Struts 与 Velocity 的集成

用五个步骤轻松替代 **JSP**

级别： 中级

George Franciscus (george.franciscus@nexcel.ca)，首席顾问，Nexcel

2005 年 10 月 17 日

Struts Recipes 的合著者 George Franciscus 带您一步步地把 Velocity 模板引擎集成进 Struts 应用程序。结果是一个快速、灵活的 JSP 替代物，同时带有希望从 Struts 得到的所有方便。

Java™ 服务器页面（JSP）技术是如此普及，以至于人们忘记了在创建 Web 页面时还有其他选择。但是最近，有些开发人员已经转向模板引擎，以获得在 JSP 中得不到的灵活性。虽然用 JSP 和模板引擎都可以把数据嵌入 HTML，但是每种技术都有自己的处理方式。Velocity 模板是一个特别流行的 JSP 替代品。Velocity 提供了平缓的学习曲线和巨大的易用性。开发人员喜欢它简洁的语法，而且性能分析也证明它的性能超出 JSP。Velocity 也非常容易集成进 Struts 应用程序。

在这篇文章中，我将介绍如何在 Struts 应用程序中集成和使用 Velocity 模板引擎。我会首先提供一个公式，然后逐步展开它。生成的应用程序组合了 Struts 和 Velocity —— 一个第一流的组合，可能会让您怀疑自己对 JSP 的忠诚！

请参阅 下载 一节，在开始之前下载这篇文章的源代码，以及 Struts、Velocity 和 Velocity 工具包。请注意，本文假设您熟悉使用 Struts 框架进行 MVC 编程。

关于模板引擎

在开始集成 Struts 和 Velocity 的简单任务之前，让我们先确保您理解模板引擎和它们在视图生成中的角色。模板引擎作为整体概念，Velocity 作为具体实现，它们的生命在 HTML 之外。Velocity 把数据合并到文本主体中不同的点上。文本可以是文字、电子邮件或 HTML。由于采用这种方式，Velocity 模板引擎有点儿像 Microsoft Word 的“邮件合并”特性。邮件合并允许您方便地把动态数据（例如姓名、地址和电话号码）合并到信件中。在早期的日子里，组织用这项特性生成大型邮件列表并把它们送到邮局，导致垃圾邮件的产生！

在 Web 应用程序中，Velocity 实现的目标与 JSP 相同：可以用它在向 `HttpServletResponse` 的 `OutputStream` 发送之前生成要发送的 HTML。在 Struts 应用程序中使用 Velocity 的一种方式是在 Struts 的 `Action` 内部写入响应，然后返回 `null` 的 `ActionForward`。虽然这种技术可行，但却有严重的缺陷：无法使用 *struts-config.xml* 文件把响应抽象出来。把视图放在 `Action` 内部，意味着如果想要修改响应，就必须修改 `Action`。

因为这种技术剥夺了 Struts 最好的一项特性（即从视图中抽象出重点的能力），所以我更愿意把所有响应指向一个 servlet，由它负责访问 Velocity 模板，合并上下文的数据，生成响应，然后再送回浏览器。稍后就会学到，Velocity 的设计者们已经把这些步骤全都捆绑在了一起：您需要做的只是跟着我来看如何一步步地实现它们。如果您还没有 访问“下载”一节，现在是访问的时候了。

Velocity 是什么？

Velocity 是一个基于 Java 的模板引擎，它提供了简单的基于模板的语言，可以用类似脚本的方式引用对象。Velocity 促进了分离团队成员之间的责任：允许 Web 设计人员专注于视图（即页面的观感），而 Java 程序员专注于后端代码。把 Java 代码从页面布局中分离出来，会让 Web 应用程序未来更易维护。当 Velocity 与 Sruts 这样的 MVC 框架结合时，就成了 JSP 或 PHP 可行的替代。

Velocity 的五步

把 Struts 与 Velocity 模板引擎组合起来很简单，也很直接；实际上，只要用五步就可以实现：

1. 把 Velocity JAR 放在类路径中。
2. 修改 web.xml 文件让它识别 Velocity servlet。
3. 把 Velocity toolbox.xml 放在应用程序的 WEB-INF 目录下。
4. 修改 struts-config，把它的视图指向 Velocity 模板而不是 JSP。
5. 为每个需要显示的页面创建 Velocity 模板。

我将用一个熟悉的搜索用例来演示 Struts 与 Velocity 的集成。在这个示例中，一个简单的应用程序允许用户按照图书的 ISBN 编号搜索图书。应用程序的结果页面显示与 ISBN 编号匹配的图书。

第 **1** 步：把 **Velocity JAR** 放在 **WEB-INF/lib** 下

如果您还没下载 Velocity，那么现在需要下载它。Velocity 本身是很棒的，但是它的工具包可以帮助您把工作做得更好更快。特别是 Struts 工具模拟了您以前熟悉的 Struts 标记。请参阅 下载 一节下载 Velocity 模板引擎和 Velocity 工具。

请注意不同时候，需要的 jar 也会略有不同。在这里我不想列出一个 JAR 列表，只是想建议您访问 Velocity 的主页（请参阅 参考资料）并阅读那里的安装指南。一旦得到了需要的 JAR，只需把它们放在 WEB-INF\lib 下面即可。

> 放弃 **Struts** 标记 —— 不!
>
> 现在，您可能会想，是不是需要放弃那些过去让您节约了许多编码时间的很好的 Struts 标记。如果不使用 JSP，那么肯定没有使用 Struts 的 JSP 标记！幸运的是，您可以使用 Velocity 工具。Velocity 的 Struts 工具提供了所有您熟悉的 Struts 方便特性，但是添加了 Velocity 的灵活性。

第 **2** 步：修改 **web.xml**，让它识别 **Velocity** 的 **servlet**

下一步是修改 Struts 的 web.xml 文件，让它识别 Velocity 的 servlet 并把所有以 *.vm* 结尾的资源请求定向到 Velocity servlet，如清单 1 所示。

清单 **1.** 修改 **web.xml**，声明 **Velocity servlet**

```
<servlet>
  <servlet-name>velocity</servlet-name> |(1)
  <servlet-class> |(2)
         org.apache.velocity.tools.view.servlet.VelocityViewServlet
  </servlet-class>

  <init-param> |(3)
    <param-name>org.apache.velocity.toolbox</param-name>
    <param-value>/WEB-INF/toolbox.xml</param-value>
 </init-param>

 <load-on-startup>10</load-on-startup> |(4)
</servlet>

<!-- Map *.vm files to Velocity -->
```

```
<servlet-mapping> |(5)
  <servlet-name>velocity</servlet-name>
  <url-pattern>*.vm</url-pattern>
</servlet-mapping>
```

让我们来看看清单 1 中发生了什么：

- （1）声明了 Velocity servlet 并给了它一个 *velocity* 句柄。
- （2）声明了 Velocity servlet 的类名。

Velocity servlet 接受“toolbox”参数。toolbox 是声明应用程序的可用工具的位置。因此，在清单 1 中，我还做了以下工作：

- （3）告诉 `VelocityServlet` 在哪里可以找到 toolbox 的配置。
- （4）设置了 `load-on-startup` 标记，确保在正确的时间装入 Velocity servlet。任何大于或等于 0 的值都会迫使容器通过调用 servlet 的 `init()` 方法来装入它。放在 `load-on-startup` 标记体中的值决定了不同的 servlet 的 `init` 方法调用的次序。例如，0 在 1 之前调用，而 1 在 2 之前调用。缺少的标记或负值允许 servlet 容器根据自己的选择装入 servlet。
- （5）声明了 servlet 映射，强迫所有用 *.vm* 结尾的资源请求定向到 Velocity servlet。请注意（5）中的`<servlet-name>` 必须与（1）中的`<servlet-name>` 匹配。交错的声明和映射会在日志中生成错误。

第 **3** 步：把 **toolbox.xml** 放在 **WEB-INF** 下

利用 Velocity，可以使用（或创建）包含许多工具的工具箱。用来登记类的工具箱中包含有用的函数，常常会用到。幸运的是，Velocity 提供了许多预先构建好的工具。还创建了许多 Struts 工具来模拟原始的 Struts 标记。如果发现需要构建自己的工具，也可以自由地构建。在清单 2 中显示的 toolbox.xml 可以在 Velocity 工具下载中找到。这个文件应当随 Velocity JAR 一起放在 WEB-INF 下。

清单 **2. toolbox.xml**

```
<?xml version="1.0"?>
<toolbox>
  <tool>
     <key>link</key>
     <scope>request</scope>
     <class>
       org.apache.velocity.tools.struts.StrutsLinkTool
     </class>
  </tool>
  <tool>
     <key>msg</key>
     <scope>request</scope>
     <class>
       org.apache.velocity.tools.struts.MessageTool
```

```
    </class>
  </tool>
  <tool>
    <key>errors</key>
    <scope>request</scope>
    <class>
      org.apache.velocity.tools.struts.ErrorsTool
    </class>
  </tool>
  <tool>
    <key>form</key>
    <scope>request</scope>
    <class>
      org.apache.velocity.tools.struts.FormTool
    </class>
  </tool>
  <tool>
    <key>tiles</key>
    <scope>request</scope>
    <class>
      org.apache.velocity.tools.struts.TilesTool
    </class>
  </tool>
  <tool>
    <key>validator</key>
    <scope>request</scope>
    <class>
      org.apache.velocity.tools.struts.ValidatorTool
    </class>
  </tool>
</toolbox>
```

第 **4** 步：修改 **struts-config**

下一步是修改 struts-config.xml，指向 Velocity 视图而不是 JSP。新的配置文件如清单 3 所示。

清单 **3.** 针对 **Velocity** 视图修改后的 **struts-config.xml**

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
```

```
<!DOCTYPE struts-config PUBLIC
          "-//Apache Software Foundation//DTD Struts Configuration 1.0//EN"
          "http://jakarta.apache.org/struts/dtds/struts-config_1_0.dtd">

<struts-config>
    <form-beans>
        <form-bean name="searchForm" type="app.SearchForm"/>
    </form-beans>

    <global-forwards>
        <forward name="welcome" path="/welcome.do"/>
    </global-forwards>

   <action-mappings>
        <action
            path="/welcome"
            type="org.apache.struts.actions.ForwardAction"
            parameter="/pages/search.vm"/> |(1)

        <action
            path="/search"
            type="app.SearchAction"
            name="searchForm"
            scope="request"
            input="/pages/search.vm"> |(2)
            <forward name="success"
              path="/pages/results.vm"/> |(3)
        </action>
    </action-mappings>
</struts-config>
```

清单 3 看起来就像一个非常典型的 Struts 应用程序，只有一个小小的不同。响应没有把客户转向到 JSP，而直接转向到 *.vm* 文件（请参阅清单 3 中的引用 1、2 和 3）。在大多数情况下，把 Struts 应用程序从 JSP 迁移到 Velocity 视图，需要做的仅仅是全局搜索，把 *.jsp* 替换成 *.vm*。其他所有东西都可以保持不变！模板可以同样保存在以前保存 JSP 的位置；所需要做的只是用 Velocity 命令代替 JSP 标记。

第 **5** 步：创建 **Velocity** 模板

在清单 4 中，可以看到示例应用程序搜索页面的 Velocity 模板。

清单 **4.** 搜索页面的 **Velocity** 模板

```
<HTML>
  <HEAD>
    <TITLE>Search</TITLE>
  </HEAD>
  <BODY>
    $!errors.msgs()||(1)
    <FORM method="POST"
      action="$link.setAction('/search')"> |(2)
      <h2>Book Search</h2>
      ISBN:<INPUT type="text" name="isbn">
      <INPUT type="submit" value="Submit" name="submit">
    </FORM>
  </BODY>
</HTML>
```

清单 4 是一个没有 JSP 或 Struts 标记的典型的 HTML 页面。但是，以下元素看起来可能不是那么熟悉：

- （1）用 `$!errors.msgs()` 得到错误消息队列中的错误消息。
- （2）用 `$link.setAction('/search')` 获得搜索转发的 URL。

这就成功了 —— 模板剩下的部分看起来几乎与以前熟悉的 HTML 文件相同。清单 5 显示了应用程序结果页面的模板。

清单 **5.** 结果页面的 **Velocity** 模板

```
<html>
  <body>

  <h1>Book Details</h1>
  <a href="$link.setForward("searchEntry")">Search again</a> |(1)

  <h3>$book.title</h3> |(2)

    <b>ISBN:</b>$book.isbn<br>|(3)
    <b>Title:</b>$book.title<br>|(4)
    <b>Author:</b>$book.author<br>|(5)
    <b>Price:</b>$book.price<br>|(6)
    <b>No Pages:</b>$book.pages<br>|(7)
    <b>Description:</b>$book.description<br>|(8)
    <b>Publisher:</b>$book.publisher<br>|(9)
  </body>
<html>
```

可以注意到，清单 5 中不包含 JSP 标记或 Struts 标记。我们来详细看看它：

- （1）用 Struts 的链接工具把 <a> 标记的 *href* 设置为 Struts 转发。
- （2）访问 `$book title` 属性。
- （3）访问 `$book isbn` 属性。
- （4）再次访问 `$book title` 属性。
- （5）访问 `$book author` 属性。
- （6）访问 `$book price` 属性。
- （7）访问 `$book pages` 属性。
- （8）访问 `$book description` 属性。
- （9）访问 `$book publisher` 属性。

回页首

讨论

这就是把 Struts 与 Velocity 模板引擎集成的全部工作。表面看起来非常简单（实际上也很简单），但是请想想是什么让这个集成能够工作的呢？

Struts 动作映射可以定义任何视图，不仅限于 JSP。在这篇文章中，我只是把动作映射修改为以 *vm* 结尾而不是以 *jsp* 结尾的返回文件。然后，我声明了 Velocity servlet，并告诉 Servlet 容器把以 *vm* 结尾的文件发送给 `VelocityViewServlet`。

`VelocityViewServlet` 把 Velocity 命令表示成 HTML 响应。通过这种方式，`VelocityViewServlet` 充当了视图响应的拦截器。Struts 控制器把视图转发给 `VelocityViewServlet`，后者在向客户端发送响应之前处理 *vm* 文件。请参阅 参考资料 获得关于将 Velocity 视图集成进 Struts 应用程序的更多内容。

回页首

结束语

正如在本文中看到的，Struts 与 Velocity 的集成很简单。只需五个步骤就可以把所有东西连在一起。针对不同的引擎和场景，采用模板引擎而不是 JSP 的优势各有不同。在 Velocity 的情况下，优势就是简单性、容易学习以及更好的性能。

回页首

下载

描述	名字	大小	下载方法
Sample code	j-sr1-source.zip	3 MB	**FTP**

➔ 关于下载方法的信息

参考资料

学习

- 您可以参阅本文在 developerWorks 全球站点上的 英文原文。
- “使用 Velocity 实现客户端和服务器端模板”（Sing Li，developerWorks，2004 年 2 月）：学习关于 Velocity 和它与 Web 应用程序集成的更多内容。
- “Developing Struts with Easy Struts for Eclipse”（Nancy Junhua，developerWorks，2004 年 4 月）：利用 Easy Struts 插件的帮助在 Eclipse IDE 中开发 Struts 应用程序。
- “集成 Struts、Tiles 和 JavaServer Faces”（Srikanth Shenoy 和 Nithin Mallya，developerWorks，2003 年 9 月）：对 Struts 友好的各种技术的概述。
- Struts Recipes （George Franciscus 和 Danilo Gurovich，Manning，2004 年）：对 Struts 技术和最佳实践的概述。
- Struts In Action （Ted Husted，Cedric Dumoulin，George Franciscus，David Winterfeldt，Manning，2002 年）：针对专业 Struts 开发人员的全面的参考资料。
- Java 技术专区：寻找 Java 编程各方面的文章。

获得产品和技术

- Struts framework：Apache 软件基金会的一个项目。
- Velocity Template Engine：针对 MVC 框架的一个基于 Java 的模板引擎。
- Velocity Struts Tools：在 Velocity 中使用熟悉的 Struts 工具。

讨论

- developerWorks blogs：加入 developerWorks 社区。

关于作者

George Franciscus 是一位 Java 企业级顾问和 Struts 权威。他是 Manning 的 Struts Recipes 和 Struts in Action 两本书的合著者。George 通过 nexcel.ca 提供技术和管理咨询服务。

对本文的评价

- 太差！ (1)
- 需提高 (2)
- 一般；尚可 (3)
- 好文章 (4)
- 真棒！(5)

建议？

反馈意见

回页首