ICON OF JAVA

ADV. JAVA

Servlets

12.Servlet Listeners

Mr. Nagoor Babu M.Tech

[ICON of JAVA]

(Sun certified & Realtime Expert)

Ex. HCL Employee

Trained Lakhs of Students
for last 14 years across INDIA

Servlet Listeners

In GUI applications, when we click on a button, when we select an item in checkboxes, when we select an item in the list, choice boxes automatically the respective GUI components may raise the respective events.

In GUI applications, all the GUI components are capable of raising the events only, they are not capable of handling the events.

In the above context, to handle the events all the GUI components will bypass the generated events to a separate implicit component called as **Listener.**

The main role of Listener in GUI applications is to listen an event from the respective GUI component, handle it by executing Listener methods and send back the response to GUI application.

The above process in GUI applications is treated as **Event Handling** or **Event Delegation Model.**

In GUI applications, all the events are represented by some predefined classes and all the Listeners are represented in the form of interfaces.

Similarly in web application execution, container may generate events at the time of creating request object, adding an attribute, replacing an attribute, removing an attribute and destroying request object.

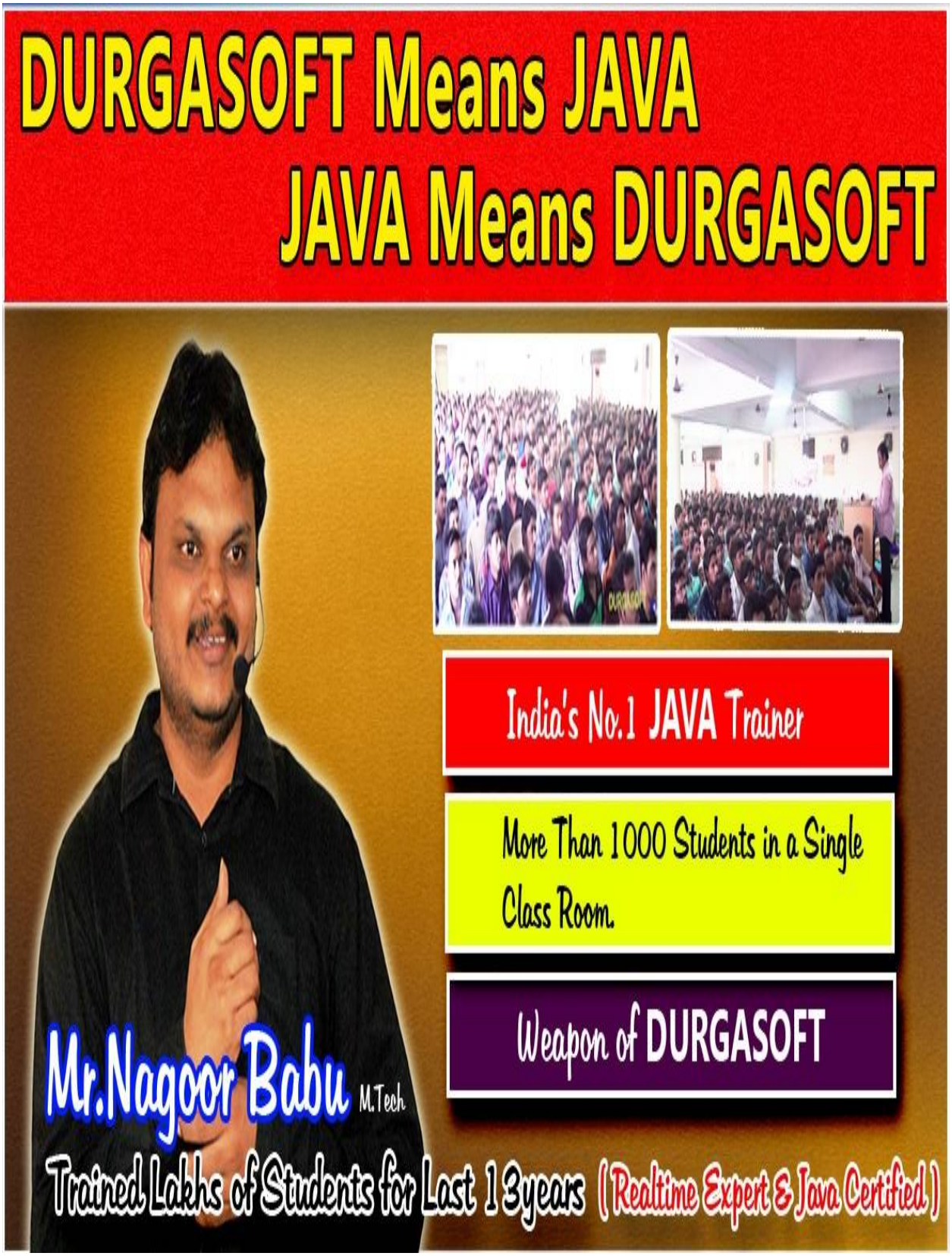
In this context, to listen the events generated by the container and to handle that events Servlet API has provided a set of interfaces called as Servlet Listeners.

Therefore, the main purpose of **Servlet Listeners** is to read all the life cycle stages of request object, ServletContext object and HttpSession objects.

In web applications, to perform event handling Servlet API has provided the following predefined library.

Where ServletRequestListener, ServletContextListener and HttpSessionListener can be used to read the life cycle stages like creation, destruction of ServletRequest object, ServletContext object and HttpSession object.

Where ServletRequestAttributeListener, ServletContextAttributeListener and HttpSessionAttributeListener can be used to read the life cycle stages of request object, context object and session object like adding a attribute, replacing an attribute and removing an attribute.


DURGASOFT Means JAVA
JAVA Means DURGASOFT
India's No.1 JAVA Trainer
More Than 1000 Students in a Single Class Room.
Weapon of DURGASOFT
Mr.Nagoor Babu M.Tech.
Trained Lakhs of Students for Last 13years (Realtime Expert & Java Certified)

Where HttpSessionBindingListener can be used to read the life cycle stages of HttpSession object like adding HttpSessionBindingListener implementation class object reference to HttpSession object and eliminating HttpSessionBindingListener implementation class object reference from HttpSession object.

Where HttpSessionActivationListener can be used to read the life cycle stages of HttpSession object like participating HttpSession object in Serialization process of Marshalling and participating HttpSession object in Deserialization process of Unmarshalling in RMI applications i.e. Distributed applications.

If we want to use Listeners in our web applications then we have to use the following steps.

Step 1: Prepare Listener class.

Here take one user defined class, it must be an implementation class to Listener interface.

Ex: public class MyListener implements Listener {-----}

Step 2: Configure Listener classin web.xml file**.**

To configure Listener class in web.xml file we have to use the following xml tags.

```
<web-app>

            ------------

<listener>

<listener-class>Fully Qualified name of Listener class</listener-class>

</listener>
```

```
</web-app>
```

The above Listener configuration is required for all Listeners except HttpSessionBindingListener and HttpSessionActivationListener.

listenerapp:-

web.xml:-

```
<web-app>
<display-name>listenerapp</display-name>
<listener>
<listener-class>HitCountListener</listener-class>
</listener>
<servlet>
<servlet-name>MyServlet</servlet-name>
<servlet-class>MyServlet</servlet-class>
</servlet>



<servlet-mapping>
<servlet-name>MyServlet</servlet-name>
<url-pattern>/listener</url-pattern>
</servlet-mapping>
</web-app>
```

HitCountListener.java:-

```
import javax.servlet.ServletContext;
import javax.servlet.ServletRequestEvent;
import javax.servlet.ServletRequestListener;

public class HitCountListener implements ServletRequestListener {
        int count=0;
         public void requestInitialized(ServletRequestEvent e) {
               System.out.println("Request Object Created");
        }
public void requestDestroyed(ServletRequestEvent e) {
count=count+1;
ServletContext context=e.getServletContext();
context.setAttribute("count", count);
System.out.println("Request Object Destroyed");
}

}
```

MyServlet.java:-

```
import java.io.IOException;
import java.io.PrintWriter;

import javax.servlet.ServletContext;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class MyServlet extends HttpServlet {

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		response.setContentType("text/html");
		PrintWriter out=response.getWriter();
		ServletContext context=getServletConfig().getServletContext();
		out.println("<center><h1>Hit Count is....."
+context.getAttribute("count")+"</h1></center>");
	}

}
```