Java Servlets

GURMEET SINGH

PG DEPTT. OF COMPUTER SCIENCE & IT

HMV JALANDHAR

Server Side Development
Web Server

- A web server is a program running on the server that listens for incoming requests and services those requests as they come in.
- Once the web server receives a request, depending on the type of request the web server might look for a web page, or it might execute a program on the server.
- It will always return some kind of results to the web browser, even if its simply an error message saying that it couldn't process the request.
- By default the role of a web server is to serve static pages using the http protocol
- Web servers can be made dynamic by adding additional processing capability to the server

Server side Extensions

Several different tools are available for extending the server capabilities

- Java enterprise architecture
- VB .Net architecture
- Active Server Pages (ASP)
- CGI-Perl scripting

These tools process incoming requests from the user and generate custom html pages

Tomcat Server

Tomcat is a stand alone web server and a servlet container

- It is open source and free for usage

It is written in Java

- You do not have to be a Java programmer to use it
- It's web server is not as fully featured as others like Apache

Installing Tomcat

- Make sure that jdk1.4 (or higher) is installed on your machine
- Download the latest windows version of Tomcat
- Run the installer by double clicking on the download
- The installer checks if JRE and JDK are available for Tomcat
- Accept the license agreement
- Installation directory: c:\Program Files\Apache Tomcat 4.0
- On installation you get a message *Completed*

HTTP

User applications implement this protocol

- Other protocols implemented by the OS.

Different applications use different protocols

- Web Servers/Browsers use HTTP
- File Transfer Utilities use FTP
- Electronic Mail applications use SMTP
- Naming Servers use DNS

Interacts with transport layer to send messages

HTTP

Lightweight protocol for the web involving a single request & response for communication

Provides 8 methods

- Get: Used to request data from server
  (By convention get will not change data on server)
- Post: Used to post data to the server
- Head: returns just the HTTP headers for a resource.
- Put: allows you to "put" (upload) a resource (file) on to a webserver so that it be found under a specified URI.
- Delete: allows you to delete a resource (file).
- Connect:
- Options: To determine the type of requests server will handle
- Trace: Debugging

HTTP Get and Post

GET and POST allow information to be sent back to the web server from a browser

- e.g. when you click on the “submit” button of a form the data in the form is send back to the server, as "name=value" pairs.

Choosing GET as the "method" will append all of the data to the URL and it will show up in the URL bar of your browser.

- The amount of information you can send back using a GET is restricted as URLs can only be 1024 characters.

A POST sends the information through a socket back to the webserver and it won't show up in the URL bar.

- This allows a lot more information to be sent to the server
- The data sent back is not restricted to textual data and it is possible to send files and binary data such as serialized Java objects.

HTTP Headers

Contains information about client and the request

Four categories of header information

- General Information: Date, caching information, warnings etc.
- Entity Information: Body of the request or response e.g. MIME type, length etc.
- Request Information: Information about client e.g. cookies, types of acceptable responses etc.
- Response Information: Information about server e.g. cookies, authentication information etc.

General & Entity information used for both client & server

Request information included by client

Response information included by server

HTTP State Tracking

Three types of tracking methods are used:

- Cookies: Line of text with ID on the users cookie file
- URL Session Tracking: An id is appended to all the links in the website web pages.
- Hidden Form Elements: An ID is hidden in form elements which are not visible to user

Custom html page allows the state to be tracked

Codes of HTTP Status

When a server responds to a request it provides a status code

Web Container automatically handles setting of status codes

Five categories of status codes

- Informational
- Success
- Redirection
- Client error
- Server error

Common Status Codes

- 200 – Request was processed normally
- 401 – Unauthorized access
- 403 – Forbidden
- 404 – Requested resource not found on server
- 405 – Method Not allowed
- 500 – Internal server error

Servlets in Java

Classes that dynamically process requests and construct responses

- Dynamically generate html pages in response to requests
- May also send data in other forms like XML or serialized Java objects
- Run in a servlet container and have access to services that the container provides

In an application processing of each request will normally be done by a different servlet.

- e.g. search catalog, check out, confirm order etc.

Client of the servlet can be any of the following

- Browser
- Applet
- Java Application

Servlet Communication

Servlet can communicate with four different entities

- Client during request/response cycle
- With servlet container to get context/config information
- With other resources on server e.g. servlets, EJBs
- With external resources like databases, legacy systems, and EIS

Client communication can be in many forms

In Http communication

- Request – Information parameters (as name value pairs)
- Response
  - HTML (Browsers)
  - WML (Mobile Devices)
  - CSV (Spreadsheets)
  - XML (Communicating with non-java systems)
  - Serialized Objects

Servlet API

Contained in two packages

- javax.servlet
- javax.servlet.Http

Contains 20 interfaces and 16 classes

- Prevalence of interfaces allows servlet implementation to be customized to container

Java Servlets

Javax.servlet package can be extended for use with any application layer protocol

- http is the most popularly used protocol
- Javax.servlet.http package is extension of the javax.servlet package for http protocol

The Servlet spec allows you to implement separate Java methods implementing each HTTP method in your subclass of HttpServlet.

- Override the doGet() and/or doPost() method to provide normal servlet functionality.
- Override doPut() or doDelete() if you want to implement these methods.
- There's no need to override doOptions() or doTrace().
- The superclass handles the HEAD method all on its own.

Servlet Package

Provides the contract between the servlet/web application and the web container

Used for creating protocol independent server applications

Servlet interface defines the core of the entire package

- Other interfaces provide additional services to the developer

Contains 12 interfaces

- 7 interfaces implemented by the package
- 5 interfaces implemented by the user

Servlet Interfaces

Server implemented interfaces

- ServletConfig
- ServletContext
- ServletRequest
- ServletResponse
- RequestDispatcher
- FilterChain
- FilterConfig

User implemented interfaces

- Servlet
- ServletContextListener
- ServletContextAttributeListener
- SingleThreadModel
- Filter

Servlet Classes

Servlet Classes

- GenericServlet
- ServletContextEvent
- ServletContextAttriubuteEvent
- ServletInputStream
- ServletOutputStream
- ServletRequestWrapper
- ServletResponseWrapper

Exception Classes

- ServletException
- UnavailableException

Generic Servlets

GenericServlet is abstract class that implements servlet interface

- Requires implementing the service() method
- Servlets normally extend from this class

Methods

- LifeCycle Methods
  - init()
  - service()
  - destroy()
- Environment Methods
  - getServletContext()
  - getInitParameter(…)
  - getInitParameterNames()
- Utility Methods
  - log(…)

javax.servlet.http

Javax.servlet package provides interfaces and classes to service client requests in protocol independent manner.

- Javax.servlet.http package supports http-specific functions.

Several of the classes are derived from the javax.servlet packaage

Some methods from the javax.servlet package are also used

Contains

- 8 interfaces
- 7 classes

HTTP Servlet Class

Extends the Generic Servlet

- Inherits the init() and destroy methods()
- Overrides the service() method

Service() method

- Signature: Protected void service(HttpServletRequest req, HttpServletResponse res)
- Forwards the request to the appropriate method
- Developer should not normally override this method

The developer needs to implement the methods corresponding to the request

- doGet(), doPost(), doHead(), doPut()

HTTP Request Interface

Extends the Generic Servlet

- Inherits the init() and destroy methods()
- Overrides the service() method

Service() method

- Signature: Protected void service(HttpServletRequest req, HttpServletResponse res)
- Forwards the request to the appropriate method
- Developer should not normally override this method

The developer needs to implement the methods corresponding to the request

- doGet(), doPost(), doHead(), doPut()

HTTP Request Interface

Extends the Generic Servlet

- Inherits the init() and destroy methods()
- Overrides the service() method

Service() method

- Signature: Protected void service(HttpServletRequest req, HttpServletResponse res)
- Forwards the request to the appropriate method
- Developer should not normally override this method

The developer needs to implement the methods corresponding to the request

- doGet(), doPost(), doHead(), doPut()

Cookie

Constructor

- Cookie (String name, String value)

Methods

- public void setMaxAge(int expiry)
- public void setValue(String newValue)

Can be added to the response by using

- void addCookie(Cookie cookie) of HttpServletResponse

Can be obtained from the request by using

- Cookie[] getCookies() method of the HttpServletRequest

How to write a servlet

Create a servletclass

- extend HttpServlet

Implement the doGet() or doPost() method

- Both methods accept two parameters
  - HttpServletRequest
  - HttpServletResponse
- Obtain parameters from HttpServletRequest Interface using
  - getParameter(String name)
- Obtain the writer from the response object
- Process input data and generate output (in html form) and write to the writer
- Close the writer

Thanks