Servlet 包 Servlet 国际化

Servlet 调试

测试/调试 Servlet 始终是开发使用过程中的难点。Servlet 往往涉及大量的客户端/服务器交互，可能会出现错误但又难以重现。

这里有一些提示和建议，可以帮助您调试。

System.out.println()

System.out.println() 是作为一个标记来使用的，用来测试一段特定的代码是否被执行。我们也可以打印出变量的值。此外：

- 由于 System 对象是核心 Java 对象的一部分，它可以在不需要安装任何额外类的情况下被用于任何地方。这包括 Servlet、JSP、RMI、EJB's、普通的 Beans 和类，以及独立的应用程序。
- 与在断点处停止不同，写入到 System.out 不会干扰到应用程序的正常执行流程，这使得它在时序是至关重要的时候显得尤为有价值。

下面是使用 System.out.println() 的语法：

```
System.out.println("Debugging message");
```

通过上面的语法生成的所有消息将被记录在 Web 服务器日志文件中。

消息日志

使用适当的日志记录方法来记录所有调试、警告和错误消息，这是非常好的想法，推荐使用 log4J 来记录所有的消息。

Servlet API 还提供了一个简单的输出信息的方式，使用 log() 方法，如下所示：

```
// 导入必需的 java 库
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class ContextLog extends HttpServlet {
  public void doGet(HttpServletRequest request,
      HttpServletResponse response) throws ServletException,
         java.io.IOException {

      String par = request.getParameter("par1");
      // 调用两个 ServletContext.log 方法
      ServletContext context = getServletContext( );

      if (par == null || par.equals(""))
      // 通过 Throwable 参数记录版本
      context.log("No message received:",
```

```
            new IllegalStateException("Missing parameter"));
        else
            context.log("Here is the visitor's message: " + par);

        response.setContentType("text/html;charset=UTF-8");
        java.io.PrintWriter out = response.getWriter( );
        String title = "Context Log";
        String docType = "<!DOCTYPE html> \n";
        out.println(docType +
          "<html>\n" +
          "<head><title>" + title + "</title></head>\n" +
          "<body bgcolor=\"#f0f0f0\">\n" +
          "<h1 align=\"center\">" + title + "</h1>\n" +
          "<h2 align=\"center\">Messages sent</h2>\n" +
          "</body></html>");
      } //doGet
  }
```

ServletContext 把它的文本消息记录到 Servlet 容器的日志文件中。对于 Tomcat，这些日志可以在 <Tomcat-installation-directory>/logs 目录中找到。

这些日志文件确实对新出现的错误或问题的频率给出指示。正因为如此，建议在通常不会发生的异常的 catch 子句中使用 log() 函数。

使用 JDB 调试器

您可以使用调试 applet 或应用程序的 jdb 命令来调试 Servlet。

为了调试一个 Servlet，我们可以调试 sun.servlet.http.HttpServer，然后把它看成是 HttpServer 执行 Servlet 来响应浏览器端的 HTTP 请求。这与调试 applet 小程序非常相似。与调试 applet 不同的是，实际被调试的程序是 sun.applet.AppletViewer。

大多数调试器会自动隐藏如何调试 applet 的细节。同样的，对于 servlet，您必须帮调试器执行以下操作：

- 设置您的调试器的类路径 classpath，以便它可以找到 sun.servlet.http.Http-Server 和相关的类。
- 设置您的调试器的类路径 classpath，以便它可以找到您的 servlet 和支持的类，通常是在 server_root/servlets 和 server_root/classes 中。

您通常不会希望 server_root/servlets 在您的 classpath 中，因为它会禁用 servlet 的重新加载。但是这种包含规则对于调试是非常有用的。它允许您的调试器在 HttpServer 中的自定义 Servlet 加载器加载 Servlet 之前在 Servlet 中设置断点。

如果您已经设置了正确的类路径 classpath，就可以开始调试 sun.servlet.http.HttpServer。可以在您想要调试的 Servlet 代码中设置断点，然后通过 Web 浏览器使用给定的 Servlet（http://localhost:8080/servlet/ServletToDebug）向 HttpServer 发出请求。您会看到程序执行到断点处会停止。

使用注释

代码中的注释有助于以各种方式进行调试。注释可用于调试过程的很多其他方式中。

该 Servlet 使用 Java 注释和单行注释（//...），多行注释（/* ...*/）可用于暂时移除部分 Java 代码。如果 bug 消失，仔细看看您刚才注释的代码并找出问题所在。

客户端和服务器端头信息

有时，当一个 Servlet 并没有像预期那样时，查看原始的 HTTP 请求和响应是非常有用的。如果您熟悉 HTTP 结构，您可以阅读请求和响应，看看这些头信息究竟是什么。

重要的调试技巧

下面列出了一些 Servlet 调试的技巧：

- 请注意，server_root/classes 不会重载，而 server_root/servlets 可能会。
- 要求浏览器显示它所显示的页面的原始内容。这有助于识别格式的问题。它通常是"视图"菜单下的一个选项。
- 通过强制执行完全重新加载页面来确保浏览器还没有缓存前一个请求的输出。在 Netscape Navigator 中，请使用 Shift-Reload，在 Internet Explorer 中，请使用 Shift-Refresh。
- 请确认 servlet 的 init() 方法接受一个 ServletConfig 参数，并调用 super.init(config)。

← Servlet 包 Servlet 国际化 →