J2EE

开发全程实录

杨中科◎编著

- 剖析企业级系统架构设计理念
- 讲解J2EE在实战开发中的应用
- 来自开发一线作者的经验结晶

清华大学出版社

J2EE 开发全程实录

杨中科　编著

清华大学出版社
北　京

内 容 简 介

J2EE 是目前企业级软件开发的首选平台。本书从架构的角度讲解了一个完整的 J2EE 系统的搭建。内容包括：正则表达式、JSP、Swing、XML 等技术在实际中的应用；Spring、Hibernate、Struts 等开源框架的实战性应用；MDA、敏捷开发等理念在实际开发中的应用；如何搭建一个高度可扩展的系统。本书观点新颖，实例丰富，对企业级系统开发中涉及到的问题进行了深入分析，并以作者在开发实践中使用 J2EE 的实际经验为基础，生动地展示了企业级系统搭建的全过程。本书可以作为有一定 Java 基础的开发人员的参考书，也可以作为大专院校学生学习实际项目开发和毕业设计的指导书，还可以作为软件开发培训班的项目实战课程教材。

图书在版编目(CIP)数据

J2EE 开发全程实录/杨中科编著. —北京：清华大学出版社，2007.7
ISBN 978-7-302-15560-7

Ⅰ.J…　Ⅱ. 杨…　Ⅲ. JAVA 语言—程序设计　Ⅳ. TP312

中国版本图书馆 CIP 数据核字(2007)第 096234 号

责任编辑：彭　欣　宋延清
封面设计：阳光智慧 + 荀博
版式设计：杨玉兰
责任印制：王秀菊
出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　　编**：100084
c-service@tup.tsinghua.edu.cn
社 总 机：010-62770175　**邮购热线**：010-62786544
投稿咨询：010-62772015　**客户服务**：010-62776969
印 刷 者：清华大学印刷厂
装 订 者：三河市溧源装订厂
经　　销：全国新华书店
开　　本：185×260　**印　张**：35　**字　数**：799 千字
附光盘 1 张
版　　次：2007 年 7 月第 1 版　　**印　　次**：2007 年 7 月第1 次印刷
印　　数：1～5000
定　　价：65.00 元

序　言

JDK 中很多类的用法我都烂熟于胸了；我已经能够使用 Struts+Hibernate 做出一个像样的论坛，公司很多人都称我是 Hibernate 高手；我做过很多上千万的大项目；我有多年的编程经验，我写的代码很多人看了都叫好；我曾经用过 Delphi 三年，写过很多小程序，什么远程监控呀、API 劫持呀、木马呀，Windows 的 API 里边藏着不少好东西呀，Delphi 的控件也真是很丰富；我还研究了 C#，用 C#的 WebForm 做东西真是方便呀；对了，我目前正在研究很火的 AJAX 技术！可是……我是软件工程师吗？

1. 重剑无锋、大巧不工

很多开发人员做了一段时间开发以后经常琢磨着怎么写出精彩、巧妙的程序来，所以在程序中使用了大量的技巧，并引以为自豪，还被同事夸奖："真是高手呀，人家独辟蹊径用这种方法解决的问题，咱都看不懂，惭愧！"

可是这样的技巧真的很好吗？

我曾经接手过一段代码，这段代码据说是一位前辈高人留下的，这段代码经常出现 Bug，多少人接手过，每次试图修改 Bug 时就会引来更多的 Bug，被人称为一块硬石头。我读过了这段代码以后倒吸了一口凉气，这确实是高人写的代码，里边嵌套了 5 层循环，数据在界面和数据库之间加载来保存去很多次，并且用了巧妙的方式拦截了一个框架 API，实现了用很复杂的代码才能实现的功能，我读了一上午，并且用了两天时间去改 Bug，可谁知仍然是越改 Bug 越多。一气之下我把原来的代码都删掉了，然后用最普通但是比较笨的实现方式重新实现，从此以后这个功能很少出现 Bug，接手的人也再没有抱怨过代码难懂。

这件事让我想起我上学时候的一件事。大四的时候我在一个小软件公司兼职，当时做的是一个呼叫中心系统。我用了很多控件的高级特性进行开发，开发速度之快让项目经理惊讶不已，一个劲地夸我是高手，所以当功能做得差不多的时候他就放我去北京求职了。晚上 12 点，我正在北京的一个旅馆中整理第二天去招聘会所需要的资料的时候，项目经理的电话打了过来："兄弟，这边你做的程序有一个间歇性 Bug，现在程序跑不起来了。我让另外一个兄弟帮着改一下你的程序，他说看了半天也不知道你是怎么实现的。"我当时心里听了特别自豪，于是乎在电话中很骄傲地给改我程序的那个兄弟讲我是怎么实现的。后来这个程序又出现了一些问题，都是只有我才能搞定，以至于当我毕业要离开那个城市的时候，项目经理苦着脸对我说："兄弟，你走了以后谁来维护你的程序呀，当初真应该让你用简单一点的技术做呀。"当时听了他这句话我心里仍然是美滋滋的，可现在想起来却好难过。

在软件开发领域中，技巧的优点在于能独辟蹊径地解决一些问题，缺点是技巧并不为大众所熟知。若在程序中使用太多的技巧，可能会造成别人难以理解。一个技巧造就的一个局部的优点对整个系统而言是微不足道的，而一个别人看不懂、改不了的 Bug 则可能是

致命的。在团队协作开发的情况下，开发时应该强调的一个重要方面是程序的易读性，在能够保证软件的性能指标且能够满足用户需求的情况下，必须让其他程序员容易读懂你的程序。

编程时不可滥用技巧，应该用自然的方式编程，简洁是一种美。不会解决问题的人尽管是“菜鸟”，但是这种人破坏力很小；而把简单的问题用复杂的方式解决的人是“半瓶子醋”，这种人破坏力极强，就像当年的我一样！只有能够把复杂问题用简洁而又直接的方式解决的人，才有望成为高手；所以现在我们更愿意看到写得朴实无华，没有什么闪亮词汇的代码。

2. 框架与工具箱

经常听到有人说“我用某某框架做了一个东西”、“我开发了一个实现某某功能的框架”，那么到底什么是框架呢？

用《设计模式》一书中的定义来说就是：“框架(Framework)是构成一类特定软件可复用设计的一组相互协作的类。……框架规定了你的应用程序的体系结构。它定义了整体结构，类和对象的分割，各部分的主要责任，类和对象怎么协作，以及控制流程。”框架实现了对具体实现细节的反向控制(IOC)，实现者无需考虑框架层已经实现好的设计，只要按照框架的要求开发就可以了，然后把开发好的东西放到框架中就可以运行。

以 Java Web 开发而言，任何人都知道 MVC 的架构方式比传统的 Model1 方式更容易管理和维护，可是在实际开发中很多人又禁不住 Model 这种简便开发方式的诱惑。那么如何强迫自己使用 MVC 模式呢？当然是使用 MVC 的开发框架了，比如 Struts。采用 Struts 后必须写一个 JSP 作为 View，必须从 Action 继承来实现 Control，必须从 ActionForm 继承来实现 Model，框架约束住了开发人员那充满幻想的大脑，让我们以最佳的设计策略进行开发。

尽管框架中经常包含具体可用的子类或者工具包，但是使用框架的好处是可复用设计而非复用实现，使用框架时，我们无需考虑一些设计策略问题，因为我们已经无形中使用了框架设计好了的“最佳实践”。

与框架相对应的是工具箱(Toolkit)。工具箱是预定义在类库中的类，它是一组相关的、可复用的、提供了通用功能类的集合。我们经常使用的 ArrayList、FileOutputStream、CGLib、Dom4j 等都是工具箱。这些工具箱不强迫我们采用某个特定的设计，它们只是提供了功能上的帮助，所以说工具箱的作用是实现复用(或者称代码复用)。

在 Java 中有一种特殊的类叫做工具类(Utils)，比如 java.lang 包中的 Math、Commons-BeanUtils 中的 BeanUtils。这些类并没有封装任何状态在里边，也不允许调用者实例化它们，它们只是暴露了一些静态方法供其他类调用。这种类在其他支持函数库的语言里(比如 C++、Delphi)被称为“伪类”，因为这些类没有封装任何东西，没有自己的状态，只是一堆函数的集合而已，原本是应该被坚决杜绝的东西，但是由于 Java 不支持函数库，所以才允许它们的存在。很多人都批评说这些类是面向过程的东西，不是真正的面向对象。但是不得不承认的是，Java 这样的面向过程与面向对象的混合产品正是工业级开发所需要的，面向过程与面向对象并不冲突，否则那些真正的面向对象语言为什么一直还生活在学院派的温室

里面呢？

3. 再次框架

“再次框架”是我想使用的一个词汇，意思是在现有的框架基础上为了实现更多应用层次的设计复用而进行的框架设计。

很多人认为使用 Struts、Spring 这样的框架开发就是基于框架开发了，就是最好的设计了，岂不知这些框架只是解决了大部分通用的问题，很多具体实现上的问题还需要进行框架设计，否则做出来的东西仍然是难以理解、难以复用、难以扩展的。很多人基于某些著名框架写出来的论坛、网站的源代码里实际上充斥着大量重复的代码、糟糕的设计，这些东西确实应该被好好地“再次框架”了。

4. 企业框架

很多企业都建立了自己的一套或自用或开放的框架，比如 SAP 的 Netweaver、用友的 UAP、金蝶的 BOS、浪潮的 Loushang、上海普元的 EOS 等。开发这些框架是需要大量的资金和人力资源的投入的，但是带来的如下好处也是非常明显的。

(1) 模块复用

在企业信息系统中很多模块是通用的，比如权限管理、组织架构管理，企业框架把这些模块抽取出来，这样具体业务系统的开发者就可以专心于具体业务功能的实现。

(2) 提高产品开发效率

企业框架通常都提供了很多通用的工具箱或者辅助开发工具，业务系统的开发者使用这些工具箱或者辅助开发工具可以用最少的工作量在最短的时间内完成功能开发。试想如果做第 1 个功能和做第 1000 个功能耗时耗力一样多的话，那要这个框架有什么用呢？

(3) 保证业务系统使用了最佳实践

企业框架常常采用了非常合理的设计，如何取得远程接口、在哪个地方进行数据校验、如何储存数据等一系列问题都已经被设计好了，开发者只要按照框架的要求进行设计，就可以保证开发出来的产品是设计合理的。

(4) 对知识管理很重要

企业框架集中了公司所有软件项目的共同点，集中了对于公司最重要的知识的精华，随着框架的应用，框架本身也会随之升级优化。一个新加入企业的员工只要理解并掌握了这个框架，就可以很好地融入到团队中来；而离职的人员也已经把自己的知识留在了这个框架中。

(5) 提高产品的一致性

此处的一致性包含两个方面，一个是产品功能的一致性，另一个是产品实现的一致性。功能的一致性主要指产品的界面、操作方式等的一致性，这保证了用户可以很容易地学习和使用系统的所有模块；实现的一致性主要就是规范产品的实现方式。

开发人员是聪明且富于想象力的，不同的开发人员写出来的程序具有个体差异性。这里举一个例子。

有一个界面的功能是：用户可以往这个界面中输入人员的姓名、年龄、性别、地址等

信息，然后单击“保存”按钮保存到数据库中。

如果没有一个统一的框架，那么开发人员就可以随意发挥了：有人会在单击“保存”按钮的时候去逐个读取控件的值，然后通过 JDBC 执行一个 Insert 语句把数据保存到数据库中；有人会使用开源的数据绑定框架把控件与数据库的字段绑定起来，然后让数据绑定框架处理数据的保存；有人会先为人员建立一个 hibernate 配置文件，然后逐个读取控件的值填充到值对象中并调用 session.save()方法把数据保存到数据库。

开发人员的这种随意发挥是项目非常大的一个风险，如何保证离职人员的代码能迅速地被接手人读懂是非常重要的一个问题，如果这么一个小小的功能就有这么多种实现方式的话，较大规模功能聚合中将面临的问题就更是可想而知了。

而使用特定的开发框架以后呢？对于上述界面功能而言，系统规定只要覆盖父类的 initDataBind 方法，并在方法里注册界面控件与数据库字段的绑定关系就可以了，其余的如何连接数据库、如何保存都由框架处理了。相信这样做的话接手的人员几乎不用看代码就能知道程序是如何实现的。

采用企业框架带来的一个主要变化就是开发人员可随意发挥的余地小了，必须在框架的约束下进行开发，无法在开发过程中体现自己的“高超本领”。从提高管理效率角度来说，软件企业应当欢迎这种变化；而另一方面，企业中具有挑战新技术激情的优秀开发人员尚可针对企业框架不断地实施改进和完善工作，为企业的技术路线注入新的活力。

前　言

现在大部分软件开发书籍都是讲解某个技术如何使用，很少有讲实战的，即使有实战案例的讲解，也是讲解网上购物、聊天室之类已经被人写烂了的系统的开发，最可怕的是书中的实现代码惨不忍睹，使得读者很容易被误导，至于如何进行合理的架构设计就更是无从谈起；少数从国外引进的高端技术书籍又大谈特谈各种在天上飞来飞去的理论，“看的时候心潮澎湃，看完之后一脸茫然”，读者不知道如何将这些理论应用到实际的开发过程当中。本书就尝试着打破这种局面，把一个真实的案例系统搭建从头讲起，不仅包含具体的实现技术，也包含一些架构方面的设计思想。

这是一本以 Java 开发语言为载体来讲解企业级信息系统开发的书，其中涉及到了 Hibernate、Struts、Spring、JSP、Swing、JDBC 等很多技术，而且案例系统在搭建过程中也较合理地使用了面向对象理念进行系统设计，但是书中不可能全面讲解这些技术的细节知识，读者可以根据需要参考与这些技术相关的资料。

在本书的序言中介绍开发框架等的概念；第 1、2、3、4 章介绍正则表达式、AOP、自定义 JSP 标记等基础知识；第 5 章给出案例系统的需求文档；第 6 章基于 Spring 技术搭建案例系统的 Remoting 部分；第 7 章构建一个基于 MDA 理念的元数据引擎；第 8 章对案例系统中用到的枚举异常类、工具类等进行介绍；第 9、10、11、12 章基于 Spring、Hibernate 等技术搭建事务、DTO 生成器、权限控制、日志记录、多数据库支持等基础模块；第 13、14 章开发登录服务、Swing 客户端基础模块以及数据选择器等自定义 Swing 控件；第 15 章实现列表界面、编辑界面和编辑界面的基类；第 16 章搭建 Web 客户端的登录界面、主菜单等基础模块，并开发 JSP 用的数据选择器等自定义标记；第 17 章则以前面章节搭建出的框架为基础实现第 5 章中的需求文档所要求的功能。

在此，我要感谢为这本书的诞生给予我帮助的所有人。首先要感谢父母对我的养育之恩，他们在我辞职写书的过程中对我无微不至的帮助更是让我永远不能忘记；其次要感谢冯仁飞、刘培德、杨勇、戴敬、张洌生等同事对我的帮助和指导；此外还要感谢 CowNew 开源团队的朋友们(特别是 KingChou 的执着精神很值得我学习)；最后我要感谢清华大学出版社的彭欣编辑，她给我的帮助使得我们的合作非常圆满，使得本书能够顺利地完成创作和出版。

相对于业界很多高手来说，我的水平是很有限的，无论在实战方面还是在理论知识方面都还有不少差距，希望读者不吝指教，以便再版时改进，您可以给我发送邮件：about521@163.com，与本书相关的后续资料将会发布到 CowNew 开源团队的网站(http://www.cownew.com)中。

杨中科

目　录

第 1 章　正则表达式

计算机专业毕业或者使用过 UNIX、Perl 等产品的读者一定对正则表达式有一定的印象，即使没有接触过正则表达式也不要被它的外貌所吓倒。虽然做到精通正则表达式比较难，但是能够掌握它的基本应用却是非常容易的，一旦把正则表达式应用于实际问题的解决，就可以非常明显地提高工作效率。

正则表达式最早是由数学家 Stephen Kleene 在对自然语言的递增研究成果的基础上提出来的，具有完整语法的正则表达式使用在字符的格式匹配方面。正则表达式是一种描述文字模式的方法，包括重复出现、不同选择方式以及为某些字符(比如数字、字母)而提供的缩写形式，我们经常会遇到甚至用到它，例如在 Windows 的搜索界面中输入“*.txt”就可以搜索到所有的文本文件，在 DOS 提示符下输入“dir *.exe”就可以显示出当前路径下所有的可执行文件，在数据库中检索所有名称中含有“Tom”的记录时可以执行如下 SQL 语句：“select * from T_Table where FName like '%Tom%'”，这些“*.txt”、“dir *.exe”、“%Tom%”就是与正则表达式有关的一种模式，这些模式将会与一些文本相匹配。

1.1　为什么要用正则表达式

您一定做过字符串解析的工作吧，那么让我们再来回顾一下其复杂性。

【例 1.1】“192.168.10.5[port=8080]”这个字符串表示 IP 地址为 192.168.10.5 的服务器的 8080 端口，请用程序解析此字符串，然后打印出“IP 地址为***的服务器的***端口是打开的”。

用普通方法进行 IP 地址字符串解析的代码如下：

```
String text = "192.168.10.5[port=8080]";
int leftBraceIndex = text.indexOf('[');
String ipAddr = text.substring(0,leftBraceIndex);
int equalsIndex = text.indexOf('=');
String port = text.substring(equalsIndex+1,text.length()-1);
System.out.println("IP 地址为"+ipAddr+"的服务器的"+port+"端口是打开的");
```

好在字符串格式并不复杂，这么简单的任务还是能轻松搞定的。

【例 1.2】“192.168.10.5[port=21,type=FTP]”这个字符串表示 IP 地址为 192.168.10.5 的服务器的 21 端口提供的是 FTP 服务，其中如果“,type=FTP”部分被省略，则默认为 HTTP 服务。请用程序解析此字符串，然后打印出“IP 地址为***的服务器的***端口提供的服务为***”。

用普通方法进行 IP 地址字符串解析的代码如下：

```
public static void parseAddr2(String text)
{
    int leftBraceIndex = text.indexOf('[');
    String ipAddr = text.substring(0, leftBraceIndex);
    String subStr = text.substring(leftBraceIndex + 1, text.length() - 1);
    String[] strings = subStr.split(",");
    String portString = strings[0];
    String port = portString.split("=")[1];
    String type = "HTTP";
    if (strings.length == 2)
    {
        String typeString = strings[1];
        type = typeString.split("=")[1];
    }
    String msg = MessageFormat.format("IP 地址为{0}的服务器的{1}端口提供的服务
                为{2}",new Object[] { ipAddr, port, type });
    System.out.println(msg);
}
```

运行如下的测试代码：

```
parseAddr2("192.168.10.5[port=21,type=FTP]");
parseAddr2("192.168.10.5[port=80]");
```

运行结果：

```
IP 地址为 192.168.10.5 的服务器的 21 端口提供的服务为 FTP
IP 地址为 192.168.10.5 的服务器的 80 端口提供的服务为 HTTP
```

【例 1.3】判断一个字符串是否是 Email 地址。

用普通方法验证 Email 正确性的代码如下：

```
public static boolean validateEmail(String text)
{
    int atIndex = text.indexOf('@');
    int dotLastIndex = text.lastIndexOf('.');

    //必须含有@和.
    if (atIndex < 0 || dotLastIndex < 0)
    {
        return false;
    }
    int textLen = text.length();

    //不能以@或者.开始或者结束
    if (atIndex == 0 || atIndex == textLen || dotLastIndex == 0
            || dotLastIndex == textLen)
```

```
        {
            return false;
        }

        //@要在最后一个.之前
        if (atIndex > dotLastIndex)
        {
            return false;
        }
        return true;
    }
```

运行如下的测试代码：

```
System.out.println(validateEmail("cownew@cownew.com"));
System.out.println(validateEmail("bcdes"));
System.out.println(validateEmail("@cownew.com"));
```

运行结果：

```
true
false
false
```

【例 1.4】从一个文本中提取 E-mail 地址，比如从下面的文本中提取 E-mail 地址：

“CowNew 开源团队的网站是 http://www.cownew.com，下面是几个开发人员的 email：kingchou@cownew.com、about521@163.com，其他相关邮件可以发送到：test@test.com。”

用普通方法完成这个任务实在是费事！

可以看到，如果没有合适的理论和工具的支持，字符串的解析是一个非常痛苦的过程，像这里的几个这么简单的例子如果都要通过大量的 if...else...语句进行解析，那编译器解析程序源码的过程不就会像遇到比天书还要难懂的代码了吗？

做计算机基础理论研究的科学家们为了解决自然语言的识别问题，提出了正则表达式的概念，从而大大简化了文本的识别工作，并且为自动机等理论提供了坚实的基础。现在正则表达式已经从计算机理论的神坛走了下来，我们能在很多工具中找到正则表达式的身影，比如 UNIX 中的 vi 编辑器、Perl 脚本语言等。此外，现在流行的开发语言比如 C#、VB、Java 等也都提供了对正则表达式的直接支持，甚至在像 JavaScript 这种脚本语言中也能发现对正则表达式的支持。正则表达式已经超出了某种语言或某个系统的局限，成为被人们广为使用的工具，我们完全可以用它来解决实际开发中碰到的一些具体问题。

1.2 正则表达式入门

正则表达式是由普通字符(例如大小写字母)以及特殊字符(称为元字符)组成的文字模式。该模式描述在查找文字主体时待匹配的一个或多个字符串。正则表达式作为模板，将某种字符模式与所搜索的字符串进行匹配。

在最简单的情况下，一个正则表达式看上去就是一个普通的查找串。例如，正则表达式“java”中没有包含任何元字符，它可以匹配“java”和“javascript”等字符串，但是不能匹配“Java”。

1.2.1 正则表达式中元字符的用法

要想学会正则表达式，理解元字符是一个必须攻克的难关，这里先给出常用元字符的用法，如表 1.1 所示。即使不能完全看懂也没有关系，我们会在后面通过更多的例子来帮助读者理解。

表 1.1 元字符的用法

元 字 符	说 明
.	匹配任何单个字符。例如正则表达式“b.g”能匹配如下字符串：“big”、“bug”、“b g”，但是不匹配“buug”
$	匹配行结束符。例如正则表达式“EJB$” 能够匹配字符串“I like EJB”的末尾，但是不能匹配字符串“J2EE Without EJBs！”
^	匹配一行的开始。例如正则表达式“^Spring”能够匹配字符串“Spring is a J2EE framework”的开始，但是不能匹配“I use Spring in my project”
*	匹配 0 至多个在它之前的字符。例如正则表达式“zo*”能匹配 “z” 以及 “zoo”；正则表达式“.*”意味着能够匹配任意字符串
\	转义符，用来将元字符当作普通的字符来进行匹配。例如正则表达式\$被用来匹配美元符号，而不是行尾；正则表达式\.用来匹配点字符，而不是任何字符的通配符
[]	匹配括号中的任何一个字符。例如正则表达式“b[aui]g”匹配 bug、big 和 bug，但是不匹配 beg。可以在括号中使用连字符“-”来指定字符的区间来简化表示，例如正则表达式[0-9]可以匹配任何数字字符，这样正则表达式“a[0-9]c”就可以匹配“a0c”、“a1c”、“a2c”等字符串；还可以制定多个区间，例如“[A-Za-z]”可以匹配任何大小写字母。还有一个相配合使用的元字符“^”，用在这里并不像前边的那个“^”一样表示匹配行开始，而是表示“排除”，要想匹配除了指定区间之外的字符，就可以在左边的括号和第一个字符之间使用^字符，例如“[^163A-Z]”将能够匹配除了 1、6、3 和所有大写字母之外的任何字符
()	将 () 之间括起来的表达式定义为“组”(group)，并且将匹配这个表达式的字符保存到一个临时区域，这个元字符在字符串提取的时候非常有用
|	将两个匹配条件进行逻辑“或”运算。“z|food”能匹配“z”或“food”。“(z|f)ood”则能匹配“zood”或“food”
+	匹配前面的子表达式一次或多次。例如正则表达式 9+匹配 9、99、999 等
?	匹配前面的子表达式零次或一次。例如，“do(es)?”可以匹配“do”或“does”中的“do”。此元字符还有另外一个用途，就是表示非贪婪模式匹配
{n}	匹配确定的 n 次。例如，“e{2}”不能匹配“bed”中的“e”，但是能匹配“seed”中的两个“e”

续表

元 字 符	说 明
{n,}	至少匹配 n 次。例如，“e{2,}”不能匹配“bed”中的“e”，但能匹配 “seeeeeeeed”中的所有“e”
{n,m}	最少匹配 n 次且最多匹配 m 次。“e{1,3}”将匹配“seeeeeeeed”中的前三个“e”

1.2.2 Java 中的正则表达式 API

在 JDK 1.3 及之前的 JDK 版本中并没有包含正则表达式的类，如果要在 Java 中使用正则表达式必须使用第三方提供的正则表达式库，最有名的就是 Jakarta-ORO，该库以前叫做 OROMatcher，是 Daniel Savarese 赠送给 Jakarta Project 的一个开源包。

使用的时候首先要创建一个实现了 PatternCompiler 接口的实例变量以创建一个“模式编译器”，Jakarta-ORO 中实现了这个接口的类就是 Perl5Compiler，这个类做到了与 Perl5 的正则表达式完全兼容。

```
PatternCompiler compiler=new Perl5Compiler();
```

接着调用“模式编译器”的 compile 方法编译正则表达式：

```
Pattern pattern= compiler.compile("b[iue]g ");
```

接着创建一个实现了 PatternMatcher 接口的实例变量以创建一个“模式匹配器”，Jakarta-ORO 中实现了这个接口的类就是 Perl5Matcher。

```
PatternMatcher matcher=new Perl5Matcher();
```

然后就可以调用 PatternMatcher 接口中的 contains、matches 等方法来进行模式匹配了。

Jakarta-ORO 的使用是非常简便的，而且效率非常高，支持的正则表达式语法也是非常全的，唯一的缺点就是它不是 JDK 中的标准包。但从 JDK 1.4 开始提供了支持正则表达式的 API，它们位于 java.util.regex 包中，由于已经有了标准 API，所以本书将会用 java.util.regex 进行正则表达式的相关操作。

java.util.regex 中定义了一些表达式的简写，可以使得表达式显得更加简洁和清晰：

- \t——制表符，等同于\u0009。
- \n——换行符，等同于\u000A。
- \d——代表一个数字，等同于[0-9]。
- \D——代表非数字，等同于[^0-9]。
- \s——代表换行符、Tab 制表符等空白字符。
- \S——代表非空白字符。
- \w——字母字符，等同于[a-zA-Z_0-9]。
- \W——非字母字符，等同于[^\w]。