JSP

9. What are the implicit objects in JSP?

JSP has following implicit objects:

1. Request
2. Response
3. Application
4. Exception
5. Page
6. Config
7. Session

10. How will you extend JSP code?

We can extend JSP code by using Tag libraries and Custom actions.

11. How will you handle runtime exceptions in JSP?

We use Errorpage attribute in JSP to catch runtime exceptions. This attribute forwards user request to the error page automatically.

12. How will you prevent multiple submits of a page that come by clicking refresh button multiple times?

We can use Post Redirect Get (PRG) pattern to solve the issue of multiple submission of same data. It works as follows:

First time when a user submits a form to server by POST or GET method, then we update the state in application database.

Then we send a redirect response to send reply to client.

Then we load a view by using GET command. There is no data is sent in this. Since this a new JSP page, it is safe from multiple submits. The code that processes the request is idempotent. So it does not do same action twice for same request.

13. How will you implement a thread safe JSP page?

We can use SingleThreadModel Interface to implement a thread safe JSP page.

We can also add <%@page isThreadSafe=”false” %> directive in JSP page to make it thread safe.

14. How will you include a static file in a JSP page?

We can use include directive of JSP to include a Static page in JSP. In this approach, we use translation phase to include a static page. We have to specify the URL of the resource to be included as file attribute in this directive.

E.g. <%@ include file="footer.html" %>

15. What are the lifecycle methods of a JSP?

A JSP has following lifecycle methods:

1. **jspInit**(): This method is invoked when the JSP is called for the first time. We can do initial setup for servicing a request in this method.

2. **_jspService**(): This method is used to serve every request of the JSP.

3. **jspDestroy**(): Once we remove a JSP from the container, we call this method. It is used for cleanup of resources like Database connections etc.

16. What are the advantages of using JSP in web architecture?

We get following advantages by using JSP in web architecture:

1. **Performance**: JSP provides very good performance due to their design of using same code to service multiple requests.

2. **Fast**: Since JSP is pre-compiled, server can serve the pages very fast.

3. **Extendable**: JSP is based on Java Servlets. This helps in extending JSP architecture with other Java technologies like JDBC, JMS, JNDI etc.

4. **Design**: It is easier to design user interface with JSP, since it is very close to HTML. UI designers can create a JSP with mock data and developers can later provide implementation of dynamic data.

17. What is the advantage of JSP over Javascript?

In JSP we can write Java code seamlessly. It allows for writing code that can interact with the rest of the application.

Javascript code is mostly executed at client side. This limits the tasks that can be done in Javascript code. We cannot connect to database server from Javascript at the client side.

18. What is the Lifecycle of JSP?

JSP has following lifecycle stages:

1. **Compilation**: When a request is made for a JSP, the corresponding JSP is converted into Servlet and compiled. If there is already a compiled form of JSP and there is not change in JSP page since last compilation, this stage does not do anything.

2. **Initialization**: In this stage, jspInit() method is called to initialize any data or code that will be later used multiple times in _jspService() method.

3. **Service**: In this stage, with each request to JSP, _jspService() method is called to service the request. This is the core logic of JSP that generates response for request.

4. **Destroy**: In this stage, JSP is removed from the container/server. Just before removal, this stage performs the cleanup of any resources held by JSP.

19. What is a JSP expression?

A JSP expression is an element of a JSP page that is used to evaluate a Java expression and convert into a String. This String is replaced into the locations wherever the expression occurs in JSP page.

E.g. <%= expression =%>

20. What are the different types of directive tags in JSP?

JSP has following directive tags:

1. **Page**: This directive is used for page related attributes. It can be put anywhere in the JSP page. But by convention we put it on the top of the page.

   E.g.
   <%@ page attribute="value" %>

2. **Taglib**: We can create custom tags in JSP and use these by taglib directive in a JSP page.

   E.g.
   <%@ taglib uri="abc.html" prefix="tag_prefix" >

3. **Include**: We use include directive to read a file and merge its content with the JSP page. This is done during compilation stage.

   <%@ include file="relative url" >

21. What is session attribute in JSP?

Session attribute in JSP is used for HTTP session mechanism. If we do not want to use HTTP session in JSP, then we set this attribute to false. If it is set to true, we can use built in session object in JSP.

22. What are the different scopes of a JSP object?

A JSP object, implicit or explicit, can have one of the following scopes:

1. **Page**: In this scope, the object is accessible from the page where it was created. Important point here is that when a user refreshes the page, the objects of this scope also get created again.

2. **Request**: In request scope, the object is accessible to the HTTP request that created this object.

3. **Session**: In this scope, the object is available throughout the same HTTP session.

4. **Application**: This is the widest scope. The object is available throughout the application in which JSP was created.

23. What is pageContext in JSP?

In JSP, pageContext is an implicit object. This is used for storing and accessing all the page scope objects of JSP.

It is an instance of the PageContext class from javax.servlet.jsp package.

24. What is the use of jsp:useBean in JSP?

We use jsp:useBean to invoke the methods of a Java Bean class. The Java Bean class has some data and setter/getters to access the data.

With this tag, container will try to locate the bean. If bean is not already loaded then it will create an instance of a bean and load it. Later this bean can be used in expressions or JSP code.

25. What is difference between include Directive and include Action of JSP?

Some of the main differences between include Directive and include Action are as follows:

1. Include directive is called at translation phase to include content in JSP. Include Action is executed during runtime of JSP.

2. It is not possible to pass parameters to include directive. Include action can accept parameters by jsp:param tag.

3. Include directive is just copying of content from another file to JSP code and then it goes through compilation. Include action will dynamically process the resource being called and then include it in the JSP page.

26. How will you use other Java files of your application in JSP code?

We can use import tag to import a Java file in JSP code. Once a file is imported, it can be used by JSP code. It is a very convenient method to use Java classes in JSP code.

For better organization of Java code, we should create a package of classes that we are planning to use in JSP code.

27. How will you use an existing class and extend it to use in the JSP?

We can use extends attribute in include tag to use an existing class and extend it in the current JSP.

E.g.

<%@ include page extends="parent_class" %>

28. Why _jspService method starts with _ symbol in JSP?

All the code that we write in a JSP goes into _jspService method during translation phase. We cannot override this method. Where as other lifecycle methods jspInit() and jspDestroy() can be overridden.

It appears that container uses _ symbol to distinguish the method that cannot be overridden by client code.

29. Why do we use tag library in JSP?

At times we want to create a UI framework with custom tags. In such a scenario, taglib is a very good feature of JSP. With taglib we can create tags that can provide custom features.

Taglib is also a nice way to communicate with UI designers who can use custom tags in the html without going into the details of how the code is implemented.

Another benefit of taglib is reusability of the code. This promotes writing code only once and using is multiple times.

30. What is the different type of tag library groups in JSTL?

JSTL stands for JavaServer Pages Standard Tag Library. In JSTL, we have a collection of JSP tags that can be used in different scenarios. There are following main groups of tags in JSTL:

1. Core tags
2. SQL tags
3. Formatting tags
4. XML tags
5. JSTL Functions

31. How will you pass information from one JSP to another JSP?

We can pass information from one JSP to another by using implicit objects. If different JSP are called in same session, we can use session object to pass information from one JSP to another.

If we want to pass information from one JSP to another JSP included in the main JSP, then we can use jsp:param to pass this information.

32. How will you call a stored procedure from JSP?

JSP allows running Java code from a .jsp file. We can call a stored procedure by using JDBC code.

We can call a CallableStatement from JSP code to invoke a stored procedure.

If we are using Spring framework, then we can use JdbcTemplate class to invoke stored procedure from a JSP.

33. Can we override _jspService() method in JSP?

No, JSP specification does not allow overriding of _jspService method in JSP. We can override other methods like jspInit() and jspDestroy().

34. What is a directive in JSP?

JSP directive is a mechanism to pass message to JSP container. JSP directive does not produce an output to the page. But it communicates with JSP container.

E.g. <%@include ..%> directive is used for telling JSP container to include the content of another file during translation of JSP.

There can be zero or more attributes in a directive to pass additional information to JSP container.

Some of the important directives in JSP are: page, include and taglib.

35. How will you implement Session tracking in JSP?

We can use different mechanisms to implement Session tracking
JSP. Some these mechanisms are as follows:

1. **Cookies**: We can use cookie to set session information and pass it to web client. In subsequent requests we can use the information in cookie to track session.

2. **Hidden Form Field**: We can send session id in a hidden field in HTML form. By using this we can track session.

3. **Session object**: We can use the built in session object to track session in JSP.

4. **URL Rewriting**: We can also add session id at the end of a URL.

   Like- www.abcserver.com?sessionid=1234

36. How do you debug code in JSP?

In simplest form we can write logger statements or System.out.println() statements to write messages to log files. When we call a JSP, the log messages get written to logs. With useful information getting logged we can easily debug the code.

Another option in debugging is to link JSP container with an IDE. Once we link IDE debugger to JSP Engine, we can use standard operations of debugging like breakpoint, step through etc.

37. How will you implement error page in JSP?

To implement an error-handling page in JSP, we first create a JSP with error page handling information. In most of the cases we gracefully handle error by giving a user-friendly message like “Sorry! There is system error. Please try again by refreshing page.”

In this error page, we show user-friendly message to user, but we also log important information like stack trace to our application log file.

We have to add parameter isErrorPage=true in page directive of this page. This tells to JSP container that this is our error page.

<%@page isErrorPage=”true” %>

Now we can use this error page in other JSP where we want to handle error. In case of an error or exception, these JSP will direct it to errorPage.

<% page errorPage=”ErrorPage.jsp” %>

38. How will you send XML data from a JSP?

In general, JSP is used to pass HTML data to web browser. If we want to send data in XML format, we can easily do it by setting contentType=”text/xml” in page directive.

E.g. <%@page contentType=”text/xml” %>

39. What happens when we request for a JSP page from web browser?

When a user calls JSP page from web browser, the request first comes to web server. Web server checks for .jsp extension of page and passes the request to JSP container like Tomcat.
The JSP container checks whether it has precompiled JSP class or not. If this is the first time this JSP is called, then JSP container will translate JSP into a servlet and compiles it.
After compiling, JSP code if loaded in memory and JSP container will call jspInit() method and _jspService() methods.
The _jspService() method will create the output that will be sent by JSP container to client browser.

40. How will you implement Auto Refresh of page in JSP?

We can use setIntHeader() method to set the refresh frequency with which we want to auto-refresh a JSP page.
We can send key “Refresh” with the time in seconds for auto refresh of the JSP page.
E.g. response.setIntHeader(“Refresh”,10)

41. What are the important status codes in HTTP?

Every HTTP request comes back with a status code from the server.
The important status codes in HTTP are as follows:

1. 200: It means the request is successful.
2. 400: It means the request was bad.
3. 401: It means request was not authorized.
4. 404: It means the resource requested was not found.
5. 503: It means the service is not available.

42. What is the meaning of Accept attribute in HTTP header?

In HTTP header, Accept attribute is used to specify the MIME types that a HTTP client or browser can handle. MIME type is the identifier for specifying the type of file/data that we are planning to pass over the internet.

43. What is the difference between Expression and Scriptlet in JSP?

We use Expression in a JSP to return a value and display it at a specific location. It is generally used for dynamically print information like- time, counter etc in a HTML code.

Scriptlet is for writing Java code in a JSP. We can define variable, methods etc in a Scriptlet. A Scriptlet can handle much more complex code and can be also reused.

44. How will you delete a Cookie in JSP?

We can use following options to delete a Cookie in JSP:

1. **setMaxAge**(): we can set the maximum age of a cookie. After this time period, Cookie will expire and will be deleted.

2. **Header**: We can also set the expiry time in header of response. Respone.setHeader(). This will also expire the cookie after specified time period.

45. How will you use a Cookie in JSP?

We can use a Cookie in JSP by performing following steps:

First we create a Cookie object. We set the name and value of the cookie to be created.

We set the expiry time of the Cookie by setting the maximum age.
We can use setMaxAge() method for this.

Finally, we can send the cookie in a HTTP Response by sending it in HTTP header. In this way cookie goes to client browser and gets stored there till the maximum age is not achieved.

Once a Cookie is set in the client browser, we can call getCookies() method to get the list of all the cookies set in Client. We iterate through the list of all the cookies and get the value of the cookie that was set in earlier request.

In this way we can use Cookie to set some information at client side and retrieve its value.

46. What is the main difference between a Session and Cookie in JSP?

A Session is always stored at the Server side. In JSP, session is a built-in object in JSP container.

A Cookie is always stored at the client side.

We can use both the methods for Session tracking. But Cookie method needs permission from user for storing cookie at the client location.

47. How will you prevent creation of session in JSP?

We can simply set the session attribute as false in page directive to prevent creation of session object.

E.g. <% @page session=”false” %>

48. What is an output comment in JSP?

We can write output in JSP in such a way that it becomes a comment in HTML code. This comment will not be visible in the web browser. But when we view page source to see HTML, we can see output comment.

An HTML comment is of following format:
<!-- comment -->
If we output comment in above format, it will be visible to client.

49. How will you prevent caching of HTML output by web browser in JSP?

We can use set the header in response object for Cache-Control to specify no caching.
Sample code is as follows:
response.setHeader("Cache-Control", "no-store");
response.setDateHeader("Expires","0");

50. How will you redirect request to another page in browser in JSP code?

We can use sendRedirect() method in JSP to redirect the request to another location or page.
In this case the request will not come back to server. It will redirect in the browser itself.
Sample code is as follows:
<% response.sendRedirect(URL); %>

51. What is the difference between sendRedirect and forward in a JSP?

Both forward and sendRedirect are mechanisms of sending a client to another page. The main difference between these two are as follows:

1. In forward, the processing takes place at server side. In case of sendRedirect() the processing takes place the client side.

2. In forward, the request is transferred to another resource within same server. In case of sendRedirect the request can be transferred to resource on some other server.

3. In forward only one request call is consumed. In case of sendRedirect two request response calls are created and consumed.

4. The forward is declared in RequestDispatcher interface.
   Where as sendRedirect is declared in HttpServletResponse object.

52. What is the use of config implicit object in JSP?

In JSP, config object is of type ServletConfig. This object is created by Servlet Container for each JSP page. It is used for setting initialization parameters for a specific JSP page.

53. What is the difference between init-param and context-param?

We can specify both init-param and context-param in web.xml file.

We use init-param to specify the parameters that are specific to a servlet or jsp. This information is confined to the scope of that JSP.

We use context-param to specify the parameters for overall application scope. This information does not change easily. It can be used by all the JSP/Servlet in that Container.

54. What is the purpose of RequestDispatcher?

We use RequestDispatcher interface to forward requests to other resources like HTML, JSP etc.

It can also be used to include the content of another page in a JSP.

It has two methods: forward and include.

We have to first get the RequestDispatcher object from the container and then we can call include or forward method on this object.

55. How can be read data from a Form in a JSP?

There is a built-in request object in a JSP that provides methods to read Form data. Some of the methods are as follows::

1. **getParameterNames():** This method returns the list of all the parameters in the Form.

2. **getParameter():** We call this method to get the value of parameter set in the Form. It returns null if the parameter is not found.

3. **getParameterValues():** If a Parameter is mentioned multiple times in a Form, we use request.getParameterValues() method to get all the values. This method returns an array of String values.

4. **getParameterMap():** This method returns the map of all the Parameters in Form.

56. What is a filter in JSP?

We can define filters in JSP to intercept requests from a client or to change response from a server.

Filter is a Java class that is defined in the deployment descriptor of web.xml of an application. The JSP container reads filter from web.xml and applies a filter as per the URL pattern associated with the filter.

JSP Engine loads all the filters in when we start the server.

57. How can you upload a large file in JSP?

To upload a file by JSP we can use <input type="file"> in the Form data being passed from HTML.

If the file is very large in size, we can set enctype=multipart/form-data.

We have to use POST method in the Form to send a file.

Once the request is received, we can implement the logic to read mulitpart data in doPost() method of JSP. There are methods in JSP framework to read large files via this method.

58. In which scenario, Container initializes multiple JSP/Servlet objects?

To initialize multiple JSP objects, we have to specify same Servlet object multiple times in web.xml.

This indicates to JSP container to initialize separate JSP/Servlet object for each element. Each of the Servlet instance will have its own ServletConfig object and parameters.