Servlet Interview Questions and Answers

1. What is Web Application?
   Any application which is present in server and can be accessed through opening the browser and entering the url is called as web application.

2. What is Servlet?
   Servlet is a technology used to develop dynamic responsive web application.
   Servlet is a java program where logic is written that runs on a server.

3. What are the ways of creating a servlet class?
   There are two ways of creating a servlet class
   1. Implementing the Servlet interface.
   2. Extending the GenericServlet or HttpServlet .

4. Difference between doGet() and doPost()?
   The doGet() method is used to handle HTTP GET requests, while the doPost() method is used to handle HTTP POST requests.
   GET requests have a limited amount of data that can be sent, while POST requests have a higher limit. This means that doGet() is suitable for simple data retrieval, while doPost() is better for sending large amounts of data or updating information on the server.
   In doGet(), the data is visible in the URL and can be bookmarked, shared or even cached by web browsers. In doPost(), the data is not visible in the URL and is sent in the request body, making it more secure.

5. Difference between ServletConfig and ServletContext?
   ServletConfig is specific to a single servlet, while ServletContext is shared across all the servlets in a web application.
   ServletConfig provides configuration information that is specific to the servlet, such as initialization parameters, while ServletContext provides context information that is global to the web application, such as the web application's name, version, and context path.
   ServletConfig is accessed by calling the getServletConfig() method from the Servlet object, while ServletContext is accessed by calling the getServletContext() method from the ServletConfig object.
   ServletConfig is created and initialized when the servlet is created, while ServletContext is created and initialized when the web application is started.

6. What is RequestDispatcher?
   RequestDispatcher is an interface in Java that is used to forward the control of a request or include the response of a servlet or JSP page to another servlet or JSP page within the same web application.

7. Difference between forward() and include()
   In the case of forward, the control of the request is transferred from the current servlet to the destination servlet or JSP page, and the response is sent back to the client by the destination servlet or JSP page. In the case of include, the control remains with the current servlet, and the response generated by the included servlet or JSP page is included in the current response.
   Forward is generally more efficient in terms of performance as it involves only one round trip between the server and the client, while include involves multiple round trips.

8. What is SendRedirect?
   sendRedirect() is a method in the HttpServletResponse interface of Java Servlet API that is used to redirect a client request to a different URL. When a web application needs to redirect the client to a different URL, it sends a redirect response to the client, which causes the client to send a new request to the new URL.

9. What is difference between forward() and sendRedirect()?

In the case of forward(), the control of the request is transferred from the current servlet to the destination servlet or JSP page, and the response is sent back to the client by the destination servlet or JSP page. In the case of sendRedirect(), the control is transferred to a new URL, and the client sends a new request to that URL.
In the case of forward(), the URL of the original request is changed to the URL of the destination servlet or JSP page, but the URL in the browser remains the same. In the case of sendRedirect(), the URL in the browser changes to the new URL.
In the case of forward(), there is only one round trip between the server and the client, whereas in the case of sendRedirect(), there are two round trips: the first for the redirect response, and the second for the new request to the new URL.
forward() is generally more efficient in terms of performance as it involves only one round trip between the server and the client, whereas sendRedirect() involves two round trips.
In the case of forward(), the request and response objects are forwarded to the destination servlet or JSP page within the same web application context. In the case of sendRedirect(), the new URL can be in a different web application or server altogether.

10. Difference between ServletRequest and HttpServletRequest

ServletRequest is an interface that defines the methods that a servlet can use to read the client request parameters, attributes, and headers. It is the base interface for HttpServletRequest, which extends ServletRequest and adds methods specific to HTTP requests.
HttpServletRequest provides additional methods that are specific to HTTP requests, such as methods to retrieve the HTTP method (GET, POST, etc.), the request URI, and the query parameters.
HttpServletRequest provides methods to manage sessions, such as getSession() and getSession(boolean create). These methods allow a servlet to obtain a session object that can be used to store and retrieve information associated with a particular client.

11. What is Session Tracking explain?

Session tracking is a mechanism that enables a web server to maintain the state of a user's interactions with a web application across multiple requests. There are several ways to implement session tracking in Java Servlets.
Cookies: Cookies are small text files that a server can send to a client browser, and the browser stores them on the client's device. Cookies are used to store session identifiers that are sent back to the server with each subsequent request. The server then uses the session ID to retrieve the user's session data.
URL rewriting: In URL rewriting, the server embeds the session ID in the URL itself, as a query parameter. The client then sends the session ID back to the server with each subsequent request. URL rewriting is less secure than cookies, and it can expose the session ID in the URL.
Hidden form fields: In this approach, the server embeds the session ID in a hidden form field in an HTML form, which is then submitted by the client to the server. The server then uses the session ID to retrieve the user's session data.
HttpSession: In HttpSession tracking, the session ID is stored in the session object created between the client and the server. The server uses the session ID to retrieve the user's session data. session tracking is typically used for secure transactions, such as e-commerce applications.

12. Servlet Lifecycle explain?

The lifecycle of a servlet refers to the sequence of events that occur from the time the servlet container loads the servlet to the time it unloads it. The lifecycle of a servlet can be divided into four phases: instantiation, initialization, request handling, and destruction.
Instantiation: When the servlet container receives a request for a servlet, it checks whether an instance of the servlet class already exists. If an instance does not exist, the container creates a new instance of the servlet class by calling its no-argument constructor.
Initialization: After creating the instance, the container initializes the servlet by calling its init() method.

The init() method is called only once, immediately after the servlet instance is created. The servlet can use the init() method to perform any initialization tasks, such as setting up database connections or loading configuration parameters.
Request handling: Once the servlet is initialized, it is ready to handle requests. When a request is received, the servlet container creates a new thread to handle the request and passes the request to the servlet's service() method. The service() method is responsible for handling the request, generating the response, and returning it to the container.
Destruction: When the servlet container decides to unload the servlet (e.g., because the web application is being shut down), it calls the servlet's destroy() method. The destroy() method is called only once, and it gives the servlet an opportunity to perform any cleanup tasks, such as closing database connections or releasing system resources.

13. What is JSP?

JSP stands for JavaServer Pages, and it is a technology used to create dynamic web pages using Java. JSP allows developers to embed Java code within HTML pages, making it possible to create web pages that can generate dynamic content based on user input or other external data sources.

14. Explain tags of JSP?

Scriptlet Tag
Scriptlet tags in JSP are used to include Java code directly into the JSP page. Scriptlet tags are enclosed in <% and %> delimiters and can contain any valid Java code, including variable declarations, method calls, and control structures such as if-else statements and loops.
Declaration Tag
Declaration tags in JSP are used to declare variables, methods, or classes that can be used in the JSP page. Declaration tags are enclosed in <%! and %> delimiters and can contain any valid Java code.
Expression Tag
Expression tags in JSP are used to include Java expressions directly in the JSP page. Expression tags are enclosed in <%= and %> delimiters and can contain any valid Java expression, including variables, method calls, and mathematical expressions.
Directive Tag
Directive tags in JSP are used to provide instructions to the JSP container about how to handle the JSP page. There are three types of directive tags in JSP: page directive, include directive, and taglib directive.
Page Directive:
The page directive is used to provide instructions to the JSP container about how to process the entire JSP page. It is placed at the beginning of the JSP page, before any other JSP elements, and is enclosed in <%@ and %> delimiters.
Include Directive:
The include directive is used to include a file or resource in the JSP page at the time of translation. It is enclosed in <%@ include and %> delimiters and can be used to include other JSP pages, HTML files, or Java classes.
Taglib Directive:
The taglib directive is used to define and import custom tags or tag libraries in the JSP page. It is placed at the beginning of the JSP page, before any other JSP elements, and is enclosed in <%@ and %> delimiters.

15. Advantages of JSP

JSPs provide a way to separate presentation logic from application logic, which leads to cleaner and more maintainable code.
JSPs allow for easy integration with Java code and provide a convenient way to display dynamic content.
JSPs have a familiar syntax for developers who are familiar with HTML and XML.
JSPs can be used to create reusable page templates and components, which can save development time.