Java 的服务器端应用程序接口

罗涛 杨德华 张建军 (上海同济大学经济与管理学院 200092)

摘要:Java Servlet API 是 javaSoft 公司推出的服务器端应用程序接口类库.本文对 Servlet 的运行环境、特点与优势、以及多种应用方式作了介绍与归纳,并以一个通过 JDBC 查询数据库的典型 Servlet 应用为例,介绍了 Servlet 程序的基本框架和执行步骤。

关键词:Java Servlet API Servlet JDBC

除了编写在计算机上独立运行的应用程序(Java Application)以及发送到客户机浏览器上运行的应用小程序 Applet 外,Java 还可以编写基于 Web 应用的服务器端应用程序 - - Servlet,一种在服务器端执行请求/应答模式的服务程序。这种简单的、灵活的 Servlet API 可以使服务器具有强大的功能,成为推动企业网络模型革命的一个重要推动力量。在这场变革中,Web 模型从静态的文档发布系统转变为基于各种操作系统及 Web 服务器的动态的客户机/服务器平台。JavaSoft 公司的 Java Servlet API 使得这一变革得以加速实现。

一、Java Servlet API 概述

用 Java 编写的 Servlet 具有许多优异特性,比如字节代码的机制、输入流/输出流的处理、内置多线程、平台无关性、方便的 TCP/IP Socket 接口,以及对大多数数据类型的支持。Servlet 可以被看作为 Java Applet 的服务器端对应部分,与 Applet 运行在浏览器端不同,Servlet 是运行在服务器端的与协议与平台无关的程序模块。动态的支持 Java 的服务器端应用,为请求/应答模式提供了一个总体框架。

1. Servlet 的运行环境

运行 Servlet, 服务器端必须装载 Java 虚拟机(Java Virtual Machine, JVM).其次,这个服务器必须支持 Java Servlet API.目前,大多数的服务器都支持 Servlet API,有的服务器本身支持 Servlet[2], 比如 JavaSoft 的 Java Web Server(JWS), O'Reilley&Associate 的 WebSite Pro。对于一些通用的服务器,比如 Netscape 的 Enterprise 和 FastTrack, Microsoft 的 IIS, StarNine 的 WebStar、Apache 以及 IBM 的 Lotus Domino Go Server,则有 Gefion Software、Live Software、New Atalanta 以及 IBM 自己开发的 Servlet 驱动程序支持。有了运行 Servlet 的条件后,用 Servlet 形式的应用程序便可以在使用不同操作系统的 Web 服务器之间互相转移使用而无需做任何改动,这就是 Servlet 的平台无关性。在下面内容中将对 Servlet 的典型特点作详细的介绍。

2. Servlet 的特点

在 Servlet 出现之前,被广泛使用的服务器端应用程序接口是 CGI 程序。CGI 程序编写简单,并被广泛的支持,但令人失望的是,它不是非常有效,服务器每调用一次 CGI 程序,就必须启动一个新的 CGI 进程,对于需求比较密集、比较繁忙的服务器来说,CGI 就力不从心了。软件商们为此提出了一些解决方案,比如,ISAPI(Microsoft)、NSAPI(Netscape),以提高服务器的性能,然而不兼容问题又始终困扰着程序员和用户们。JavaSoft 推出的 Java Servlet API 可以说是解决这些问题的好办法,是一种标准的理想的 CGI 替代品。

作为 Java 扩展的 Java Servlet API 与 CGI 及其他 API 相比有如下特点[1][2]:

(1)多线程。Servlet 的运行机制是,进行第一次调用时,Java 虚拟机将 Servlet 程序装载入服务器的内存,Servlet 便驻留内存,直到服务器被关闭后才退出。以后其他的、并发的用户请求就可以使用同一个 Servlet 的实例,所不同的是它们启动了不同的线程,一般来说,Servlet 的线程池可以容纳 10 个并发的线程,即使是繁忙的需求也可以从容应付。这种运行机制比起每应答一个请求,便需要启动一个进程,执行后还要释放系统资源的方式,效率上的优势是不言而喻的。利用这一特性,服务器可以自如的利用一个线程不间断的完成搜集信息、监视指定资源、定时提供状态报告或者其他持久性的工作。如果要使用 CGI 程序来实现类似功能,解决方法则是非常复杂的。

(2)平台无关性。前面已提到过,只要服务器端装有 Java 虚拟机,并且服务器支持 Servlet API,Servlet 就可以未经改动的嵌入各种不同的服务器中。因为 Java 的平台无关性同样也适用于作为 Java 的扩展,Servlet API。对以往的编程人员来说,最痛苦的莫过于一遍又一遍的重复编写同样的程序代码,在一种服务器上编写的代码在另一种服务器上便不能运行。Servelt 的出现解决了这一问题,它的跨平台特性真正实现了"一次编写,处处运行"。利用这种特点,Servlet 应用程序可以成为标准的模块或组件,极大的提高软件重用性。比如,可以将网上交易的一些通用功能模块标准化,其他类似的应用便不用费时费力的重新开发。

(3)安全性。Servlet 是用 Java 语言编写的,Java 不允许内存存取越权侵犯和强制类型越权侵犯的情况。Java 的安全性同样也继承给了 Servlet。因此即使是有缺陷的 servlet 也不会以象 C 语言编写的服务器扩展环境那样,以崩溃的方式摧毁服务器。

(4)资源优势。Java 作为一种编程语言已被人们所广泛接收,Internet 和 Intranet 可以说是构筑在 Java 之上的。用 Java 语言编写的 Servlet 得以方便的使用这些 Java 组件、JavaBean,这也构成了 Servlet 独具的资源优势。

(5)灵活的使用方式。Servlet 的使用方式灵活多变,它可以被本地系统所调用,也可以被远程应用所唤醒。Servlet 之间可以互相链接,一个 Servlet 的运行结果可以成为另一个 Servlet 的输入,由此一个 Servlet 能够调用另一个 Servlet,或者依次调用某几个其他 Servlet 而完成一些特定的应用。

二、Servlet 的使用方式

Servlet 比较普遍的用法有三种,一是作为 CGI 应用的替代。这是 Servlet 最为广泛的应用方式,这种 Servlet 模仿了 CGI 程序的形式,但效率远远高于 CGI 程序,用法上也要灵活的多。调用方法一般是直接通过 URL 调用,比如,http://host/servlet/myServlet. 作为 CGI 替代的 Servlet 适用于产生页面比较简单,且不经常变化的场合。第二种用法是生成动态的网页。也就是编写的网页既包括静态的 HTML 和动态的 Java 代码。通常这种网页有特殊的文件扩展名,比如.jhtml。在网页被调用时,页面中的 java 代码被编译成为 Servlet 文件,同时被直接调用,动态的从数据库中取出数据,网页中的 java 代码处便被动态的生成。Servlet 的第三种用法类似于前一种,它的功能类似于 CGI,具体用法是用<servlet>的标识嵌入到网页中去。这种形式的网页的文件名扩展是.shtml。在调用这种网页时,页面中的<servlet>标签内的 Servlet 被执行完成其功能,产生结果被动态的嵌入网页中以<servlet>标识的部分。也就是嵌入网页的 Servlet。Servlet 的应用形式,可以归纳为以下几种:

1.用户验证

在电子商务或其他 Web 应用中,Servlet 可以作为验证用户身份,提供安全保障的途径。Servlet 可以接收用户通过 HTML 页面传来的帐号数据、密码口令,从而对用户进行身份验证,注册登记等等。这些数据也可以是信用卡数据,定单数据,求购指令或查询关键字,可以成为一次网上交易的开端,也可以是一次网络检索查询的起点。

2.数据库存取

Servlet 可以通过 JDBC(Java Database Connectivity)[3]对数据库进行存取。身份验证、客户登记、信息查询等等工作都需要进行后代数据库的存取,这些操作可以使用 Applet 来完成,但是使用 Servlet 代替某些 applet 的功能可以大大减轻客户端负担,加快系统响应速度,提高系统性能。

3.文件存取

通过 Servlet 可以对网络或服务器端本地文件进行存取操作。

4.动态生成网页

可以将 Servlet 嵌入网页,动态的改造原本静态呆板的风格,每次调用这种网页时,同时执行嵌入网页的 Servlet,存取数据库数据,网页的内容是动态生成的。

5.在线协作

Servlet 可以被用于多个客户端请求之间的协调。由于其多线程机制,Servlet 可以同时打开并保持与一系列数据库的连接,能同时处理多个请求,并使这些请求彼此同步,由此可用于支持某些需要同步协作的应用,诸如在线会议这样的协作系统。

6.组合应用

一个 Servlet 能把请求发送给其他服务器或其他 Servlet。这种技术能在映象同一内容的服务器间平衡装载数据。或者,可以将一个单一的逻辑任务按照任务类型或组织边界分解开,分配给多个服务器执行。应用 Servlet 之间可以进行通信的特性,我们可以把多个执行特点功能的 Servlet 之间互相链接起来,构成一个"Servlet 链",在这个链中一个 Servlet 的输出作为另一个 Servlet 的输入,并使它们组合完成单个 Servlet 难以完成的功

能。Servlet 的粒度十分有利于简化程序的编制过程，提高软件的重用度。

三、Servlet 基本框架

1. Servlet 类库结构

Servlet API 是一系列预先定义的 Java 类[1]。为编写 Servlet，需要从 JavaSoft 公司下载的 Java Servlet 开发工具集(JSDK)目前的版本为 2.0 版，JSDK 由许多个 Servlet 的类库(packages)组成，这些包的名称以 javax 开始，其中 javaxservlet 和 javax.servlet.http 两个包中保含基本的构建 Servlet 的类和接口。JSDK 中还包括 sun.servlet 包，这个包中包含服务器运行 Servlet 所需的类。一个 Servlet 可以从继承 JSDK 中最基本的 GenericServlet 类，并且重载其中方法入手。JSDK 中的 HttpServlet 是用于处理 Web 形式的客户请求的，而现有的网络应用大多是与 Web 服务有关的。HttpServlet 是从 GenericServlet 派生而来的，其中定义了针对 Web 应用的一些常用方法。它是从 GenericServlet 派生而来。Servlet 类库的体系结构如图 1 所示。

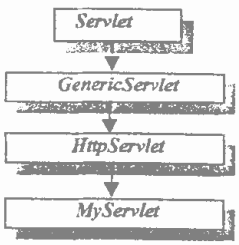

图 1

2. Servlet 基本框架

以下简要介绍 Servlet 的基本结构，一个 Servlet 是从 javax.servlet.Servlet 接口实现而来的，这个接口中定义了 5 个方法[4]。

(1)init()：init()是用来初始化 Servlet 的，比如建立数据库连接可以在 init()中实现。init()接收 ServletConfig 对象为一个参数，并初始化 Servlet。

(2)ServletConfig()用于将 ServletConfig 对象传给 init()，ServletConfig 是用于保存服务器参数的。

(3)service()：service()方法是 Servlet 的核心部分，服务器通过调用 Service 来处理请求，servic 以 ServletRequest 和 ServletResponse 的对象为参数。其中 ServletRequest 用于封装客户端的请求参数。而 ServletResponse 则是用于把处理后的消息返回客户端。

(4)destroy()：destroy()是一个 Servlet 在生命周期的最后阶段，用于 Servlet 的清除。

(5)getServletInfo()，这个方法用于返回 Servlet 的版权信息。

调用 Servlet 时，服务器只调用每个 Servlet 的一个实例，Servlet 从被第一次调用起，便驻留内存，以便迅速的处理请求，service()方法是一个同步线程(Synchronized)，也就是说可以同时被几个线程同时调用。

四、实例分析

一个典型的 Servlet 应用是从 HTML 表单的数据请求开始的。在处理请求时，Servlet 要通常执行以下三个步骤：①分析请求，比如客户端发出请求，从一个表单传来数据，Servlet 必须收集数据，根据请求的类型将数据赋值。②执行请求，比如执行数据查询的指令，连接相应的数据库，并查询数据。③生成应答，将查询所得的数据组织成为 HTML 形式的文档或其他形式的文档并返回客户端。以下为详细的介绍。

1. 分析请求

在分析请求的阶段中，Servlet 应用其中的 ServletRequest 的对象来完成操作，在 HttpServlet 中则是 HttpServletRequest.这个步骤用于分析浏览器是如何传送参数的，一般浏览器用 POST 和 GET[注 1]两种方法来传递参数，在 HttpServlet 中 service()方法用于分析请求是哪种形式的，POST 的或 GET 的？并由此将它们分别传送给 doPost()或 doGet()处理，这也是 HttpServlet 与 GenericServlet 的区别，也就是说在 HttpServlet 中你必须重载 doPost()方法或 doGet()方法，而不是象在 GenericServlet 中重载 service()方法。

调用 queryServlet 的 HTML 代码如下

```
< FORM ACTION = " http://myServer: 3072/servlet/sampleServlet" Method = "POST" >
< INPUT NAME = "query" >
< INPUT TYEP = "SUBMIT" >
</FORM>
```

ACTION 指出 Servlet 的 URL，方法是 POST.第一个 INPUT 是输入框，第二个 INPUT 是一个递交按钮。以下的例子 GenericServlet 类型的。

```
public synchronized service(ServletRequest req, ServletResponse res)
```

```
throws IOException {
//装载 DB2 的 JDBC 驱动程序
try {
    Class. forName ( " COM. ibm. db2. jdbc. app.
DB2Driver").newInstance();
    }
//抛错机制
  catch (Exception e) {
  e.printStackTrace();
}
//将参数负责赋给 queryNo
queryNo = req.getParameter("query");
//与数据库相连接
try {
  url = "jdbc:db2:sampleDatabase";
   con = DriverManager. getConnection ( url, "
userID", "password" );
  } catch( Exception e ) {
  e.printStackTrace();
}
ServletOutputStream out = res.getOutputStream();
//设置 content 的类型和其他返回文件格式
res.setContentType("text/html");
//调用数据查询方法
loadData(out);
//调用应答方法
doResponse(out);
}
```

2. **执行请求**

请求分析完毕后，Servlet 产生回应，开始执行请求，比较典型的操作可以是通过 JDBC 连接到数据库，根据参数查找数据，或者是对服务器的文件进行读写操作。

```
public void loadData( ) {
//执行数据查询并将查询结果赋值于 content
try {
  stmt = null;
  rs = null;
  stmt = con.createStatement();
  rs = stmt. executeQuery ( " SELECT text FROM
table WHERE infoNo = '" + queryNo + "'");
while(rs.next()){
    content = rs.getString("text").trim();
  }
} catch( Exception e ) {
    e.printStackTrace();
}
}
```

3. **生成应答**

在执行完请求后，Servlet 给客户端返回信息，Servlet 与客户端的通信通过 ServletResponse 的对象来完成 (httpServlet 中则是通过 httpServletResponse)．在这个过程中，如果发生错误，可以通过两种方式来进行处理，一是利用 Servlet 中的处理机制抛错，二是定义 Servlet 中的 sendError 方法，如果没有错误，Servlet 可以动态的生成一个页面文件返回客户端。返回页面时前，首先通过 ServletResponse 的对象的方法 setContentType()来确定返回文件的类型，可以是 text/html 型，image/jpeg 型，或者是 image/gif 型。

```
public void doResponse(ServletOutputStream out)
  throws ServletException, IOException {
out.println("<HTML><HEAD><TITLE>");
out.println(title);
out. println ( " </TITLE > </HEAD > < BODY
>");
out.println("<font size = 3>" + Content + "</font
><br>");
out.println("</BODY></HTML>");
out.close();
}
```

注 1：在浏览器传送参数的方法中，GET 是将参数放在标识 Servlet 的 URL 后传送，而 POST 则是以表单的形式来传递参数。一般来说 GET 是缺省的方法，并且使用超链接来调用 Servlet 时，只能使用 GET 方法。但必须注意的是，用 GET 方法只限于数据量较少的情况，而且由于 Servlet 的 URL 和参数都显示在调用它的浏览器上，故不适用于有保密要求的情况。

参考文献

[1] http://java. sun. com/marketing/collateral/servlets.html

[2] http://www.servletCentral.com/

[3] http://www. javaSoft. com/products/jdbc/index.htm

[4] http://java.sun.com/docs/books/tutorial/servlets/

(来稿时间：1999 年 1 月)