**Servlets**

**Q1) What are Servlets?**

Java Servlets are programs that run on a Web or Application server and act as a middle layer between a request coming from a Web browser or other HTTP client and databases or applications on the HTTP server.

**Q2) What are the advantages of servlets over CGI?**

Servlets offer several advantages in comparison with the CGI.

1. Performance is significantly better.
2. Servlets execute within the address space of a Web server. It is not necessary to create a separate process to handle each client request.
3. Servlets are platform-independent because they are written in Java.
4. Java security manager on the server enforces a set of restrictions to protect the resources on a server machine. So servlets are trusted.
5. The full functionality of the Java class libraries is available to a servlet. It can communicate with applets, databases, or other software via the sockets and RMI mechanisms.

**Q3. What are the major tasks of servlets?**

Servlets perform the following major tasks:

1) Read the explicit data sent by the clients (browsers). This includes an HTML form on a Web page or it could also come from an applet or a custom HTTP client program.

2) Read the implicit HTTP request data sent by the clients (browsers). This includes cookies, media types and compression schemes the browser understands, and so forth.

3) Process the data and generate the results. This process may require talking to a database, executing an RMI or CORBA call, invoking a Web service, or computing the response directly.

4) Send the explicit data (i.e., the document) to the clients (browsers). This document can be sent in a variety of formats, including text (HTML or XML), binary (GIF images), Excel, etc.

5) Send the implicit HTTP response to the clients (browsers). This includes telling the browsers or other clients what type of document is being returned (e.g., HTML), setting cookies and caching parameters, and other such tasks.

**Q4. Explain servlet life cycle.**

A servlet life cycle can be defined as the entire process from its creation till the destruction. The following are the paths followed by a servlet.

1 The servlet is initialized by calling the init () method.

2 The servlet calls service() method to process a client's request.

3 The servlet is terminated by calling the destroy() method.

4 Finally, servlet is garbage collected by the garbage collector of the JVM.

**Q5. What is difference between Get and Post method?**

|  |  |
| --- | --- |
| **Get** | **Post** |
| 1) Limited amount of data can be sent because data is sent in header. | Large amount of data can be sent because data is sent in body. |
| 2) Not Secured because data is exposed in URL bar. | Secured because data is not exposed in URL bar. |
| 3) Can be bookmarked | Cannot be bookmarked |
| 4) Idempotent | Non-Idempotent |
| 5) It is more efficient and used than Post | It is less efficient and used |

**Q7. What is the purpose of RequestDispatcher Interface?**

The RequestDispacher interface provides the facility of dispatching the request to another resource it may be html, servlet or jsp. This interceptor can also be used to include the content of antoher resource.

**Q8. Difference between forward() method and sendRedirect() method ?**

|  |  |
| --- | --- |
| **forward() method** | **sendRedirect() method** |
| 1) forward() sends the same request to another resource. | 1) sendRedirect() method sends new request always because it uses the URL bar of the browser. |
| 2) forward() method works at server side. | 2) sendRedirect() method works at client side. |
| 3) forward() method works within the server only. | 3) sendRedirect() method works within and outside the server. |

**Q9. What is difference between ServletConfig and ServletContext?**

The container creates object of ServletConfig for each servlet whereas object of ServletContext is created for each web application.

**Q11. Define ‘init’ and ‘destroy’ methods in servlets.**

Servlets ​​Init Method is used to initialise a Servlet.

After the web container loads and instantiates the servlet class and before it delivers requests from clients, the web container initializes the servlet. To customize this process to allow the servlet to read persistent configuration data, initialize resources, and perform any other one-time activities, you override the init method of the Servlet interface.

When a servlet container determines that a servlet should be removed from service (for example, when a container wants to reclaim memory resources or when it is being shut down), the container calls the destroy method of the Servlet interface.

**Q12. Why do we need Servlet Filter?**

Ans: We need Servlet Filters for the following reasons:

1. Logging the request parameters to log files.
2. Authentication and Authorization of the request for the needed resources.
3. Formatting of the request body/header before sending it to the servlet.
4. Compressing response data sent to the client.
5. Change the response by adding some cookies and header information.

**Q13. Explain MVC pattern.**

Ans: Model-View-Controller (MVC) is a design pattern that divides a software application into three segments namely the Model, the View and the Controller.

A model deals with the behaviour of the application. It contains the data and business logic of the application. It notifies views and controllers when there is a change in its state.

A view renders the information to the user so that it looks attractive and appealing. It takes information from the model using which it generates output.

A controller takes input from a user and sends the command to model or view. It controls the flow of the application.

**Q14. Can you create a Deadlock condition on a servlet?**

Yes, a Deadlock situation can be created on a servlet by calling doPost() method inside doGet() method, or by calling a doGet() method inside doPost() method will successfully create a deadlock situation for a servlet.

**Q15. What is a Cookie?**

Ans: A cookie is a little bit of data passed from one client request to the next. A cookie has a name, a single value, and optional properties including comments, link and domain identifiers, a max-age, and a sequence number.

**Q16 What is the life-cycle of a servlet?**

Servlet is loaded.

Servlet is initialised

Service the requested

Servlet is destroyed

**Q17. What is Session Tracking?**

Ans: The term "session" merely refers to a certain period.

Session tracking is a method of keeping track of a user's current status. The HTTP protocol is a stateless communication mechanism. Every time a user sends a request to the server, it treats it as a new request. As a result, we must keep track of a user's current state to identify that user.

**Q18. What is the difference between ServletConfig and ServletContext?**

**Ans:** In ServletAPI, the import interfaces ServletConfig and ServletContext are both used. The main difference between ServletConfig and ServletContext is that ServletConfig has been used by just one servlet to retrieve info, whereas ServletContext is utilised by numerous objects.  
**ServletConfig vs ServletContext Comparison Chart**

|  |  |
| --- | --- |
| **ServletConfig** | **ServletContext** |
| Each servlet class has its own ServletConfig object. | The ServletContext object is used throughout the web application. |
| The ServletConfig object will be created during the servlet's initialization phase. To build the ServletConfig object for the first time, we must explicitly supply the request. | Servlet's Object Context will be built during the deployment of the web application. Even before making the initial request, the ServletContext object may be available. |
| The ServletConfig object will be available as long as a servlet is running, but it will be removed once the servlet is finished. | The ServletContext object will be available as long as a web application is running on the server, and it will be deleted whenever the application is withdrawn. |
| When only one servlet requires information shared by it, the ServletConfig object is utilised. | When an application requires information to be exchanged, the ServletContext object is used. |
| To obtain a Servletconfig object, use the getServletConfig() function. | To retrieve a ServletContext object, use the getServletContext() function. |
| The init-param> tag will appear under the servlet-class> tag in web.xml. | The context-param> tag will appear under the web-app> tag in web.xml. |

**Q19. Can servlet have a constructor?**

Yes, Servlets can have a constructor but it is not the right way to initialize the Servlet and you should use the init() method provided by the Servlet.

**Q20. When to use doGet() and when doPost()?**

It is always preferred to use the get method because the get method is faster than the post method. Use post method only id data is sensitive if data is greater than 1024 characters and if your application doesn’t need bookmarks.

**Q21. What is Session Tracking?**

In general in web applications, a container will prepare a request object similarly to represent a particular user we have to prepare a separate session. In this context, to keep track of all the session objects at the server machine we need to set explicit mechanisms called Session Tracking Mechanisms.

**Q22. What is the need for session Tracking in the web application?**

Session Tracking is used to recognize the particular user. It is a way to maintain the state (data) of the user about a series of requests from the same user (that is, requests originating from the same browser) across the same period of time. Each time the user requests to the server, the server always treats the request as the new request.

**Q23. What are Cookies?**

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.

By default, each request is considered as a new request. In cookies technique, we add cookie with response from the servlet. So cookie is stored in the cache of the browser. After that if request is sent by the user, cookie is added with request by default. Thus, we recognize the user as the old user.

There are 2 types of cookies in servlets.

Non-persistent cookie

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

Persistent cookie

It is valid for multiple session . It is not removed each time when user closes the browser. It is removed only if user logout or signout.