***WEB DEVELOPMENT***

## >Web Development in Java

--Java is a commonly used language for web development, especially on the server-side. **Java web applications** are distributed applications that run on the internet. Web development with Java allows us to create dynamic web pages where users can interact with the interface.

### >What is a web application?

### --A web application is an application accessible from the web. A web application is composed of web components like Servlet, JSP, Filter, etc. and other elements such as HTML, CSS, and JavaScript. The web components typically execute in Web Server and respond to the HTTP request.

### > Servlet API

--The javax.servlet package comes with many interfaces like servlet, filter, filterchain, servletconfig, etc. Servlet increases the capability of servers that are used to host applications. The **web applications** developed via servlet in Java follow the request-response model. A servlet has a life cycle starting from being initialized to getting collected by the garbage collector.

# >Servlet Interface

# **--Servlet interface provides** common behavior to all the servlets. The Servlet interface defines methods that all servlets must implement.

- It provides 3 life cycle methods that are used to initialize the servlet, to service the requests, and to destroy the servlet and 2 non-life cycle methods.

>Methods of Servlet interface

--1. **public void init(ServletConfig config) =** initializes the servlet. It is the life cycle method of servlet and invoked by the web container only once.

2. **public void service(ServletRequest request, ServletResponse response) =**

Provides response for the incoming request. It is invoked at each request by the web container.

3. **public void destroy() =** is invoked only once and indicates that servlet is being destroyed.

4. **public ServletConfig getServletConfig() =** returns the object of ServletConfig.

5. **public String getServletInfo() =** returns information about servlet such as writer, copyright, version etc.

# >Servlet Life Cycle

--The web container maintains the life cycle of a servlet instance. Let's see the life cycle of the servlet:

🡪Web Container :: A web container is **the component of a web server that interacts with Java servlets**. A web container manages the life cycle of servlets; it maps a URL to a particular servlet while ensuring that the requester has relevant access-rights.

1. Servlet class is loaded.
2. Servlet instance is created.
3. init method is invoked : The web container calls the init method only once after creating the servlet instance. The init method is used to initialize the servlet. It is the life cycle method of the javax.servlet.Servlet interface.

--Syntax of the init method of the servlet interface —

**“ public** **void** init(ServletConfig config) **throws** ServletException **“**

1. service method is invoked : The web container calls the service method each time when a request for the servlet is received. If the servlet is initialized, it calls the service method. Notice that servlet is initialized only once.

--syntax of the service method of the servlet interface—

**“public** **void** service(ServletRequest request, ServletResponse response)      **throws** ServletException, IOException **”**

1. destroy method is invoked : The web container calls the destroy method before removing the servlet instance from the service. It gives the servlet an opportunity to clean up any resource for example memory, thread etc.

--syntax of the destroy method—

**“ public** **void** destroy() **“**

**>**Cookies in Servlet

-- A **cookie** is a small piece of information that is persisted between the multiple client requests.

-A cookie has a name, a single value, and optional attributes such as a comment, path and domain qualifiers, a maximum age, and a version number.

### **🡪Types of Cookie**

There are 2 types of cookies in servlets.

1. Non-persistent cookie
2. Persistent cookie

### **--Non-persistent cookie**

It is **valid for single session** only. It is removed each time when the user closes the browser.

### **--Persistent cookie**

It is **valid for multiple session**. It is not removed each time the user closes the browser. It is removed only if the user logs out or signs out.

>Servlet Filter

-- A **filter** is an object that is invoked at the preprocessing and postprocessing of a request.

It is mainly used to perform filtering tasks such as conversion, logging, compression, encryption and decryption, input validation etc.

The **servlet filter is pluggable**, i.e. its entry is defined in the web.xml file, if we remove the entry of filter from the web.xml file, filter will be removed automatically and we don't need to change the servlet.

>Servlet Listeners

-- Servlet Listener is used for listening to events in a web container, such as when you create a session, or place an attribute in a session or if you passivate and activate in another container, to subscribe to these events you can configure listener in web.xml, for example HttpSessionListener.

>JDBC API

--There are four components that make up a jdbc architecture or a jdbc application.

They are the jdbc client, the jdbc API, the jdbc driver and the driver manager. The jdbc client is the application code and the jdbc API is the standard API from Oracle.

-- The jdbc driver is a program which is an interface between our jdbc client and the underlying database and the driver manager is a helper class which finds a driver and establishes a connection to the database.

>JavaServer Pages Technology

--It gives a simple and fast way to create dynamic content. It facilitates the addition of snippets of servlet code into the text-based document. JSP contains static data expressed in text-based format namely HTML, Wireless Markup Language (WML) or XML; JSP technology elements which determine the dynamic content construction by the page.

>JavaServer Pages Standard Tag Library

--It has the iterator and conditional tags to handle flow control, tags for accessing databases with SQL, manipulating XML documents, internalization, and commonly used functions.

>JavaServer Faces Technology

--It forms the UI framework to build web applications.

>Java Message Service API

--With the combination of Java technology with enterprise messaging, the JMS API forms a powerful tool to solve enterprise computing problems.

>JavaMail API and the JavaBeans Activation Framework

--It is used to send e-mail notifications. In conjunction, one may use the JavaBeans Activation Framework (JAF) API, which determines the type of data, encapsulates the access, and discovers the operations available.

>Java API for XML Processing

--It is flexible and JAXP facilitates the use of any XML-compliant parser or the XSL processor within the application and supports the W3C schema.

>Java Naming and Directory Interface

--It provides the functionality of naming and directory, enables the applications to access several naming and directory services.

>Overall Conclusion

--Depending on the complexity of the web application, the other technologies involved are J2EE Connector Architecture; Java Authentication and Authorization Service (JAAS); Java Architecture for XML Binding (JAXB); SOAP with Attachments API (SAAJ); Java API for XML Registries (JAXR); Java Transaction API (JTA), and, etc.

- To know more about the Java technologies used in web applications, you may get in touch with the expert Java application developers at OrangeMantra. Share your web application requirements here at the leading Java web development company and get the perfect solution within the desired time frame. Timely reporting of the project will take the development work in the right direction.