# 尚马教育 JAVA WEB课程

# Servlet详解

文档编号：B12

创建日期： 2017-06-13

最后修改日期：2019-09-23

版 本 号：V2.0

电子版文件名：尚马教育-第二阶段-2.servlet详解.docx

**文档修改记录：**

|  |  |  |  |
| --- | --- | --- | --- |
| 更新日期 | 更新作者 | 更新说明 | 版本号 |
| 2018-08-01 | 王绍成 | 初始版本 | V1.0 |
| 2019-09-13 | 郭雪岩 | 内容更新 | V2.0 |

**主讲人：**

**郭雪岩**

目录

[1. Servlet线程特性](#_Toc29591_WPSOffice_Level1) [3](#_Toc29591_WPSOffice_Level1)

[2. Servlet域对象](#_Toc30013_WPSOffice_Level1) [4](#_Toc30013_WPSOffice_Level1)

[3. Servlet中常用类和方法](#_Toc12268_WPSOffice_Level1) [5](#_Toc12268_WPSOffice_Level1)

[3.1. ServeltContext接口](#_Toc30013_WPSOffice_Level2) [5](#_Toc30013_WPSOffice_Level2)

[3.2. ServletConfig接口](#_Toc12268_WPSOffice_Level2) [5](#_Toc12268_WPSOffice_Level2)

[3.3. HttpRequest（很常用）](#_Toc24660_WPSOffice_Level2) [6](#_Toc24660_WPSOffice_Level2)

[3.4. HttpResponse](#_Toc6127_WPSOffice_Level2) [7](#_Toc6127_WPSOffice_Level2)

[3.5. HttpSession](#_Toc6169_WPSOffice_Level2) [7](#_Toc6169_WPSOffice_Level2)

## Servlet线程特性

* 肯定会有多个客户端同时请求访问一个Servlet，Web服务器怎么处理多个请求呢？

多用户访问时，servlet会针对每个请求开启新线程来处理数据收发，因为对每个请求的处理是固定的，一次收发请求就结束。

* 第一次访问Servlet时，服务器将创建一个该Servlet类的对象，并调用doXXX方法生成响应；多个客户端访问同一个Servlet时，不再创建新的对象，而是共用同一个Servlet对象。可以说，Servlet是多线程单实例的。

注意：多线程访问单实例的情况下，会不会产生线程安全问题？

每个线程间，栈是独立的，运行方法时，会压入栈帧，所以运行方法时线程间不会互相影响。

因为单实例，所以如果servlet中有成员变量，那么所有线程都会访问到同一个实例的成员变量，会造成线程安全问题。

所以使用servlet时，尽量不要建立成员变量。

## Servlet域对象

* 域对象，是指服务器交互过程中可以用来存储数据的空间，相对独立，生效的时机不同，都可以用来存储信息
* Servlet三大域对象的应用
* Request

一个用户可有多个

每次请求都会产生一个

请求结束对象消失

建议用于页面间值的传递

* Session

一个用户一个

每一个浏览器都会产生一个

会话过期或者关闭浏览器对象消失

建议存放一些用户信息，不要把过多的信息存放在session里

* ServletContext（application）

全局只有一个

是一个全局的储存信息的空间，服务器开始就存在，服务器关闭才释放。

为了节省空间，提高效率，只存放少量的重要信息

域对象统一方法：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | **void** | **setAttribute(String key,Object value)** | **以key/value的形式**  **保存对象值** | | **Object** | **getAttribute(String key)** | **通过key获取对象值** | |

## Servlet中常用类和方法

### ServeltContext接口

|  |  |
| --- | --- |
| 方法 | 说明 |
| public void setAttribute  (String name，Object object) | 绑定一个java对象和一个属性名，并存放到ServletContext中，  参数name指定属性名，参数Object表示共享数据 |
| pulbic Object getAttribute  (String name) | 根据参数给定的属性名，返回一个Object类型的对象 |
| public String getContextpath() | 返回当前web应用的URL入口 |
| public String getInitParameter  (String name) | 返回web应用范围内匹配的初始化参数值。在web.xml中，  <web-app>元素中的<context-param>元素表示应用范围内的初始化参数 |

### ServletConfig接口

|  |  |
| --- | --- |
| 方法 | 说明 |
| public String getInitParameter(String path) | 获取web.xml中指定Servlet的初始化参数值 |
| public ServletContext getServletContext() | 获取ServletContext实例 |
| public String getServletName() | 获取当前Servlet的名称 |

### HttpRequest（很常用）

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 方法 | 说明 | | public String getParameter  (String name) | 获取页面提交指定名称的参数值 | | public String[] getParameterValues(String name) | 获取页面提交相同名称参数的数组值 | | public void  setCharacterEncoding("UTF-8") | 设置字符编码格式 | | public Map getParameterMap() | 返回一个保存了请求的所有参数和值的Map对象 | | public void setAttribute  (String name,Object obj) | 向request范围内设置属性，参数name为属性名，obj为属性值 | | public Object getAttribute  (String name) | 返回一个请求中属性名为name的属性值 | | public String getContextPath() | 返回当前Web项目的根路径 | |

* Request对象中封装了很多获取跟http相关的路径的方法

|  |
| --- |
| request.getContextPath(); 项目名/WebDemo  request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+request.getContextPath()+"/";  结果如下：http://localhost:8080/WebDemo/  request.getRemoteAddr() 客户端ip地址  request.getServletPath() 请求的servlet的地址/student  request.getRealPath("/") 在tomcat中的路径  request.getRequestURI(); 获得请求路径 /web\_demo/student |

### HttpResponse

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 方法 | 说明 | | Public PrintWriter getWriter() | 获取响应的打印输出流对象 | | Public void setContentType  ("text/html;charset=UTF-8") | 设置响应内容的类型和字符编码格式 | |

Response对象中主要封装跟输出信息相关的方法。

### HttpSession

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | 方法 | 说明 | | setAttribute  (String key,Object value) | 以key/value的形式保存对象值 | | getAttribute(String key) | 通过key获取对象值 | | getMaxInactiveInterval() | 获取session的有效非活动时间，以秒为单位 | | getId() | 获取session对象的编号 | | invalidate() | 设置session对象失效 | |

* Session使用

HttpSession session = request.getSession();从request对象中获取会话(session)对象每个浏览器都会产生单独的session对象

session作为会话对象，经常用来保存用户的相关登录信息，方便服务器调用（用户本身信息，权限信息等）

* 例：用户登录，通过session做是否已登录的访问控制