Servlet(Server Applet)是Java Servlet的简称, 称为小服务程序或服务连接器, 用Java编写的服务器端程序, 主要功能在于交互式地浏览和修改数据, 生成动态Web内容。

狭义的Servlet是指Java语言实现的一个接口, 广义的Servlet是指任何实现了这个Servlet接口的类, 一般情况下, 人们将Servlet理解为后者。

Servlet运行于支持Java的应用服务器中。

从原理上讲, Servlet可以响应任何类型的请求, 但绝大多数情况下Servlet只用来扩展基于HTTP协议的Web服务器。

最早支持Servlet标准的是JavaSoft的Java Web Server, 此后, 一些其它的基于Java的Web服务器开始支持标准的Servlet。



Servlet 是在[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8)上运行的小程序。

这个词是在Java [applet](https://baike.baidu.com/item/applet)的环境中创造的, Java applet是一种当作单独文件跟网页一起发送的小程序, 它通常用于在客户端运行, 结果得到为用户进行运算或者根据用户互作用定位图形等服务。

服务器上需要一些程序, 常常是根据用户输入访问数据库的程序。

这些通常是使用[公共网关接口](https://baike.baidu.com/item/%E5%85%AC%E5%85%B1%E7%BD%91%E5%85%B3%E6%8E%A5%E5%8F%A3)(**C**ommon **G**ateway **I**nterface, CGI)应用程序完成的。

然而, 在服务器上运行Java, 这种程序可使用Java编程语言实现。

在通信量大的服务器上, JavaServlet的优点在于它们的执行速度更快于CGI程序。

各个用户请求被激活成单个程序中的一个线程，而**无需**创建单独的进程, 这意味着[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8)端处理请求的[系统开销](https://baike.baidu.com/item/%E7%B3%BB%E7%BB%9F%E5%BC%80%E9%94%80)将明显降低。

**实现过程**

最早支持Servlet技术的是JavaSoft的Java Web Server。

此后, 一些其它的基于Java的Web Server开始支持标准的Servlet API。

Servlet的主要功能在于交互式地浏览和修改数据, 生成动态Web内容.

这个过程为:

(1).客户端发送请求至服务器端

(2).服务器将请求信息发送至 Servlet

(3).Servlet生成响应内容并将其传给[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8), 响应内容动态生成, 通常取决于客户端的请求;

(4).服务器将响应返回给客户端

Servlet 看起来像是通常的 Java 程序

Servlet 导入特定的属于 Java Servlet API 的包。

因为是对象[字节码](https://baike.baidu.com/item/%E5%AD%97%E8%8A%82%E7%A0%81), 可动态地从网络加载, 可以说Servlet对Server就如同Applet对Client一样, 但是, 由于Servlet运行于Server中, 它们并不需要一个[图形用户界面](https://baike.baidu.com/item/%E5%9B%BE%E5%BD%A2%E7%94%A8%E6%88%B7%E7%95%8C%E9%9D%A2), 从这个角度讲, Servlet也被称为FacelessObject。

一个Servlet就是Java编程语言中的一个类, 它被用来扩展[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8)的性能, 服务器上驻留着可以通过“请求-响应”编程模型来访问的应用程序。

虽然Servlet可以对任何类型的请求产生响应, 但通常只用来扩展Web服务器的应用程序。

目前最新版本为4.0。

Servlet的命名可以看出sun命名的特点, 如Applet表示小应用程序;

Scriptlet = Script + Applet, 表示小脚本程序;

同样Servlet = Service + Applet, 表示小服务程序。

生命周期

客户端请求该 Servlet;

加载 Servlet 类到内存;

实例化并调用init()方法初始化该 Servlet;

service() (根据请求方法不同调用doGet()或者doPost(), 此外还有doHead()、doPut()、doTrace()、doDelete()、doOptions()、destroy());

加载和实例化Servlet, 这项操作一般是动态执行的, 然而, Server通常会提供一个管理的选项, 用于在Server启动时强制装载和初始化特定的Servlet;

Server 创建一个 Servlet的实例

第一个客户端的请求到达 Server

Server调用Servlet的init()方法, (可配置为Server创建Servlet实例时调用, 在web.xml中<servlet>标签下配置<load-on-startup>标签, 配置的值为整型, 值越小Servlet的启动优先级越高);

一个客户端的请求到达 Server;

Server创建一个请求对象, 处理客户端请求;

Server创建一个响应对象, 响应客户端请求;

Server激活Servlet的service()方法, 传递请求和响应对象作为参数;

service()方法获得关于请求对象的信息, 处理请求, 访问其他资源, 获得需要的信息!

service()方法使用响应对象的方法, 将响应传回Server, 最终到达客户端。

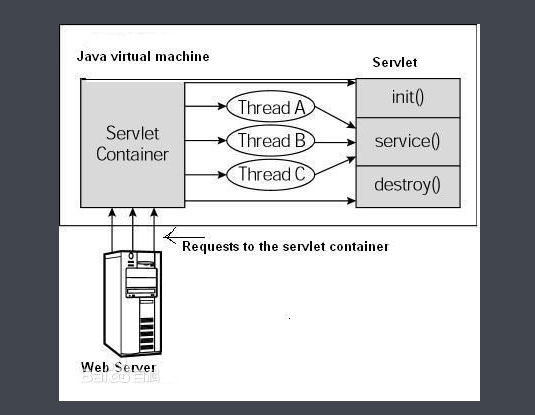
service()方法可能激活其它方法以处理请求, 如doGet()或doPost()或程序员自己开发的新的方法;

对于更多的客户端请求, Server创建新的请求和响应对象, 仍然激活此Servlet的service()方法, 将这两个对象作为[参数传递](https://baike.baidu.com/item/%E5%8F%82%E6%95%B0%E4%BC%A0%E9%80%92)给它。

如此重复以上的循环, 但无需再次调用init()方法。

一般Servlet只初始化一次(**只有一个对象**), 当Server不再需要Servlet时(一般当Server关闭时), Server调用Servlet的destroy()方法。

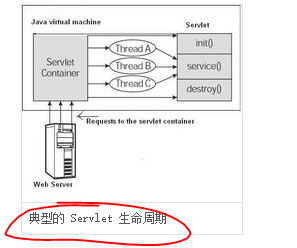
下图显示了一个典型的 Servlet 生命周期方案:



(1).第一个到达服务器的HTTP请求被委派到Servlet容器;

(2).Servlet容器在调用service()方法之前加载Servlet;

(3).然后Servlet容器处理由多个线程产生的多个请求, 每个线程执行一个单一的Servlet实例的service()方法。



## 工作模式

## 比较

## 版本

## 规范

## 编程接口

[HTTPServlet](https://baike.baidu.com/item/HTTPServlet) 使用一个 HTML 表单来发送和接收数据。

要创建一个 HTTPServlet，请扩展 [HttpServlet](https://baike.baidu.com/item/HttpServlet) 类， 该类是用专门的方法来处理 HTML [表单](https://baike.baidu.com/item/%E8%A1%A8%E5%8D%95)的 GenericServlet 的一个子类。

 HTML 表单是由 <form> 和 </form> 标记定义的。

表单中典型地包含输入字段(如文本输入字段、[复选框](https://baike.baidu.com/item/%E5%A4%8D%E9%80%89%E6%A1%86)、[单选按钮](https://baike.baidu.com/item/%E5%8D%95%E9%80%89%E6%8C%89%E9%92%AE)和选择列表)和用于提交数据的按钮。

当提交信息时, 它们还指定[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8)应执行哪一个Servlet（或其它的程序）。

 HttpServlet 类包含 init()、destroy()、service() 等方法。其中 init() 和 destroy() 方法是[继承](https://baike.baidu.com/item/%E7%BB%A7%E6%89%BF)的。

(1).

(2).

(3).

(4).

## 

## 常见容器

Tomcat, Jetty, resin, Oracle Application server, WebLogic Server, Glassfish, Websphere, JBoss 等等。

(提供了Servlet功能的服务器, 叫做Servlet容器, 对web程序来说, Servlet容器的作用就相当于桌面程序里操作系统的作用, 都是提供一些编程基础设施).

## 建议

在Web应用程序中, 一个Servlet在一个时刻可能被多个用户同时访问。

这时Web容器将为每个用户创建一个线程来执行Servlet。

如果Servlet不涉及共享资源的问题, 不必关心多线程问题。

但如果Servlet需要共享资源, 需要保证Servlet是[线程安全](https://baike.baidu.com/item/%E7%BA%BF%E7%A8%8B%E5%AE%89%E5%85%A8)的。

下面是编写线程安全的Servlet的一些建议:

(1).用方法的局部变量保存请求中的专有数据。对方法中定义的局部变量，进入方法的每个线程都有自己的一份方法变量拷贝。任何线程都不会修改其他线程的局部变量。如果要在不同的请求之间共享数据，应该使用会话来共享这类数据。

(2).只用Servlet的成员变量来存放那些不会改变的数据, 有些数据在Servlet生命周期中不发生任何变化, 通常是在初始时确定的, 这些数据可以使用成员变量保存, 如数据库连接名称、其他资源的路径等。

(3).对可能被请求修改的成员变量同步, 有时数据成员变量或者环境属性可能被请求修改, 当访问这些数据时应该对它们同步, 以避免多个线程同时修改这些数据。

(4).如果Servlet访问外部资源, 那么需要同步访问这些资源, 例如, 假设Servlet要从文件中读写数据,

当一个线程读写一个文件时, 其他线程也可能正在读写这个文件。

文件访问本身不是线程安全的, 所以必须编写同步访问这些资源的代码。

在编写线程安全的Servlet时, 下面两种方法是不应该使用的:

a.在Servlet API中提供了一个SingleThreadModel 接口, 实现这个接口的Servlet, 在被多个客户请求时一个时刻只有一个线程运行, 这个接口已被标记不推荐使用。

b.对doGet()或doPost()方法同步, 如果必须在Servlet中使用同步代码, 应尽量在最小的代码块范围上进行同步, 同步代码越小, Servlet执行得才越好。