Java Servlets

CSE 486/586
Nov 02, 2004

References:
http://java.sun.com/j2ee/1.4/docs/tutorial/doc/Servlets.html
http://www.webdevelopersjournal.com/articles/intro_to_servlets.html
Vijay Arthanari's presentation on Java Servlets

What are Servlets?

- A Servlet is a Java class hosted on a web server and works on a pure request-response model.
- The requests can be of any type i.e. HTTP, SOAP, etc.
- HTTPServlet specially provided for HTTP
- Commonly used to provide dynamic content on websites. This "dynamic content" is scaled to complete web applications.

Why Servlets?

- CGI has been in use for providing dynamic content
- Servlets v/s CGI
  - Single process, multiple threads per server
  - Platform independence for server machine
  - Fault tolerance (no direct execution on shell required
  - Easier sharing of resources (database connections, etc.)

Supported by

- Intrinsically (no additional "plugins" needed)
  - Tomcat
  - Sun Java Web Server
  - Sun Java System Application Server
  - All J2EE Servers
- Also plugins are available for
  - Apache Web Server
  - Microsoft IIS Server
  - etc..

What do Servlets do? Receives Requests…

- Reads data sent by user on the click of a button or a link or sent by another java application
- The data can be present in the query string or it can be present in the HTTP message itself (possible in case of Java applications)
- Requests are represented by an HttpRequest object

What does a Servlet do? Processes it to generate response…

- Processing could involve querying database or RMI / CORBA / RPC components
- It could involve computation inside the servlet itself
- Variables required for a particular client session can be stored in the HttpSession object associated with the particular HttpRequest

What does a Servlet do? Sends out responses

- The response is formatted as per the client's need.
  - Web browser → HTML
  - SAAJ client → SOAP message
- Appropriate headers can be set for the response indicating content-type, status, etc.

Servlet Life Cycle

- init() → service() → destroy()
- init()
  - Invoked only once upon creation.
  - Used for one-time initializations
  - Servlet created when server is started or upon first request (depending on the "load-on-startup" tag in web.xml)

Servlet Life Cycle

- init() → service() → destroy()
- service()
  - On receiving each request from the client, a new thread of control is created for the Servlet and the service method is called.
  - This checks the HTTP request type (GET, POST, PUT etc.) and invokes doXXX methods based upon the request type. doGet invoked for HTTP GET request, etc.

Servlet Life Cycle

- init() → service() → destroy()
- destroy()
  - Method called when servlet instance is unloaded from memory
  - The instance is removed by either a default or specified idle time (servlet-timeout tag in web.xml).
  - It can be used to close database connections, halt background threads, commit hit counts to disk and other such cleanup activities.

Example

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class HelloWorld extends HttpServlet {
    public void doGet(HttpServletRequest request, HttpServletResponse
     response) throws IOException, ServletException{
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        String name = request.getParameter("Name");
        out.println("<html>");
        ......
        ......
        out.println("</html>");
    }

    public void doPost(HttpServletRequest request, HttpServletResponse
     response) throws IOException, ServletException{
        doGet(request, response);
    }
}
```

Useful objects and methods

- HttpRequest: Represents the request that is sent from the client
  - getParameter(): Get parameter values from the query string
  - getSession(): Get HttpSession object corresponding to this request
- HttpResponse: Represents the response sent back to the client
  - getWriter(): Get the PrintWriter object that is used to write data into this response
  - encodeURL(): Used to encode URLs by adding session IDs to them

Useful objects and methods

- Printwriter: Used to write into the content of the response object
  - print()
  - println()
- HttpSession: Used to store session-related variables. When the same user sends another request, the servlet may need to refer to some data that was created for this user on his last request
  - setAttribute(): Used to set attribute (variable) of required name and with required value
  - getAttribute(): Used to fetch attribute by the name

Useful objects and methods

- ServletContext: Similar to HttpSession, except that these variables are common across all sessions of this servlet.

  Can be used to store information such as hit counts, etc.

  Can be accessed by HttpServlet.getServletConfig().getServletContext()
  - setAttribute(): Used to set attribute (variable) of required name and with required value
  - getAttribute(): Used to fetch attribute by the name

HTTP Requests

- An HTTP request consists of a request method, a request URL, header fields, and a body.
- A request URL may be of the form:

  http://[host]:[port][request path]?[query string]
- HTTP 1.1 defines the following request methods:
  - GET: Retrieves the resource identified by the request URL
  - HEAD: Returns the headers identified by the request URL
  - POST: Sends data of unlimited length to the Web server
  - PUT: Stores a resource under the request URL
  - DELETE: Removes the resource identified by the request URL
  - OPTIONS: Returns the HTTP methods the server supports
  - TRACE: Returns the header fields sent with the TRACE request