
3.1	事件和支持类层次结构	15
3.2	低层次与高层次事件类型	16
3.3	对 J2SE 中的事件类的描述	16
3.3.1	EventObject 类	16
3.3.2	AWTEvent 类	16
3.3.3	定义在 Java.awt.event 包内的低层次事件类	16
3.3.4	在 java.awt.event 包内定义的高层次事件类	17
3.3.5	定义在 java.awt.event 包内的特殊事件类	18
3.3.6	定义在 javax.swing.event 包内的低层次事件类	18
3.3.7	定义在 javax.swing.event 包内的高层次事件类	18
3.3.8	定义在 javax.swing.event 包内的高层次事件接口	19
3.3.9	定义在 javax.swing.event 包内的事件支持类	19
3.3.10	定义在 java.beans 包内的事件和支持类	20
3.3.11	定义在 javax.swing.text.html 包内的事件类	20
3.3.12	定义在 javax.swing.undo 包内的事件支持类	20



3.3.13 定义在 J2SE 内的其他事件类	21
3.4 用户生成的事件	21
3.5 用户定义的事件	24
第 4 章 事件监听器	25
4.1 对接口的简要回顾	25
4.2 J2SE 中的事件监听器接口	26
4.2.1 EventListener 接口	26
4.2.2 包含在 java.awt.event 包内的监听器接口	26
4.2.3 包含在 javax.swing.event 包内的监听器接口	27
4.2.4 包含在 java.beans 包内的监听器接口	28
4.2.5 定义在 J2SE 中的其他监听器接口	28
4.3 事件监听器对象	29
4.4 创建事件监听器	29
4.4.1 方法 1: 让 GUI 组件充当事件监听器	29
4.4.2 方法 2: 把事件监听器作为单独类来实现	31
4.4.3 方法 3: 把事件监听器作为内部类来实现	32
4.4.4 方法 4: 把事件监听器作为匿名内部类来实现	33
4.5 监听器适配器类	35
4.6 包含在 java.awt.event 包内的监听器适配器类	37
4.7 包含在 javax.swing.event 包内的监听器适配器类	38
4.8 用户定义的事件监听器接口	38
4.9 把事件源连接到事件监听器	38
4.10 从事件监听器断开事件源	39
4.11 getListeners() 方法	42
4.12 事件监听器管理器类	42
4.12.1 AWTEventMulticaster 类	43
4.12.2 EventListenerList 类	43
第 5 章 关于事件处理的细节内容	44
5.1 决定事件源	44
5.2 事件消耗	48
5.3 事件消耗方法	48
5.4 事件队列	50
5.5 EventQueue 类	50
5.6 事件分派线程	53

第 II 部分 JAVA 事件详解

第 6 章 事件类和接口	57
6.1 ActionEvent 类	57
6.2 ActiveEvent 接口	62
6.3 AdjustmentEvent 类	68
6.4 AncestorEvent 类	72
6.5 AWTEvent 类	76
6.6 CaretEvent 类	78
6.7 ChangeEvent 类	81
6.8 ComponentEvent 类	84
6.9 ContainerEvent 类	88
6.10 DocumentEvent 接口	92
6.11 DocumentEvent.ElementChange 接口	95
6.12 DocumentEvent.EventType 类	96
6.13 EventObject 类	97
6.14 FocusEvent 类	98
6.15 HierarchyEvent 类	102
6.16 HTMLFrameHyperlinkEvent 类	107
6.17 HyperlinkEvent 类	112
6.18 HyperlinkEvent.EventType 类	117
6.19 InputEvent 类	118
6.20 InputMethodEvent 类	120
6.21 InternalFrameEvent 类	123
6.22 InvocationEvent 类	127
6.23 ItemEvent 类	131
6.24 KeyEvent 类	135
6.25 ListDataEvent 类	141
6.26 ListSelectionEvent 类	145
6.27 MenuDragMouseEvent 类	149
6.28 MenuEvent 类	153
6.29 MenuKeyEvent 类	156
6.30 MouseEvent 类	162
6.31 PaintEvent 类	167
6.32 PopupMenuEvent 类	167
6.33 PropertyChangeEvent 类	171
6.34 TableColumnModelEvent 类	177
6.35 TableModelEvent 类	180



6.36	TextEvent 类	184
6.37	TreeExpansionEvent 类	187
6.38	TreeModelEvent 类	192
6.39	TreeSelectionEvent 类	196
6.40	UndoableEditEvent 类	201
6.41	WindowEvent 类	205
6.42	J2SE 中的其他事件类	208
第 7 章	事件支持类	210
7.1	PropertyChangeSupport 类	210
7.2	SwingPropertyChangeSupport 类	212
7.3	UndoableEditSupport 类	214
7.4	VetoableChangeSupport 类	222
第 8 章	事件监听器接口	230
8.1	ActionListener 接口	230
8.2	AdjustmentListener 接口	231
8.3	AncestorListener 接口	233
8.4	AWTEventListener 接口	234
8.5	CaretListener 接口	236
8.6	CellEditorListener 接口	237
8.7	ChangeListener 接口	241
8.8	ComponentListener 接口	242
8.9	ContainerListener 接口	246
8.10	DocumentListener 接口	247
8.11	EventListener 接口	248
8.12	FocusListener 接口	248
8.13	HierarchyBoundsListener 接口	250
8.14	HierarchyListener 接口	253
8.15	HyperlinkListener 接口	256
8.16	InputMethodListener 接口	257
8.17	InternalFrameListener 接口	259
8.18	ItemListener 接口	261
8.19	KeyListener 接口	262
8.20	ListDataListener 接口	265
8.21	ListSelectionListener 接口	266
8.22	MenuDragMouseListener 接口	267
8.23	MenuKeyListener 接口	268
8.24	MenuListener 接口	270

8.25	MouseListener 接口	271
8.26	MouseListener 接口	274
8.27	MouseMotionListener 接口	278
8.28	PopupMenuListener 接口	280
8.29	PropertyChangeListener 接口	281
8.30	TableColumnModelListener 接口	283
8.31	TableModelListener 接口	284
8.32	TextListener 接口	285
8.33	TreeExpansionListener 接口	286
8.34	TreeModelListener 接口	289
8.35	TreeSelectionListener 接口	291
8.36	TreeWillExpandListener 接口	292
8.37	UndoableEditListener 接口	293
8.38	VetoableChangeListener 接口	294
8.39	WindowListener 接口	296
8.40	J2SE 的其他监听器接口	299
第 9 章	监听器适配器类	301
9.1	ComponentAdapter 类	301
9.2	ContainerAdapter 类	302
9.3	FocusAdapter 类	305
9.4	HierarchyBoundsAdapter 类	308
9.5	InternalFrameAdapter 类	309
9.6	KeyAdapter 类	313
9.7	MouseAdapter 类	314
9.8	MouseInputAdapter 类	315
9.9	MouseMotionAdapter 类	318
9.10	WindowAdapter 类	320
第 10 章	事件生命周期方法	322
10.1	AbstractAction 类方法	322
10.2	AbstractButton 类方法	322
10.3	AbstractCellEditor 类方法	323
10.4	AbstractListModel 类方法	323
10.5	Button 类方法	323
10.6	Checkbox 类方法	324
10.7	CheckboxMenuItem 类方法	324
10.8	Choice 类方法	324
10.9	Component 类方法	325



10.10	Container 类方法	328
10.11	DefaultBoundedRangeModel 类方法	329
10.12	DefaultButtonModel 类方法	329
10.13	DefaultListCellRenderer 类方法	329
10.14	DefaultListSelectionModel 类方法	330
10.15	DefaultSingleSelectionModel 类方法	330
10.16	JApplet 类方法	330
10.17	JComboBox 类方法	330
10.18	JComponent 类方法	331
10.19	JDialog 类方法	332
10.20	JEditorPane 类方法	332
10.21	JFrame 类方法	332
10.22	JInternalFrame 类方法	333
10.23	JList 类方法	333
10.24	JMenu 类方法	333
10.25	JMenuBar 类方法	333
10.26	JMenuItem 类方法	334
10.27	JPopupMenu 类方法	335
10.28	JProgressBar 类方法	335
10.29	JScrollBar 类方法	336
10.30	JSlider 的类方法	336
10.31	JTabbedPane 类方法	336
10.32	JTextArea 类方法	336
10.33	TextField 类方法	336
10.34	JTree 类方法	337
10.35	JViewport 类方法	337
10.36	List 类方法	337
10.37	MenuComponent 类方法	338
10.38	MenuItem 类方法	338
10.39	PropertyChangeSupport 类方法	339
10.40	ScrollBar 类方法	339
10.41	SwingPropertyChangeSupport 类方法	339
10.42	TextComponent 类方法	339
10.43	TextField 类方法	340
10.44	Timer 类方法	340
10.45	VetoableChangeSupport 类方法	340
10.46	Window 类方法	341

第Ⅲ部分 高级主题

第 11 章	事件监听器管理器类	345
11.1	AWTEventMulticaster 类	345
11.2	EventListenerList 类	353
第 12 章	用户定义事件类和事件监听器	360
12.1	创建用户定义的事件监听器	360
12.2	创建用户定义的事件类	361
12.3	定义支持用户定义事件的组件	363
12.4	综合运用	365
第 13 章	综合应用	370
13.1	临界点热度率程序	370
13.2	Java 文档编辑器	383
第 14 章	分布式事件	399
14.1	分布式事件模型	399
14.2	远程事件类	400
14.3	远程事件监听器接口	407
14.4	远程事件监听器支持类	408
附录	Java GUI 组件及其生成的事件	420

第 I 部分 基础知识

本部分描述了有关 Java 事件的一些基本方面，是写给那些不熟悉事件和事件处理概念的初学者。尽管如此，第 I 部分中的内容也会使有经验的编程人员从中受益。有关用户生成的事件、实现事件监听器的各种方式和事件队列的内容，都是本部分将要介绍的内容。

这部分以及整本书都没有要求读者懂得 Java 事件或事件处理，但确实要求读者对 Java 编程的基础有一定程度的熟悉。读者应当了解基本的类结构、继承、构造函数、导入包、声名变量、重载类方法等等。其他经常用于示例中的概念为：关键字“this”、内部类、类中接口方法的实现。

本书不打算成为一本 AWT 或 Swing 参考书。其中提供的示例通常是简单独立的应用程序，这些应用程序说明了 Java 事件类型、监听器以及各种支持和生命周期方法的基础知识。同时也很好的说明了 Java 事件处理进程，然而每个示例中 GUI 的各个方面以及组成界面的组件和容器经常会出现，但没有对它们进行解释。如果对构造示例的方法或者怎样使用 GUI 组件不熟悉或者好奇，请参考一本好的 AWT 或 Swing 参考书，希望可以获得深入的解释。

由于编写和运行 Java 代码确实是学习语言 and 了解程序怎样运作的最好方法，因此我们鼓励读者输入或下载示例，并且在自己的机器上试验其效果。欢迎读者通过改变某些代码来试验示例，看看会产生什么效果。关于 Java 事件处理的说明是通常可以有几种方法完成相同的事情。通过运用这些示例，您可能会发现您的喜好是什么以及什么著作对您和您的程序最有帮助。

最后一点有关示例部分的说明是，您阅读此书时将会注意到，有的示例中使用 AWT GUI 组件而有的用 Swing GUI 组件。因为 Swing 组件比相应的 AWT 组件功能更强，而且对于创建有效的应用程序有更大的潜力，因此大多数的示例用的都是 Swing 组件。在 `javax.swing.event` 包中的许多事件只能由 Swing 组件生成。但有时使用 AWT 类是合适的。例如，在第 6 章“事件类和接口”的“MouseEvent 类”部分，示例中运用 Canvas 对象来实现简单的画板。因为 Canvas 是 AWT 组件，而且混合和匹配 AWT 和 Swing 组件是一种糟糕的编程方式，所以整个示例只使用了 AWT 组件。与此相似，稍后在第一部分中讨论的事件消耗只与由 AWT 组件生成的事件有关，因此那部分的示例也使用了 AWT 组件。

介绍完毕基本情况后，现在让我们来介绍 Java 事件。

第1章 Java事件介绍

本章中，我们将讲述 Java 事件处理中最基本的内容。首先会对什么是 Java 事件及事件的作用加以描述，接着将讨论事件的两种基本类型：本地事件和分布式事件。在介绍两种基本的本地事件模型时，会顺便介绍一下 Java 事件处理的简要历史。本章最后以一个简单的示例演示 Java 事件处理的关键元素。

1.1 什么是事件

任何一本关于 Java 事件处理的书都应首先解决“什么是事件”这一基本问题。对事件的正式定义为：事件是代表另一对象抽象状态下变化的对象。事件本身就是抽象。生成事件的过程是无法从对象外部直接观察的。

然而这一定义到底意味着什么呢？让我们来看一个有着不同按钮、菜单、组合框等的图形用户界面(GUI)的示例。当和 GUI 相互作用时，您会期待某事的发生。比如说，如果选中了 Quit 菜单，这一行为的预期结果是应用程序将结束。当 Quit 菜单被选中时，必须存在能通知程序的机制。在 Java 中，这个机制就是事件。当 Quit 菜单被选中时，事件生成且输送到另外一段名为事件监听器的代码。事件监听器包括处理事件的代码。在 Quit 菜单项这种情况下，监听器可能包括命令 `System.exit(0)`。

几乎所有围绕 GUI 进行的活动都会生成事件。在空的界面上移动鼠标光标将会生成鼠标事件。调整窗口的大小将会生成容器事件。当用户操作特定的程序时，将会生成一连串真正的事件。幸运的是，Java 事件模型可让您挑选出将要处理的事件。

事件并不仅限于 GUI 组件。当与组件关联的属性发生变化时，Java Bean 组件框架为生成事件做准备。此属性可能是原始变量的值，可能是 Bean 关联的对象中的变化，或是任何类实例变量或从超类继承来的字段的实例变量。生成事件的能力甚至超过了 GUI 组件和 Java Bean 的功能。Java 应用程序接口(API)提供支持和工具类，使得事件生成能力扩展到几乎任何用户定义的种类。

考虑事件的一种方式是把它们看成 Java 的信使。它们从事件源、GUI 组件或其他与之相互作用的对象那里将信息传输到任何对此感兴趣的事件监听器。事件传输的信息总是包括对事件源的引用，可能也包含指出事件确切特性的附加信息。

事件对象和其他包含在 Java API 内的对象有相似特点。它有构造函数、字段和类方法。事件对象可作为参数传递到方法。与其他对象不同的一点是，事件通常由系统自动生成，而且自动传递到适当的事件处理程序。对于事件过程的大部分具体细节，编程人员是不知道的。然而，创建用户定义事件来适应特殊应用程序的特定需要却是可能的。



1.2 本地事件和分布式事件

本地事件是使用单一 Java 虚拟机(JVM)且运行于单一平台上的应用程序生成的事件。本地事件应用于 Java API 中的大多数 GUI 组件。它也是 Java Bean 架构的一部分,这意味着非 GUI 组件也可以生成事件。当一定的交互作用发生时,GUI 组件通常会自动生成事件。Java API 提供所需要的事件监听器接口和工具,以将事件监听程序注册到组件。

分布式事件就是由远程目标生成的事件。生成事件的应用程序也许是运行于不同的平台或使用不同的 JVM,或者这两种情况兼而有之。分布式事件发生在使用 Jini 或 JavaSpaces 技术的基于网络或 Web 的应用程序中。本书的重点是介绍本地事件,但是在第 III 部分中提供了分布式事件模型和它的类及接口的简要介绍。

1.3 Java 事件处理的演变

这里,有必要对 Java 编程语言的发展做一个简要的概括,我们主要关注实现和处理事件的方式。

Java 1.0 (1995)。Java 于 1995 年出现在公众面前。它的发展历经数年,一开始是用于消费者电子设备服务的平台独立的语言,后来是作为创建 Internet 应用程序的方式。它借用了 C 和 C++的元素,但是性能却远远超出这些语言。一旦开发者们开始去解决怎样使用 Java 来创建 Internet 应用程序这一问题,他们认识到了 Java 事件模型的必要性。Java 1.0.2 事件模型提供了基本的事件处理功能。这是一种包容(Containment)模型。所有事件类型都封装在单一的类即 Event 类里面。所有事件对象都由单一的方法 `handleEvent()` 来处理,这些都定义在 Component 类中。为此,只有 Component 类的子类才能充当事件处理程序。事件处理传递到组件层次结构。如果目标组件不能完全处理事件,事件被传递到目标组件的容器。

Java 1.1 (1997)。Java 的首次公开发行人是编程界的一次革命,但不久,Java 1.0.2 的缺陷和特殊问题就变得明显起来。Java 研发小组开始致力于开发 Java 1.1,修正 Java 的某些不足之处且增加了一些重要的附加功能。Java 1.1 代表了 Java 语言的第二次革命。有了 Java 1.1,为方法命名的约定在“get-set-is”命名法的帮助下开始变得标准化了。此外还增加了一些重要的新的功能,如“远程方法调用”(Remote Method Invocation, RMI),“Java 本地接口”(Java Native Interface, JNI),“Java 数据库连接”(Java Database Connectivity, JDBC),以及 Java Bean 的概念。最重要的是,本书的目的所在,也就是 Java 事件模型,事件在 Java 语言中怎样得到处理的基本框架都被完全重写了,并且从 Java 1.0.2 模型迁移到委托(Delegation)事件模型,在后者中,事件源生成事件,然后把事件处理“委托”给另一段代码。

J2SE v1.2(1999)。“J2SE v1.2”是 Java 2 平台标准版 1.2(Java 2 Platform Standard Edition version 1.2)的简写。与其说从 Java 1.1 到 J2SE v1.2 这一变化是革命性的,不如说是进化性的。其中一个大的变化就是加入了 Swing 包。它们是功能更强大,更加可定制的 GUI 组件和与它们相关连的支持类。Java 1.1 事件模型仍然是标准事件处理模型,而且或多或少地维持了原样。引入了 `javax.swing.event` 包,它包含新的事件类和事件监听器接口,为 Swing GUI 组件提供所

要求的附加事件处理能力。

J2SE v1.3 (2000)。J2SE 1.3 版是 1.2 版的改进。重点放在了提高 Java 语言的性能。事件包中加入了一些附加事件类和接口，但是整个事件模型几乎维持了原样。在 1.3 版中引入了 **Rebot** 类，它能模拟鼠标和键盘事件，并能用于自动化测试、自运行演示、及其他要求鼠标和键盘控制的应用程序。

1.4 Java 事件模型

Java 事件模型真是一个绝好的名称，可以用来描述 Java 生成和处理事件的框架。Java 事件模型经历了从 Java 1.0.2 到 Java 1.1 的革命性的变化。此变化发生于 1997 年。我们当然希望到本书出版时，开发者们已经不再用 Java 1.0.2 事件模型进行新的开发了。有可能的是，还会有一些遗留下来的代码使用 1.0.2 事件模型。为此，为了介绍 Java 事件处理的整个历史，这两种事件模型都在以下章节中得以描述。

1.4.1 Java 1.0 事件模型

正如其名称所隐含的一样，Java 1.0 事件模型提供了处理到 Java 1.0.2 事件的方法学。在 1997 年，随着 Java 1.1 的发行而出现的 Java 1.1 事件模型取代了这一事件模型。随着最新的 Java 发行版即 Java 2 Platform Standard Edition version 1.3 的出现，大多数的 Java 1.0 事件模型逐渐被开发人员放弃了。正因如此，不应当再把 Java 1.0 事件模型用于新的代码了。在这里给出对 Java 1.0 事件模型描述只是为了对 Java 事件处理方法的发展提供某些历史环境。

在 Java 1.0 事件模型中，每一种事件类型都封装在单一的类即 `java.awt` 包所含有的 `Event` 类里面。`Event` 对象包括如下有关信息：它所代表的事件类型，事件生成的时间，事件发生的地点，事件生成时所按的键。系统生成的 `Event` 对象将有如下事件 ID：

```
Public static int ACTION_EVENT
Public static int KEY_ACTION
Public static int KEY_ACTION_RELEASE
Public static int KEY_PRESS
Public static int RELEASE
Public static int LIST_DESELECT
Public static int LIST_SELECT
Public static int MOUSE_DOWN
Public static int MOUSE_DRAG
Public static int MOUSE_ENTER
Public static int MOUSE_EXIT
Public static int MOUSE_MOVE
Public static int MOUSE_UP
Public static int SCROLL_ABSOLUTE
```