

## Interview Question

- ① Define JDBC & Explain the Architecture of JDBC
- ② Explain all the 6 standard steps of JDBC
- ③ Define API & mention the 2 forms of API
- IMP ④ Explain Abstraction, Interface & loose coupling
- ⑤ Name the different Driver classes provided by different DB servers.
- ⑥ Write a syntax to present all the 6 stand steps of JDBC
- ⑦ In how many ways, a Driver class can be loaded?
- IMP ⑧ What are costly resources, define null pointer exception? and mention where exactly the costly resources are closed.
- ⑨ Mention the helper methods which is used to create Implementation object of Statement, PreparedStatement & Callable Statement Interface.
- ⑩ Define BatchUpdate & Explain whether Batch is advantageous w.r.t Statement (I) preparedStatement (II)?
- IMP ⑪ Define JDBC Transaction & Explain why JDBC transaction is needed along with the concept of ACID rules? (Atomicity)
- ⑫ Define Stored procedure and write all the steps to execute a stored procedure.
- ⑬ Where are all the helper class methods present & in which package JDBC API is present in?
- ⑭ Once the Data's are fetched from the DB where it is stored in java?
- ⑮ Difference b/w Statement, PreparedStatement & Callable Statement?

Static → only presentation logic  
dynamic → All 3 types

Date: \_\_\_\_\_  
Page No.: \_\_\_\_\_

## Servlets & JSP

- ① Define Server? Servlet & write the differences b/w a server & a servlet?
- ② Define JEE container? and mention its functionalities.
- ③ what are the 3 different types of data?
- ④ what are the differences b/w static & dynamic Application which involves all the 3 different types of logic?
- ⑤ Explain Servlet life cycle with code?
- ⑥ what are the different types of Servlet, Explain them in brief?
- ⑦ what are the life cycle methods of a Servlet, Explain in brief?
- ⑧ Explain HttpServlet in brief? including the contents of HttpServlet request & HttpServlet Response?
- ⑨ what are the different type of HttpServletRequest?
- ⑩ Define Servlet chaining, Explain the need for Servlet chaining along with the types & one real time example for each.
- ⑪ Define JSP? along with the need?
- ⑫ what are Directives of JSP
- ⑬ what is the default scope of JSP?
- ⑭ Explain the lifecycle of JSP along with the lifecycle methods of JSP.
- ⑮ Difference b/w a Servlet & JSP?
- ⑯ Briefly Explain cookies & Sessions?

Syntax:

`Class.forName("com.mysql.jdbc.Driver");`

Static method

java.lang

Checked Exception  $\rightarrow$  ClassNotFoundException

(2)

Establish the connection with DB Server.

`java.sql.Connection con = null;`

$\rightarrow$  JDBC Drivers in the form of jar  
 $\rightarrow$  mysql Driver

`con = DriverManager.getConnection("jdbc:mysql://localhost:`

Interface  $3306? user=root \& password=donga");$

$\rightarrow$  JDBC API  $\rightarrow$  DriverManager  $\rightarrow$  get Connection(url);

$\rightarrow$  JDBC API  $\rightarrow$  static Helper Method present in  
java.sql.DriverManager

2 Static methods  $\rightarrow$  return type is Connection  
registerDriver();  $\rightarrow$  Driver Manager

get Connection();  $\rightarrow$  returns only one implementation object of  
Connection Interface

get Connection() has 3 overloaded variant's

① `get Connection(String url);`

② `get Connection(String url, properties info);`

③ `get Connection(String url, String user, String password);`

## Gramming JEE

(1) Define JDBC?

Java DataBase Connectivity is a specification given in the form of Abstraction API to achieve loose coupling b/w Java Application & DB server Vendor.

(2)

\* why do we go to achieve loose coupling?

→ Since it doesn't affect our java Application.

(2)

\* Is it mandatory to use an API? to communicate b/w two application's

→ It is not mandatory to use an API to communicate b/w two application's, but the result is tight coupling.

In

(2) what are the 6 standard steps of JDBC?

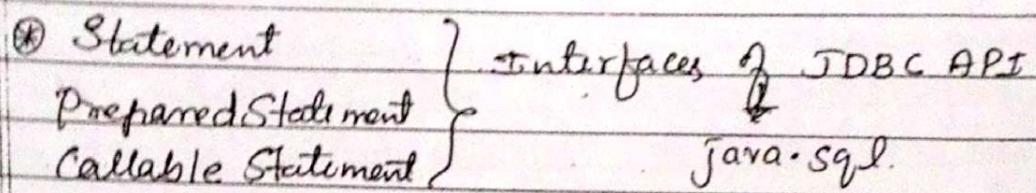
① Load & Register the Driver.

2 ways → 1) creating an object of Driver class & register with DriverManager using static method called registerDriver();  
result → tight coupling

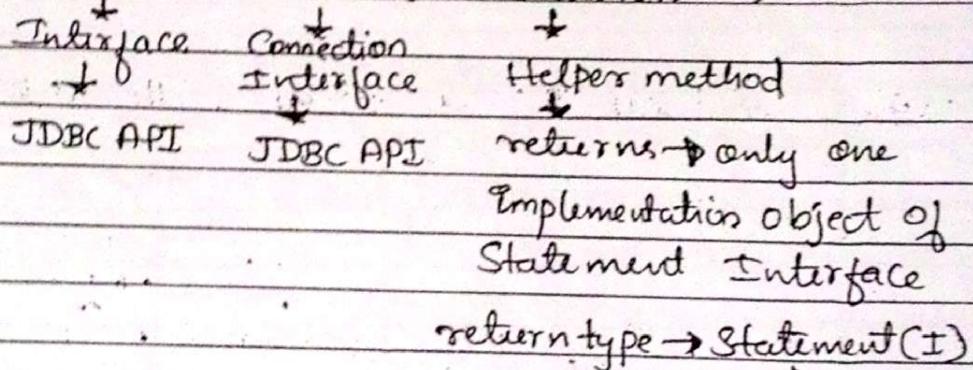
2) using static method called forName() present in java.lang

③ Create a Statement  $\leftrightarrow$  platform.

platform can be created by using



java.sql.Statement = con.createStatement();



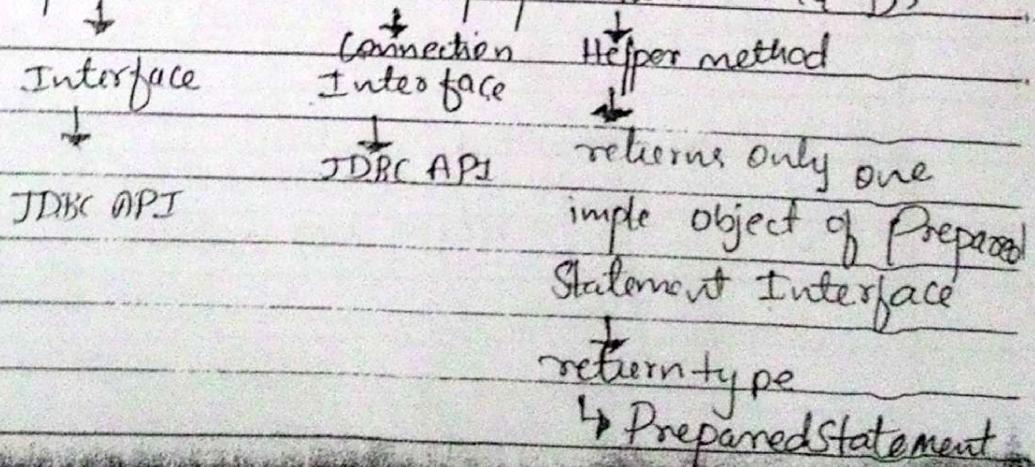
CreateStatement();  $\rightarrow$  Declared in Connection Interface

Type of method  $\rightarrow$  Non-static.

$\rightarrow$  Since interface can't have static method.

#

java.sql.PreparedStatement = con.prepareStatement(query);



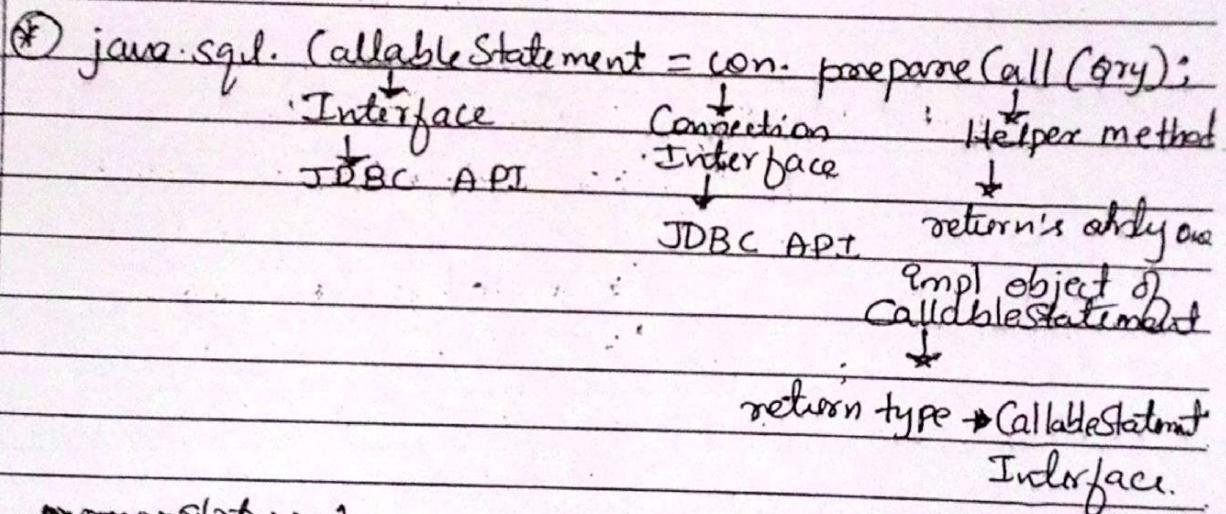
`PreparedStatement (qry);`

↳ Declared in Connection Interface

↳ Type of method → Non-static.

↳ Since Interface can't have static method.

`PreparedStatement (qry)` method takes the query at the time of implementation object creation of prepared Statement Interface.



PreparedStatement

`prepareCall (qry)` → Declared in Connection Interface.

↳ Type of method → Non-static

→ Since Interface can't have static method.

⑦ Execute SQL queries @ Statement

+ boolean `execute (SQL qry);`

↳ Generic method

↳ return type → Boolean

Can execute any kind of SQL queries.

True → DQL  
False → DML, DDL

\* \* Declared in Statement (1)

↳ Can be used to execute any kind of DML, DDL, DQL statements.

what is the outcome of DML statements?

→ 0-n Integer value

outcome of DQL → ResultSet

① + boolean execute();

+ int getUpdateCount(only DML)

↳ returns → count of total no of records affected

↳ return type → Integer

↳ throws SQLException if a DDL

② DQL qry is passed

③ + int executeUpdate(only DML)

specialized method ↳ returns → 0-n int total no of records affected

↳ return type → Integer

↳ throws SQLException if DDL ③ DQL qry is passed.

④ + ResultSet executeQuery(only DQL)

Specialized method ↳ returns → produced data stored in cursor/buffer memory

↳ return type → ResultSet

↳ throws SQLException if a DML or DDL qry is passed.

Syntax :-

stmt.execute(qry);

stmt.executeUpdate(qry);

stmt.executeQuery(qry);

} Compile + Execute

each time

→ whenever we use any methods to execute the query with Statement Interface, the query is taken in execute method's where compilation along with execution takes place each time, hence not a good practise to deal with more number of data's

w.r.t PreparedStatement.

`pstmt = con.prepareStatement (qry);`

↳ compilation takes place only once

`pstmt.executeUpdate();` } only execution  
`pstmt.executeUpdate();` } happens  
`pstmt.executeQuery();` } multiple times.

Note \*\*

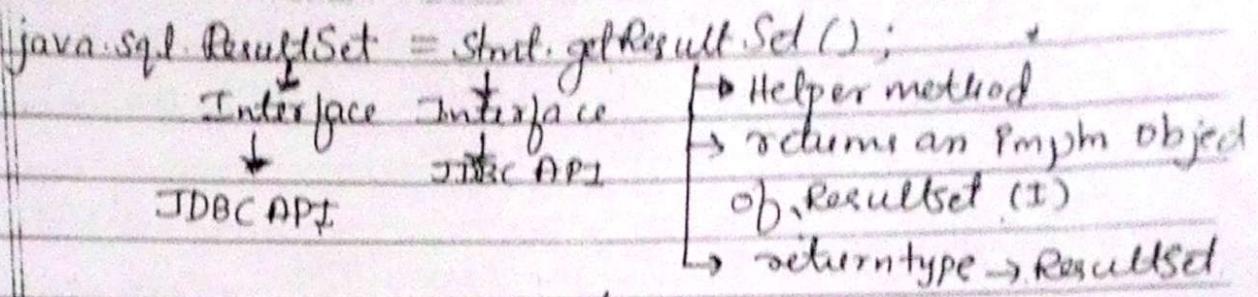
→ hence PreparedStatement is used when there are n number of data's involved.

→ CallableStatement is used in case of stored procedure.

### ⑤ Process the Resultant Data.

- \* Processed data is stored in cursor memory & buffer memory which can fetch with the help of ResultSet Interface.

(D)



Type of method → Non-static

↳ since interface can't have static method.

Declared in Stmt. Interface but can be used w.r.t Pstmt & cstmt also.

ResultSet is used to fetch the processed data from cursor / buffer memory

⑥ rs = Stmt.executeQuery();

#### ⑥ Close the Costly Resources

\* Resources which uses system properties is known as costly resources.

\* Scanner class, PrintWriter, & all the interfaces of JDBC API are costly resources.

\* costly resources are closed within finally block using if condition to avoid Null Pointer Exception

\* Costly resources are closed to improve the performance of an application but it is not mandatory.

\* Pointing to an object which is not present is known defines an exception called NullPointerException

\* Name the Driver classes along with Port numbers & fully qualified class names given by different DB Server's.

Port No

- |                               |   |      |
|-------------------------------|---|------|
| (1) MS-SQL → Sql-ServerDriver | → |      |
| (2) My-SQL → Driver           | → | 3306 |
| (3) Derby → Embedded Driver   | → | 1527 |
| (4) Oracle → OracleDriver     | → | 1521 |

#### \* Advantages of JDBC?

- ① To achieve loose coupling b/w java Application & DataBase Server.
- ② Platform Independent.

~~NOTE~~

All the Driver classes (JDBC Driver) must mandatory implements `java.sql.Driver Interface` (JDBC API).

~~Editor~~

Difference b/w Statement, PreparedStatement, CallableStatement

- \* What is JDBC Transaction? mention the need & advantages of JDBC Transaction?  
→ JDBC is considered as a single Business unit which is used execute multiple SQL statements.  
→ To maintain data consistency we go with JDBC Transaction.  
→ Advantages: - we can achieve ACID properties

- \* Define Autocommit & relate this in sql?
- Autocommit → Do everything  $\oplus$  Do nothing means autocommit.
- Do everything refers to → complete transaction  $\oplus$  successful transaction, where data consistency is maintained.
- Do nothing → refers to incomplete  $\oplus$  unsuccessful transaction where the data's are not consistently maintained.

Autocommit mode by default is set to true because of which the data's are automatically saved into the DB.

hence in case of Transaction the autocommit mode is disabled by setting it to false and once the data's are consistently maintained, we explicitly save the data's into the DB by using commit()

- \* rollback() method is used to rollback the entire operation where execution starts from the beginning again.
  - \* whenever we use rollback() with ~~savepoint~~ savempoint, the operation gets rolled back only after the savempoint is set.
  - \* Define Stored procedure ; write the steps to execute stored procedure.
- Stored Procedure is a sql subroutine which is created, stored & executed inside a DB server.

Steps:

- ① Create an implementation object of callable statement
- ② Set the values for IN parameters
- ③ Register the OUT parameter types.
- ④ Execute the procedure
- ⑤ Process the resultant data
- ⑥ Close the callable statement implementation object.

~~full  
length~~

Q) Once the data's are fetched from DB where it is stored in Java?

→ It is stored in the form of object created by default constructor of that current class where the concept of ORM is introduced.

\* Difference b/w Statement, CallableStatement & PreparedStatement

Statement	PreparedStatement	CallableStatement
Supports placeholder	Supports placeholder	Supports
does not support EP	Support EP	Supports EP
disadvantageous with Batch update	disadvantageous with Batch update	disadvantageous
createStatement	prepareStatement	prepareCall
executeQuery();	prepareStatement(query);	prepareCall(query);

## Differences

Statement	Prepared Statement	Callable Statement
① Statement compile & execute the query on every instance of execution.	② prepared statement compile once and execute many times (EP)	③ callable statement compile once & execute many time.
④ whenever we use batch with statement then a single batch can contain any kind of DML queries. (advantage)	④ whenever we use batch with prepared statement then a single batch can contain multiple similar kind of queries (disadvantage)	④ "
⑤ does not support placeholders	⑤ supports placeholder so, value can be entered by the user dynamically at the run time.	⑤ supports placeholder. — " —
⑥ query's are bind at the time of execution.	⑥ query are bind at the time of implementation object creation	⑥ " —
does not supports stored procedure	⑦ supports dynamic's stored procedure	⑧ supports stored procedure.

Q) what are different types of servers?

The different types of servers are as follows

1) DB Server : oracle, mysql, Sybase, Informix etc.

2) Application Server : JBOSS, IBM <sup>web</sup>sphere,  
oracle weblogic.

3) Web Server :- ApacheTomcat, OracleGlassFish

Q) what is the port no for ApacheTomcat & can  
the port number be changed.

→ 8080

& it can be change within Server.xml file  
which is a part of conf folder.

Webapps :- used to deploy web application onto  
ApacheTomcat Server.

work :- stores the data w.r.t translated  
Servlet's & is named by ApacheTomcat  
Server as filename.jsp.java which by  
default extends a class HttpJSPBase

3) with the difference a server & a servlet

Server

Server is a Software which  
is used to process the client  
request & generate the  
client request.

<sup>manages the</sup>  
~~reponde~~

Servlet : Servlet is a server side  
java program.

Servlet is used to perform all  
the 3 type's of logic  
(presentation persistence, &  
business logic) and also  
is used to process the http client

(\*) Explain all the 3 types of logic with Example?

### ① presentation logic

Note : \* Static application is used give only presentation logic.

\* dynamic application is an integration of all 3 difference types of logic

\* Define Servlet life cycle & explain the entire Servlet life cycle. 1, don't stop

Servlet life cycle represents the phases of a Servlet Object from the creation until the destruction.

There are 4 different phases of Servlet lifecycle

① Instantiation phase ② Object creation phase

② Initialization phase

③ Service phase

④ destruction phase.

On

The entire lifecycle of Servlet is managed by JEE Container.

For each & every client request, one request & response object is created by JEE Container within the Service phase always without multitasking.

req1

res1

- action ② initialization

Ques A Servlet is single threaded or multi threaded?

Ans → Servlet is multi threaded & can be made single threaded in 2 diff ways

① by writing a class which implements a marker class interface by name SingleThreadedModel.

SingleThreadedModel is a deprecated Interface.

② By making service() has synchronized.

\* Init() → called only once

Service() → called multiple times

destroy() → called only once

JEE

Container.

Write the different types of Servlet?

Abstract class

java.servlet.GenericServlet

① Not specific to any protocol

② Doesn't support Session

javax.servlet.Http.HttpServlet

HttpServlet → 8 types of http Request

- Get, Post, Put, Delete, Trace, option, Head, Connect

One Abstract method

Two concrete methods

① Specific to http protocol

② supports Session

Service()

Init()

only concrete method without

mandatory to override

destroy()

any abstract class

optional to override

doxxx - protected.

\* Post Request maintains data security since the data's are carried as a part of Http Request body to the server which is not displayed to the end user.

\* Post Request is used to deal with Unlimited data.

## Get

- \* Get request deals with only limited data ie 1024 character's

- \* In case of get Request, the data's are not carried to the server as a part of url which is displayed to the end user, hence not secure.

~~Note~~  
~~100% of a question~~ By default the type of Request is Get.

## List of Interfaces

- ① Servlet
- ② ServletConfig
- ③ ServletContext
- ④ ServletRequest
- ⑤ ServletResponse
- ⑥ HttpServletRequest
- ⑦ HttpServletResponse
- ⑧ RequestDispatcher
- ⑨ HttpSession

## Implicit Object's of JSP

- ① request - HttpServletRequest
- ② response - HttpServletResponse
- ③ config - ServletConfig
- ④ application - ServletContent
- ⑤ page - Object
- ⑥ session - HttpSession
- ⑦ Exception - Throwable
- ⑧ pageContent - pageContent
- ⑨ out - JspWriter

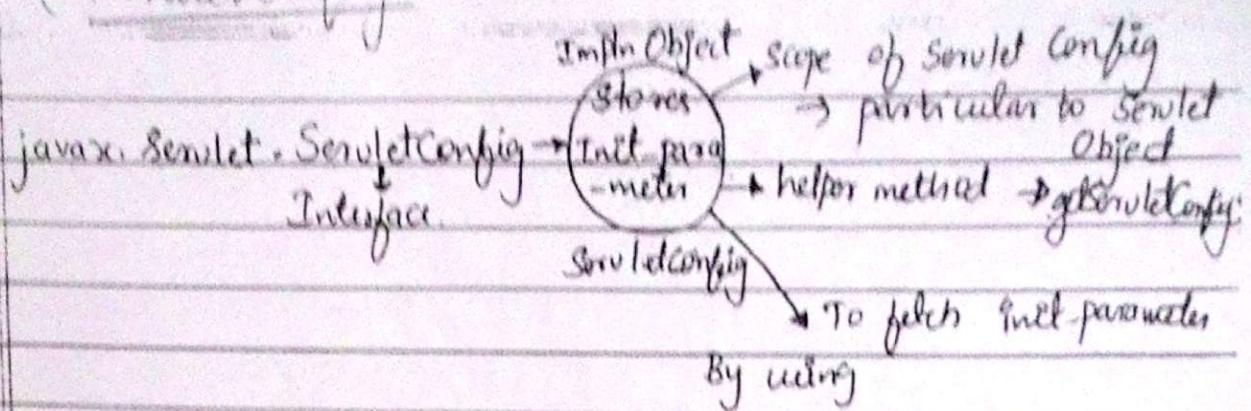
There are 3 different classes used in `<%@ page %>` which are follows

`javax.servlet.GenericServlet`

`javax.servlet.Http.HttpServlet`

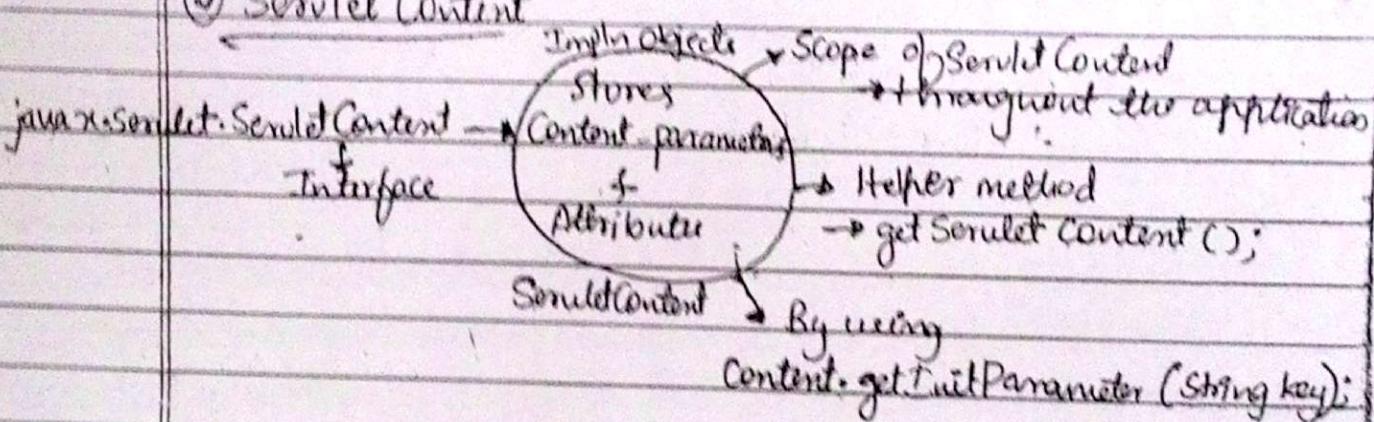
`java.io.PrintWriter`

## ① ServletConfig



String name = config. getInitParameter (String key)  
declare Init - parameters in → web. xml → within a servlet tag

## ② Servlet Content



content - parameters → Data in the form of k & v pair  
→ Both are String

Attribute → Data in the form of k & v pair  
unique → obj  
String

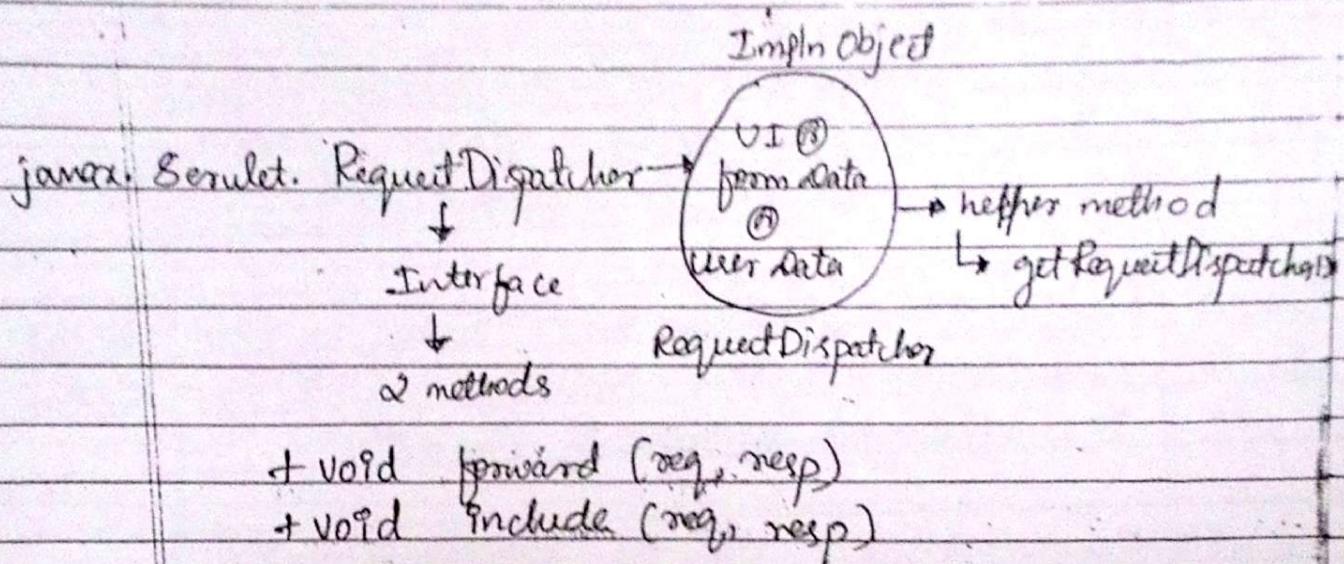
n, declare content - parameters → web. xml → outside the  
no of parameters → declare init - parameters → web. xml → within a servlet tag.  
can be declared.

+ void setAttribute (String key, Object obj)

+ Object getAttribute (String key) ↪ to add object into scope

Fetch the object back from scope

## Request Dispatcher



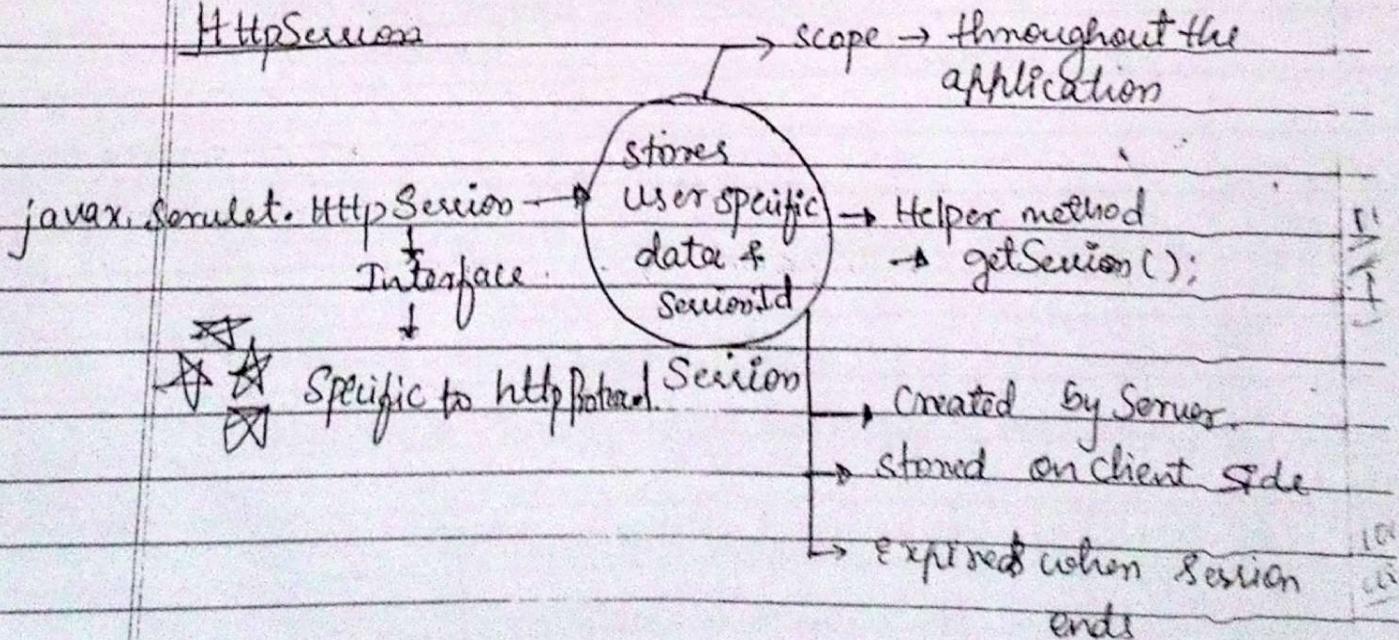
No of Servlet Object's for an Application → n

No of Config object's → depends on Servlet object

No of Context object for an application → Only one.

No of web.xml. → Only web.xml.

## HttpSession



\* Define Servlet chaining, mention the need for Servlet chaining and when is it performed.

\* Chaining from a Servlet to another resource which can be either a html, jsp or another Servlet is known as Servlet chaining.

\* Need :- To make the data's context of one Servlet available to another resource which can either be html, jsp or another Servlet.

\* Servlet Chaining can be performed only when a ServletRequest is made initially.

forward() :- used to maintain data consistency & type of request where data's are involved.

SendRedirect() :- used to chain to an external resource where the data are not involved.

~~Least  
Important~~

### Cookies & Session :-

\* Define Jsp & Explain in brief

\* what are the life cycle phases & methods of JSP?

\* Difference b/w Servlet & JSP?

\* Difference b/w lifecycle methods of Servlet & JSP?

\* Define Servlet Chaining, mention the need for Servlet chaining and when is it performed.

\* Chaining from a Servlet to another resource which can be either a html, jsp or another Servlet is known as Servlet Chaining.

\* Need :- To make the data's output of one Servlet available to another resource which can either be html, jsp or another Servlet.

\* Servlet Chaining can be performed only when a Servlet Request is made initially.

forward() :- used to maintain data consistency & type of request as where data's are produced.

sendRedirect() :- used to chain to an external resource where the data are not involved.

~~Ques~~  
~~Important~~

Cookies & Session :-

\* Define Jsp & Explain in brief

\* what are the life cycle phases & methods of JSPs

\* Difference b/w Servlet & JSP?

\* Difference b/w lifecycle methods of Servlet & JSP?

3 different Directives of JSP are

- 1) Page Directive
- 2) Include Directive
- 3) TagLib Directive

there are 4 different Scopes of JSP?

- \* 1) page - default scope of JSP
- 2) request - `HttpServletRequest`
- 3) Session - `HttpSession`
- 4) Application - `ServletContext`

~~100%  
checked~~

write all the 9 implicit object's of Jsp

(\*) what are JSTL tags? mention few.

~~NOTE :- By default JSP Extends HttpServlet &~~  
~~By default all the Translated Servlets~~  
extends a class `HttpJspBase`

Statement

Prepared Statement

(\*) Define Jsp & explain in brief?

- \* JSP is a high level Abstraction of Servlet
- \* Each & every JSP internally acts as a Servlet
- \* whenever a first-client request is made to JSP  
the JEE Container converts the JSP into ~~Translated~~  
Servlet which is known as Translated-Servlet (①)  
TServlet
- \* All the translated Servlets are stored in work folder of  
ApacheTomcat

- \* all the Translated Servlets by default Extends the class by name HttpJSPBase.
- \* Once the translated servlet created two different situations occurs
  - ① On successful compilation a TranslatedServlet class is generated and the response is rendered back to the client
  - ② If the compilation fails then the JEE container throws an error called 500 error. (server error) that mean an error with 500 code
- \* By default JSP extends Http.

Need: Jsp i.e.: used to give only presentation logic

— presentation logic is a logic which deals with presenting the contents onto the application.

### (\*) Lifecycle phases      lifecycle methods

- |                         |   |
|-------------------------|---|
| 1) Translation phase    | (1) <code>jspInit()</code>  |
| 2) Instantiation phase  | (2) <code>-jspService(HttpServlet<br/>Request, HttpServletReponse)</code> |
| 3) Initialization phase |   |
| 4) Service phase        | (3) <code>jspDestroy()</code>   |
| 5) Destruction phase    |   |

### (\*) Difference b/w Servlet & JSP

#### Servlet

- 1) Servlet is a Server Side java program which is used to perform all the 3 type of logic along with which takes the input from client Request.

#### JSP

- 1) Jsp is a Highlevel Abstraction of Servlet used to perform only presentation logic.

② we can write HTML code in Servlet

we can't write any HTML code inside jsp element

③ Servlet lifecycle has 4 phases

JSP lifecycle has 5 phases.

④

### lifecycle methods : difference.

#### Servlet

① init() is a method which is used to initialize the resources of servlet object by taking servletConfig as a parameter

~~only once~~

it can be overridden & it is public in nature

\* jspInit() is a method which is used to initialize the resources of JSP object it can be overridden & it is protected in nature

② service() is a method which is used to process the client request it is the only one abstract method present in servlet overriding is mandatory & it is public in nature

~~multiple~~

② jspService() is a method which is used to process the client request which is protected in nature & can't be overridden

③ destroy() → is used to close the costly resources can be overridden

~~only once~~

③ jspDestroy() is used to close the costly resources of JSP object can be overridden

## Difference b/w Cookie & a Session

Cookie	Session
1) Cookie is class present in javax.servlet	Session is an Interface present in javax.servlet & http
2) <sup>Class</sup> Cookie ^ is a temporary storage area which is used to store small amount of data on the client side which is created by Server.	Session ^ interface is a created by Server & stored at client side, used to store user specific data and session id
3) Cookie can be created by using Cookie (String name, String value) constructor.	Session can be created by using an helper method called getSession ()
4) Cookie can be disabled by the client (or) browser at any part of time	Session Object Expires once the browsing session ends and user client can be destroyed the session
5) When cookie age is set with -ve then the type of cookie is non-persistent which ends once the browser session ends when the type is PC ends for the first client request where the type is PC Expires after the time delay given by Server clocked	age is set either to a -ve or +ve or 0 the Session object expires once the browsing session ends.

## Functionalities of JSP Container

- 1) loads the application
- 2) parse web.xml
- 3) Create Servlet object by calling default constructor
- 4) create ServletConfig object - "
- 5) create one reg & resp objects for each & every request
- 6) destroy the Servlet object

Welcome file :-

A file @ a page which is automatically displayed whenever a client uses an application is known as welcome file @ landing page.

index.html → welcome file