QUESTIONS

||----------JAVA Theory----------||

1. What is constructor chaining, this calling & super calling statement?
2. Features of JAVA?
3. Why Java is by default encapsulated?
4. Explain the all oops principles?
5. What is Threads and it process like single & multi Threads?
6. What is single Threads?
7. What is multithreads?
8. What is Dead lock?
9. What is collision threads?
10. What is Exception Handling & Types?
11. Difference b/w runnable interface & threads class ?
12. Difference b/w Interface & Abstract class ?
13. What are the java types and class members ?
14. What is constructor & and it's types ?
15. Why we should go for parameterized constructor ?
16. Difference b/w pass-by-value & pass-by-reference ?
17. What is Methods overLoading & with an realtime Example?
18. What is Methods overOverriding & with an realtime Example?
19. What is Polymorphism and other names of it ?
20. What is encapsulation & with an realtime Example?
21. What is singleton Design pattern ?
22. What is collection API ?
23. What is different b/w Constructor & Methods?
24. Which is the Supermost class in collection API ?
25. Which is the legacy class of collections ?
26. What is Inheritance & all tpyes with an realtime Example?

||----------Programming ----------||

Numbers

--1. Prime Number with the all variation

--2. Strong Number

--3. Arm strong Number

--4. Xylem & Phloem Number

--5. Happy Number

--6. Prefect Number

--7. Number Palindrome

--8. Special 2 digit Number

--9. Disarium Number

--10. Prime Factors

String

--1. Revers the string

--2. Palindrome the string without using the equals method & for-loop

--3. Frequency of the characters in the given string

--4. Remove the duplicate in the given string

--5. Anagram & Pangram

--6. Count the vowels, consonants, Special characters & numbers

--7. Password Validation

--8. Convert the lower case to upper case & upper case to lower case

--9. i/p = 2a3b4c

o/p= aabbbcccc

--10. Sum of the number in given string i/p="KA04EA5646";

--11. Find the length of string without using inbulit method.

--12. Fetch the character in the given string without using 'charAt( )' method.

Array

--1. Bubble sort

--2. Rotation of an array

--3. Frequency of an element

--4. Palindrome of an array

--5. Merge, Insert, remove in array

--6. Remove the duplicate in array

--7. 2d array operation

--8. Basics operation on Array

Pattern

--1. Diamond-Allow, Allow square, Solid, Solid square

--2. All alphabet Pattern

--3. Butterfly Pattern

--4. X Pattern

--5. Alone with your name printing

--6. Basics Pattern

J2EE Interview Questions

JDBC

1.What is JDBC?

2.Who are Database vendors? Name any 5 database vendors?

3.What is Database driver software and who provides it?

4.Explain Class.forName()

5.Explain registerDriver() method

6.What is Connection in JDBC?

7.What is Result Set?

8.Explain few methods of Result Set.

9.What is Statement?

10.What is Create statement

11.What is Prepared statement

12.Explain difference between Statement and Prepared Statement

13.What is the difference between execute(), executeQuery() and executeUpdate()

14.What are placeholders or delimiters

Hibernate

1.What is ORM Framework?

2.List few of the ORM Frameworks?

3.What is Hibernate?

4.What is EntityManagerFactory?

5.What is Persistence in Hibernate with JPA?

6.What is EntityManager?

7.What is EntityTransactions?

8.What is Mapping in Hibernate?

9.Different Annotations used in Hibernate?

10.Explain OneToOne uni-directional mapping with example?

11.Explain OneToMany uni-directional mapping with example?

12.Explain ManyToOne uni-directional mapping with example?

13.Explain ManyToMany uni-directional mapping with example?

14.Explain Query parameters in JPQL with example?

15.Explain OneToOne bi-directional mapping with example?

16.Explain OneToMany bi-directional mapping with example?

17.Explain ManyToOne bi-directional mapping with example?

18.Explain ManyToMany bi-directional mapping with example?

19.What is difference between JoinColumn and mappedby

20.What is cascading? List all cascading types

21.Explain cascading with example

22.Explain fetch type in hibernate with example

23.List all fetch types

24.List different fetch types of different mappings

25.Expain entity lifecycle in hibernate

26.What is caching?

27.Explain hibernate caching first level and second level

28.What is persistence.xml

29.What is persistence unit

Servlet

1.What is Servlet?

2.What is Web Application?

3.What is the difference between application server and web server?

4.What is generic servlet?

5.What is HTTPServlet?

6.Difference between HttpServlet and Generic Servlet?

7.What is the difference between doGet() and doPost()?

8.What is deployment descriptor?

9.Explain Servlet Lifecycle?

10.What is Request Dispatcher?

11.What is Servlet Context?

12.How to declare and read the context parameter with an example?

13.What is servlet config? How to read the config parameters with an example?

14.What is the difference between servlet config and context

15.What is welcome file list?

16.What is PrintWriter?

17.What is web Container?

18.What is difference between forwarding and redirecting.

19.Explain forward and include with an example?

20.What are the different mechanism that can be used for session tracking?

21.What is query String?

22.What is cookie? how to create cookies? What are the drawbacks of cookies?

23.What is HttpSession?

24.How to get HttpSession object?

Spring

Spring IOC

1.What is Spring in java?

2.What is IOC Container?

3.What is DI-Dependency Injection?

4.Explain different ways of DI using xml configurations

5.What is Bean Factory?

6.What is Application Context in spring

7.What is difference between application context and bean factory?

8.What is the difference between scopes in spring-singleton and prototype

9.Explain dependency injection in ways in spring XML based configuration
a. Constructor injection
b. Setter Injection

10.What is the use of init-method and destroy method attributes of bean tag.

Spring MVC

1.Explain MVC architecture

2.What is Spring MVC

3.What is front controller

4.What is handler mapping

5.What is controller

6.what is view resolver

7.Explain the workflow of Spring MVC

8.What is Model

9.What is ModelAndView

10.What is @ModelAttribute

11.What is @Controller

12.What is @RequestMapping

13.What is difference between pathvariable and querystring

14.What is URL, URI and URN

15.What is JSTL

16.Explain few of the important JSTL tags with example.

17.Explain Spring MVC Configuration with no XML (class level configuration)

18.List HTTP method annotations supports in spring MVC @Postmapping etc..

Spring Boot

1.What is spring boot

2.What are starter files in spring boot

3.What are the advantages of spring boot

4.What is the difference between spring mvc and spring boot

6.List the annotations used in spring boot

7.Explain @RestController, @Service. @Repository, @Controller and @Componenet

8.What is Status Code

9.List few important status codes

10.Explain exception handling in spring boot

11.What is repository entity

GIT

1.What is Git?

2.What are the advantages of git?

3.List other repositories similar to git?

4.List git commands?

QUESTIONS & ANSWERS

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 Java Server 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.