JAVA Servlets

Kanika Lakhani
Temporary Assistant Professor
Dept. of Computer Science
Faculty Of Technology & Engg.
Maharaja Sayajirao University
kanika.lakhani-cse@msubaroda.ac.in

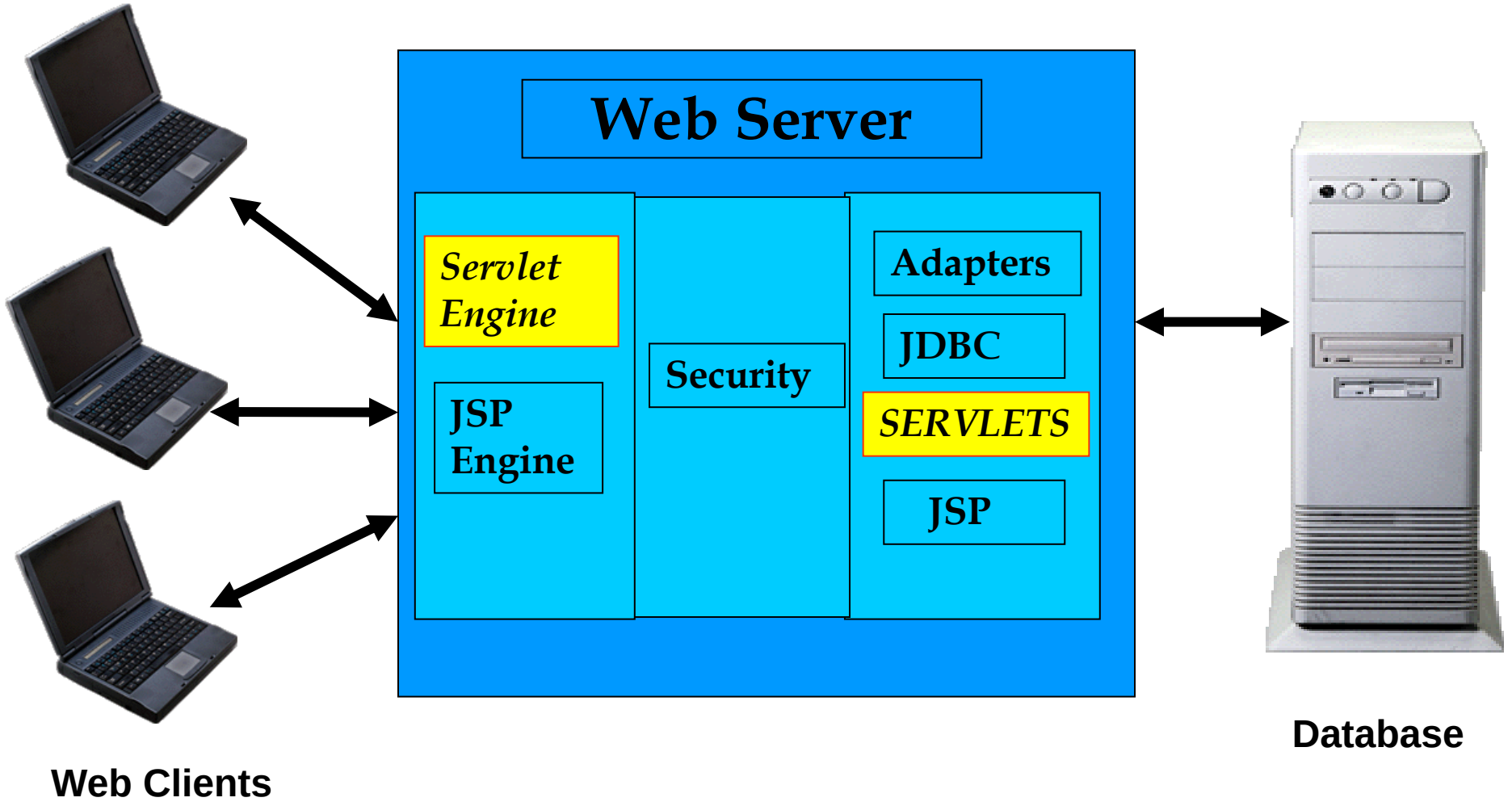
Java Servlet

- Using Java there exists a way to generate dynamic web pages and that way is Java Servlet.
- Servlets are the Java programs that runs on the Java-enabled web server or application server.
- They are used to handle the request obtained from the web server, process the request, produce the response, then send response back to the web server.

What is Servlet?

- Servlet is a program which is useful for processing of request sent by different web applications and responding to them based on the request generated
- They reside in Web Server and useful in building Web pages that are accessed in Enterprise applications
- They are used for Resource sharing, load balancing and fault tolerance in the application Servers
- They are used to develop web pages that are dynamic, based on user input, robust and handle data connectivity effectively
- They resembles Applets in many aspects regarding the structure and life cycle

Servlet Block Diagram



Features of Servlets

Feature	Description
Portability	As Servlets are developed in Java environment they can be easily run in any type of system platform or architecture without any structural modification
Performance	Servlets are initialized once and keep on running for multiple requests by creating new threads for transactions without shutting down
Extensibility	Servlets use all object oriented features of Java program so that they can be easily extended to any classes written in Java code

Properties of Servlets

- Servlets work on the server-side.

- Servlets are capable of handling complex requests obtained from web server.

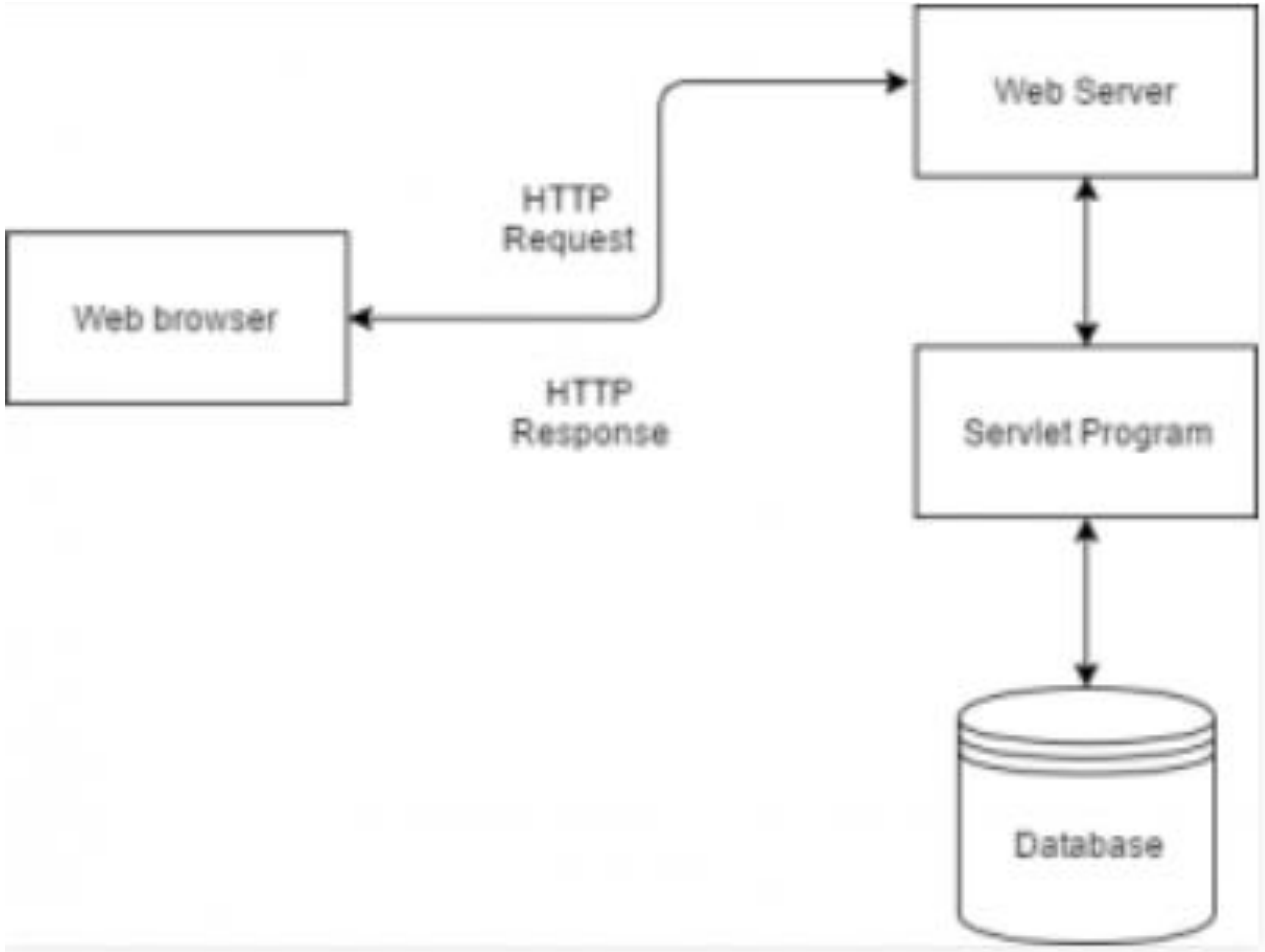
Applets	Servlets
A Java applet is a small application which is written in Java and delivered to users in the form of bytecode.	A servlet is a Java programming language class used to extend the capabilities of a server.
Applets are executed on client side.	Servlets are executed on server side.
Applets are used to provide interactive features to web applications that cannot be provided by HTML alone like capture mouse input etc.	Servlets are the Java counterpart to other dynamic Web content technologies such as PHP and ASP.NET.
Life cycle of Applets init(), stop(), paint(), start(), destroy().	Lifecycle of servlets are:- init( ), service( ), and destroy( ).
Packages available in Applets are :- import java.applet.*; and import java.awt.*.	Packages available in servlets are:- import javax.servlet.*; and import java.servlet.http.*;
Applets use user interface classes like AWT and Swing.	No User interface required.
Applets are more prone to risk as it is on the client machine.	Servlets are under the server security.
Applets utilize more network bandwidth as it executes on the client machine.	Servlets are executed on the servers and hence require less bandwidth.
Requires java compatible browser for execution.	It accepts input from browser and generates response in the form of HTML Page, Javascript Object, Applets etc.

Execution of Servlets

Execution of Servlets involves six basic steps:

1. The clients send the request to the web server.
2. The web server receives the request.
3. The web server passes the request to the corresponding servlet.
4. The servlet processes the request and generates the response in the form of output.
5. The servlet sends the response back to the web server.
6. The web server sends the response back to the client and the client browser displays it on the screen.

Servlet Architecture



Need for server side Extensions

- The **server-side extensions** are the technologies that are used to create dynamic Web pages.
- To provide the facility of dynamic Web pages, Web pages need a container or Web server.
- To meet this requirement, independent Web server providers offer some proprietary solutions in the form of **APIs**(Application Programming Interface).
- These **APIs** allow us to build programs that can run with a Web server.
- **Java Servlet** is also one of the component APIs of **Java Platform Enterprise Edition** which sets standards for creating dynamic Web applications in Java.

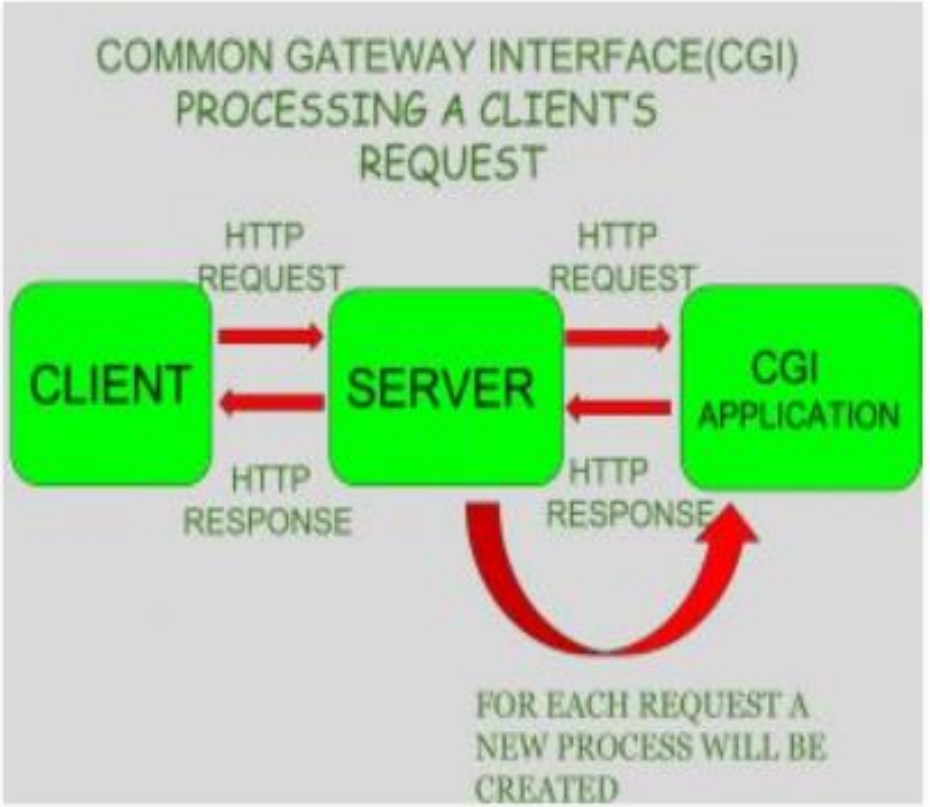
- The Servlet technology is similar to other Web server extensions such as **Common Gateway Interface**(CGI) scripts and **Hypertext Preprocessor** (PHP).

- However, Java Servlets are more acceptable since they solve the limitations of **CGI** such as low performance and low degree scalability.

What is CGI ?

- **CGI** is actually an external application which is written by using any of the programming languages like**C** or **C++**

- This is responsible for processing client requests and generating dynamic content.



In CGI application, when a client makes a request to access dynamic Web pages, the Web server performs the following operations :

- It first locates the requested web page *i.e* the required CGI application using URL.
- It then creates a new process to service the client’s request.
- Invokes the CGI application within the process and passes the request information to the server.
- Collects the response from CGI application.
- Destroys the process, prepares the HTTP response and sends it to the client.

Difference between Servlet and CGI

Servlet	CGI(Common Gateway Interface)
Servlets are portable and efficient.	CGI is not portable
In Servlets, sharing of data is possible.	In CGI, sharing of data is not possible.
Servlets can directly communicate with the web server.	CGI cannot directly communicate with the web server.
Servlets are less expensive than CGI.	CGI are more expensive than Servlets.
Servlets can handle the cookies.	CGI cannot handle the cookies.

Servlets API’s

- javax.servlet(Basic)
- javax.servlet.http(Advance)

Component	Type	Package
Servlet	Interface	javax.servlet.*
ServletRequest	Interface	javax.servlet.*
ServletResponse	Interface	javax.servlet.*
GenericServlet	Class	javax.servlet.*
HttpServlet	Class	javax.servlet.http.*
HttpServletRequest	Interface	javax.servlet.http.*
HttpServletResponse	Interface	javax.servlet.http.*
Filter	Interface	javax.servlet.*
ServletConfig	Interface	javax.servlet.*

Advantages of a Java Servlet

- Servlet is **faster** than CGI as it doesn’t involve the creation of a new process for every new request received.
- Servlets as written in Java are **platform independent**.
- Removes the overhead of creating a **new process** for each request as Servlet doesn’t run in a separate process. There is only a single instance which handles all requests concurrently. This also saves the memory and allows a Servlet to easily manage client state.
- It is a server-side component, so Servlet inherits the **security** provided by the Web server.
- The **API** designed for Java Servlet automatically acquires the advantages of Java platform such as platform independent and portability.
- It can use the wide range of APIs created on Java platform such as **JDBC** to access the database.

The Servlet Container

- **Servlet container**, also known as **Servlet engine** is an integrated set of objects that provide run time environment for Java Servlet components.

- In simple words, it is a system that manages Java Servlet components on top of the Web server to handle the Web client requests.

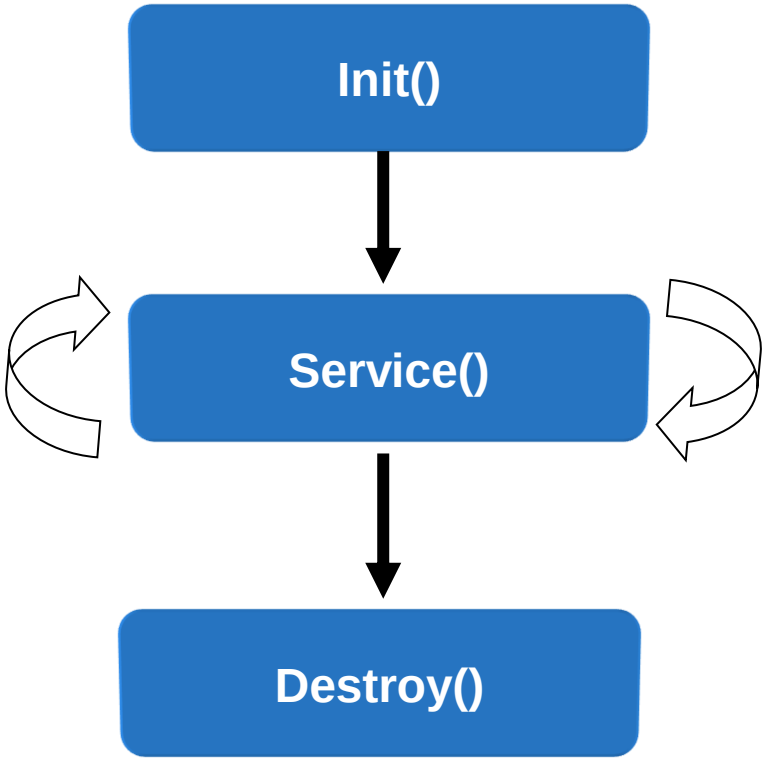
Services provided by the Servlet container

- **Network Services :** Loads a Servlet class. The loading may be from a local file system, a remote file system or other network services. The Servlet container provides the network services over which the request and response are sent.
- **Decode and Encode MIME based messages :** Provides the service of decoding and encoding MIME-based messages.
- **Manage Servlet container :** Manages the lifecycle of a Servlet.
- **Resource management :** Manages the static and dynamic resources, such as HTML files, Servlets and JSP pages.
- **Security Service :** Handles authorization and authentication of resource access.
- **Session Management :** Maintains a session by appending a **session ID** to the URL path.

Servlet API

- The Servlet API is provided in two packages
  - Javax.servlet - Javax.servlet.http
- **javax.servlet** is the base Servlet API
  - It contains interface Servlet and its generic implementation, GenericServlet
  - This package also contains two interfaces ServletRequest and ServletResponse
- **javax.servlet.http** package contains the classes that are used to develop HTTP specific Servlets
  - HttpServlet is extended from the GenericServlet base class and it also implements the Servlet interface
  - HttpServlet provides the template or framework for handling all HTTP request provided by the clients

Servlet Life Cycle



- There are three phases in Servlet Life cycle they are
  - **Initialization**
  - **Handling Client requests**
  - **Cleanup process**
- ***Javax.servlet.Servlet*** interface defines five methods that describe the life cycle of an Servlet
  - **Init()**
  - **Service()**
  - **getServletInfo()**
  - **getServletConfig()**
  - **Destroy()**

Servlet Life Cycle (Contd.)

Method Name	Description
`Init()`	It is called once when the Servlet is loaded. It takes ServletConfig as parameter that initializes the Servlet
`Service()`	It handles requests passed by clients in separate threads. It has ServletRequest and ServletResponse objects as parameters
`getServletInfo()`	It is useful to get Servlet information as it returns version, copyrights and other Servlet related information
`getServletConfig()`	It returns the ServletConfig object passed as parameter to the init() method
`Destroy()`	It is called when Servlet is unloaded. It is useful for cleanup of recourses like Data connections, temporary files etc

Key Points

- Servlet is a program which is useful for processing of request sent by different web applications and responding to them based on the request generated
- Features of Servlets are Portability, Performance, Extensibility and Security
- javax.servlet is the base Servlet API
- javax.servlet.http package contains the classes that are used to develop HTTP specific Servlets
- There are three phases in Servlet Life cycle they are
  - Initialization
  - Handling Client requests
  - Cleanup process

Types of Servlets

- There are two types of Servlets
  - Generic Servlets
    - This type of Servlets are derived from the ***javax.servlet.GenericServlet*** class that implements ***javax.servlet.Servlet interface*** and they perform the ordinary Servlet request and response life cycle of Servlet
  - HTTP Servlets
    - This type of Servlets extends the javax.servlet.HttpServlet class that was derived from javax.servlet.GenericServlet class

Working of Generic Servlets

- Whenever a server encounters a request, it invokes Servlet and passes it to the Request and Response objects for getting result
- When the Servlet container first loads a Servlet it invokes the Servlets init() method to initialize the Servlet
- Then, as requests are made to execute the Servlet, the Servlet container repeatedly invokes the Servlets service() method to provide the required service
- Finally, when the Servlet container no longer needs the Servlet, it invokes the Servlets destroy() method and unloads it from memory
- **During the lifetime of a single Servlet instance, the init() and destroy() methods will be invoked only once, whereas the service() method will be invoked many times**

Working of HTTP Servlets

- This class overrides the service() method to handle different types of the HTTP requests with the corresponding **doXXX()** methods

- The **doXXX()** methods that override service() method are
  - doGet()
  - doPost()
  - doPut()
  - doTrace()
  - doDelete()

- For most of the applications doGet() and doPost() are used because they are meant to handle request from HTML format

Security in Servlets

- Servlets use Security concepts of **Java Security Manager** for preventing un-trusted code from performing restricted actions
- Java Security manager uses the Java security policy to define restriction on the code and specifies the permissions to be given to the classes
- Web Logic Server provides a sample Java security policy file, it is located at **WL_HOME\server\lib\ weblogic.policy**
- To use the Java Security Manager security policy file with Web Logic Server deployment, the location of the web logic policy file has to be specified
- In web logic server the default position for the policy file to be stored is **file:/weblogic/application/defaults/Web**

Key Points

- **Generic Servlets** is type of Servlets are derived from the javax.servlet.GenericServlet class that implements javax.servlet.Servlet interface and they perform the ordinary Servlet request and response life cycle of Servlet
- **HTTP Servlets** is type of Servlets extends the javax.servlet.HttpServlet class that was derived from javax.servlet.GenericServlet class
- For most of the applications **doGet() and doPost()** are used because they are meant to handle request from HTML format
- Servlets use Security concepts of Java Security Manager for preventing un-trusted code from performing restricted actions

doGet()-doPost() Methods

- The doPost() and doGet() methods handle requests from the appropriate HTML form and execute business logic
- An HTML form has an attribute called "METHOD" that defines how the data will be sent to the server
- The GET method appends the data to the URL and sends it to the server that way
- The POST method bundles the data in a packet and sends the packet of data to the server

doGet()-doPost() Methods

syntax of doPost(...)

```
void doPost(HttpServletRequest request,
HttpServletResponse response) throws
ServletException, IOException
```

syntax of doGet(...)

```
void doGet(HttpServletRequest request,
HttpServletResponse response) throws
ServletException, IOException
```

Example - doPost() method

Servlet to accept a name from the HTML interface and on clicking submit button display message “welcome to Servlets” along with name

Client API using HTML

```
<HTML>
<HEAD><TITLE> INTERFACE </TITLE></HEAD>
<BODY>
<FORM METHOD=POST ACTION="<HOST NAME>">
ENTER YOUR NAME <INPUT TYPE=TEXT NAME="YNAME"><BR>
<INPUT TYPE=SUBMIT VALUE="DISPLAY">
</FORM>
</BODY>
</HTML>
```

Servlet Class

Example - doPost() method

```
import javax.servlet.*;
import java.io.*;
import java.util.*;
import javax.servlet.http.*;
public class cntservlet extends HttpServlet
{
public void doPost(HttpServletRequest
  request,HttpServletResponse response) throws
  ServletException,IOException
{
response.setContentType("text/html");
String name=request.getParameter("yname");
PrintWriter out=response.getWriter();
out.println("<html><head><title>page2</title></head>");
Out.println("<body");
out.println("Welcome to Servlets"+name);
out.println("</body></html>");}}
```

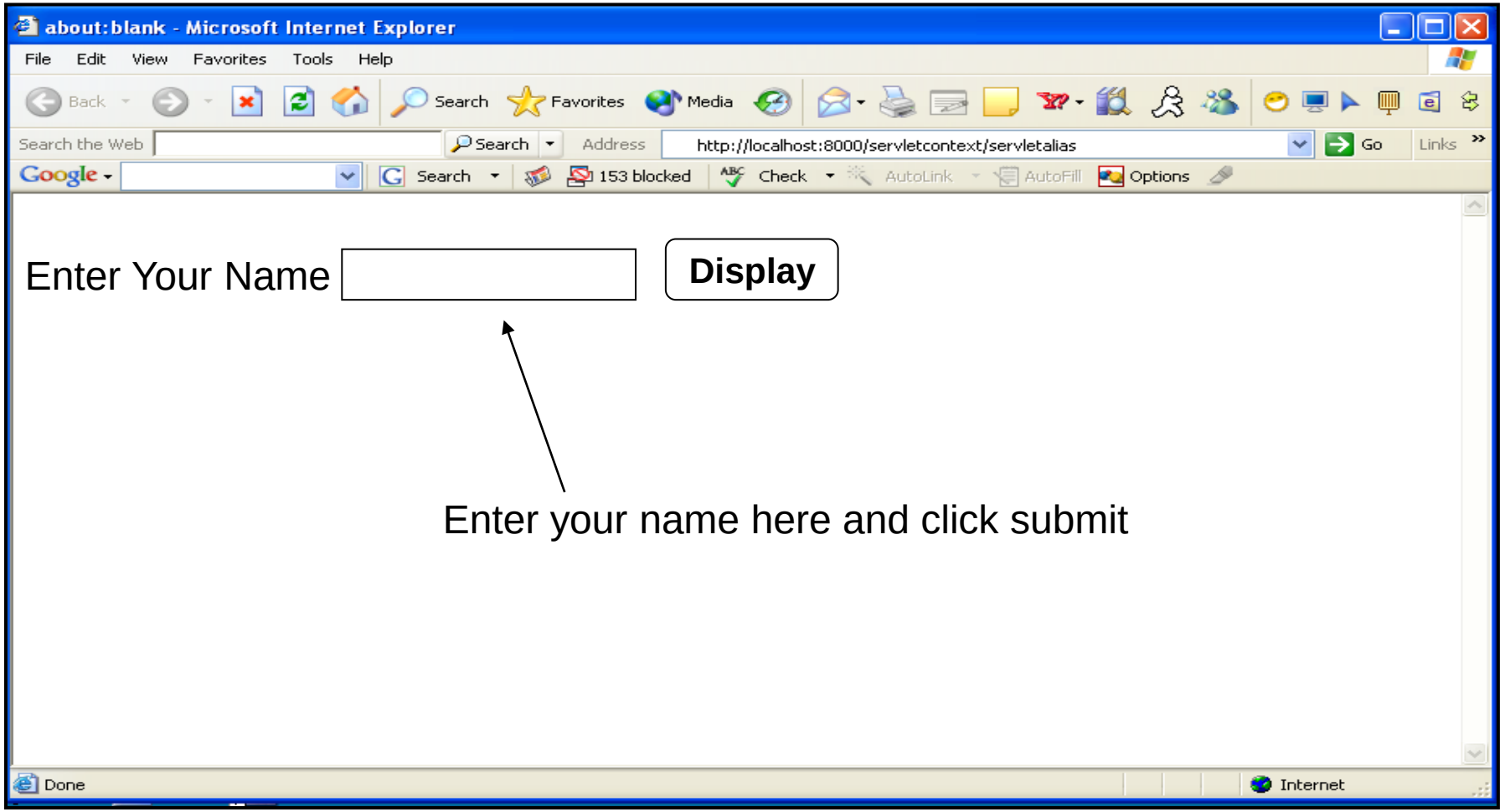
set response headers *before* actually returning any of the content

send output back to the client

the character encoding which was specified in the response message

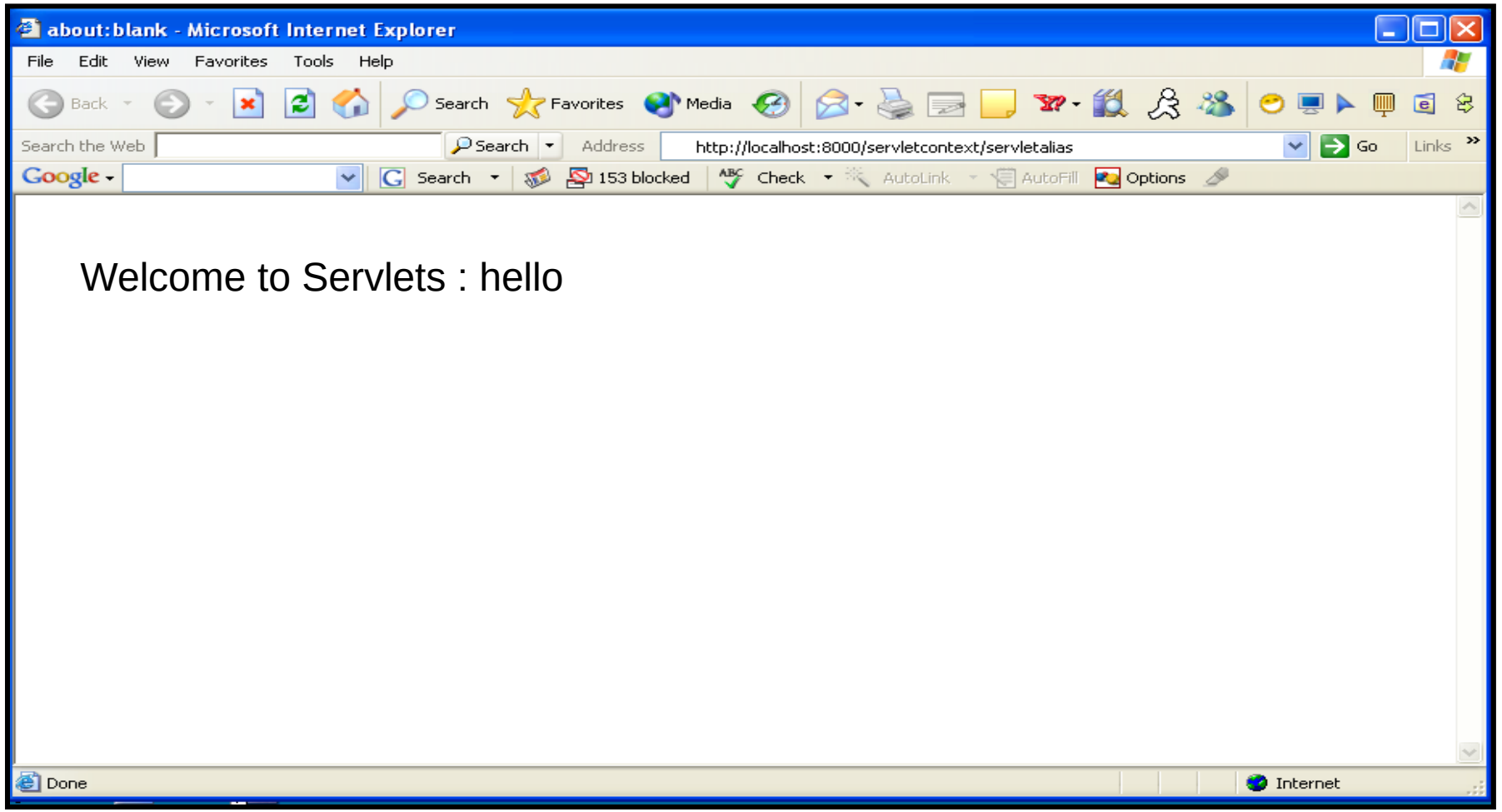
Executing Servlets

Displays the HTML API



Executing Servlets

Output :



Example on doGet() method

Create a Servlet to accept two numbers from HTML interface and display its sum.

Client API using HTML

```
<HTML>
<HEAD><TITLE> INTERFACE </TITLE></HEAD>
<BODY>
<FORM METHOD=GET ACTION="<HOST NAME>">
ENTER FIRST NUMBER <INPUT TYPE=TEXT NAME="FNUMBER"><BR>
ENTER SECOND NUMBER <INPUT TYPE=TEXT NAME="SNUMBER"><BR>
<INPUT TYPE=SUBMIT VALUE="SUM">
</FORM>
</BODY>
</HTML>
```

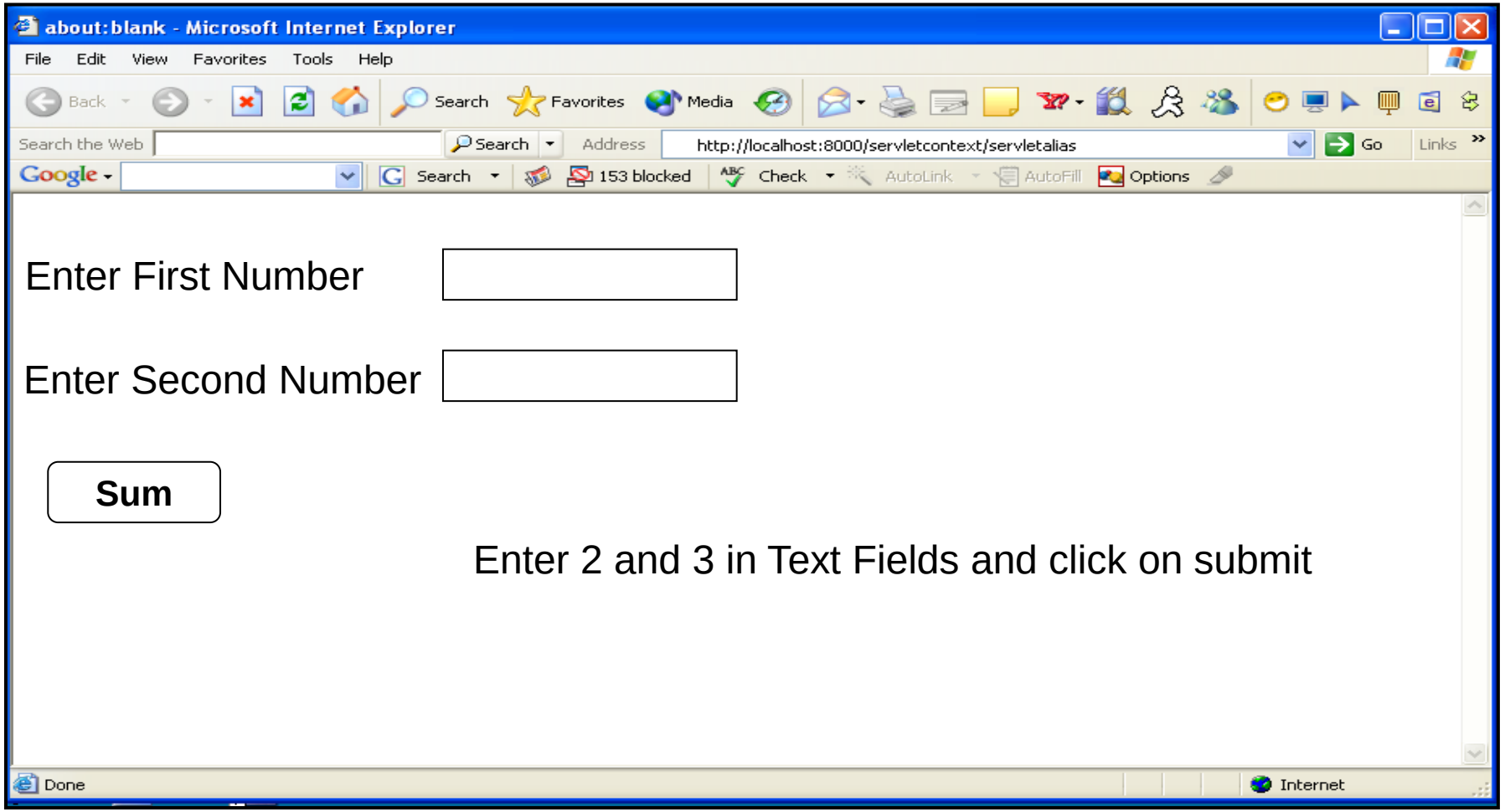
Servlet Class

Example on doGet() method

```
import javax.servlet.*;
import java.io.*;
import java.util.*;
import javax.servlet.http.*;
public class cntservlet extends HttpServlet
{
public void doGet(HttpServletRequest
request,HttpServletResponse response) throws
ServletException,IOException
{
response.setContentType(“text/html”);
int no1=Integer.parseInt(request.getParameter(“fnumber”));
int no2=Integer.parseInt(request.getParameter(“snumber”));
PrintWriter out=response.getWriter();
out.println(“<html><head><title>page2</title></head>”);
Out.println(“<body”);
out.println(“The sum is :”+(no1+no2));
out.println(“</body></html>”);}}
```

Executing Servlets

Displays the HTML API



Executing Servlets

Output :