Servlet 应用体系结构

作者：importGuitar
原文链接：https://ld246.com/article/1573009608101
来源网站：链滴
许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

<h3 id="容器">容器</h3>
<blockquote>
Servlet 没有 main()方法。它们受控于另一个 Java 应用，这个 Java 应用称为容器。
</blockquote>
Tomcat 就是一个容器。如果向服务器请求一个 Servlet，此时服务器会把请求交给部署该 Servle
的容器，容器向 Servlet 提供 HTTP 请求和响应，且由容器调用 Servlet 方法，如 doGet()或 doPost(
。
<h4 id="容器能作甚-">容器能作甚？</h4>
1.通信支持。容器知道自己与 web 服务器之间的协议，让 Servlet 与服务器轻松对话。我们不
再担心 web 服务器和我们的 web 代码之间的 API。
2.生命周期管理。容器负责加载类、实例化和初始化 Servlet、调用 Servlet 方法，并使 Servlet 能够
垃圾回收。
3.多线程支持。容器会自动为它接收的每个 Servlet 请求创建一个新的 Java 线程。
4.声明方式实现安全。利用容器，可以使用 xml 部署描述文件来配置和修改安全性。
5.JSP 支持。容器负责把 JSP 代码翻译成真正的 Java。
<h4 id="容器如何处理请求">容器如何处理请求</h4>
1.用户点击一个链接，其 URL 指向一个 Servlet 而不是静态页面。
2.容器“看出来”这个请求指向一个 Servlet，所以容器会创建两个对象：HttpServletResponse 和 H
tpServletRequest。
3.容器根据请求中的 URL 找到正确的 Servlet，为这个请求创建或分配一个线程，并把请求和响应对
传递给这个 Servlet 线程。
4.容器调用 Servlet 的 service()方法（在这之前还要加载类、实例化和初始化 Servlet）。根据请求
不同类型，service()方法会调用 doGet()或 doPost()方法。此处假设调用的 doGet()。
5.doGet()方法生成动态页面，并把页面填入响应对象。
6.线程结束，容器把响应对象转换为一个 HTTP 响应，把他发给客户，然后删
请求和响应对象。
<h3 id="映射">映射</h3>
<h4 id="Servlet的三个名字">Servlet 的三个名字</h4>
<pre><code class="language-java highlight-chroma"><span c
ass="highlight-cl"><servle
>
  <span class=
highlight-o"><servlet<span class="highlight-
">-name></spa
>ListenerTester</
span class="highlight-n">servlet-<span class="h
ghlight-n">name>
  <span class=
highlight-o"><servlet<span class="highlight-
">-class></span
com.<span class="h
ghlight-na">shangxiaoying.<span class="highli
ht-na">ListenerTesterServlet</<span class="h
ghlight-n">servlet-c
ass>
  <span class="high
ight-o"></servlet
gt;
  <span class="high
ight-o"><servlet-
/span>mapping>
  <span class=
highlight-o"><servlet<span class="highlight-
">-name></spa

>ListenerTester</
span class="highlight-n">servlet-<span class="h
ghlight-n">name>
  <span class=
highlight-o"><url
pattern>/</span
ListenTest.<span cla
s="highlight-na">do</<span class="highlight
n">url-pattern</spa
>>
<span class="high
ight-o"></servlet<span class="highlight-o">
mapping></spa
>
</code></pre>
说明：
<code><servlet-name></code> 元素用于把一个 <code><servlet></code> 元素绑到一个特定的 <code><servlet-mapping></code> 元素。这个名只在这个部署文件中使用。br>
<code> <servlet-class></code> 元素内放完全限定类名（但是不要加.class 扩展名）。
<code> <url-pattern></code> 元素内是用户看到并使用的 Servlet 名，这是虚构的，并不具体 Servlet 类的名字。
当请求到来时，容器会根据 <code><servlet-mapping></code> 里的 <code><url-pttern></code> 找到与请求的 URL 对应的 <code><servlet-mapping></code> 元素，后根据 <code><servlet-mapping></code> 里的 <code><servlet-name></code> 素找到对应的 <code><servlet></code>,然后根据 <code><servlet></code> 里的 <ode> <servlet-class></code> 找到具体的 Servlet 类。
<h3 id="MVC">MVC</h3>
<blockquote>
MVC 的关键是业务逻辑与表示分离，而且要在二者之间放上别的东西，这样业务逻辑本身就能为一个可重用的 Java 类存在，它根本不用对视图有任何了解。
</blockquote>
MVC 就是把业务逻辑从 Servlet 中抽出来放在一个“模型”中，
模型（Model）就是可重用的普通 Java 类，是业务数据和处理该数据的方法的组合。系统中只有这分与数据库通信。
视图（View）负责表示，从控制器获取模型状态，还要获得用户输入，并交给控制器。
控制器（Controller），从请求获得用户输入，并明确这些输入对模型有什么影响；告诉模型自更新，并让视图得到新的模型状态。