Interview Quelion

1. Define JDBC & Explain the Architecture of JDBC
2. Explain all the 6 standard step's of JDBC
3. Define API & mention the 2 forms of API
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.?
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 & CallableStatement Interface.
10. Define BatchUpdate & Explain whether Batch is advantageous w.r.t Statement(I) & PreparedStatement(I).
11. Define JDBC Transaction & Explain why JDBC transaction is needed. along with the concept of ACID rules? (Automicity)
12. Define Stored procedure and write all the step's 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 & CallableStatement?

Static → only presentation logic
dynamic → All 3 types

Date:
Page No:

Servlet's & JSP

1. Define Server?, Servlet & write the differences b/w a server & a servlet?
2. Define JEE container? and mention it's 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 type's 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 Http Request?
10. Define Servlet chaining, Explain the need for Servlet chaining along with the type's & 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 & Session?

Date :
Page No :

Syntax :-

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

Static method

java.lang

Checked Exception → ClassNotFoundException

(2) Establish the Connection with DB Server.

java.sql.Connection con = null

JDBC Drivers in the form of jar
→ mysql-Driver.

con = DriverManager.getConnection ("jdbc:mysql://localhost:3306?user=root&password=dinga");

Interface

JDBC API

java.sql

2 static methods

registerDriver();

getConnection();

Helper class

• getConnection (url);

Static Helper Method present in Driver Manager

return type is Connection Interface

returns only one implementation object of Connection Interface.

getConnection() has 3 overloaded variant's

① getConnection (String url);
② getConnection (String url, properties info);
③ getConnection (String url, String user, String password);

Programming JEE

(1) Define JDBC?

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

(Eg)
* Why do yo go to achive loose coupling?
→ Since it doesn't affect our java Application. (2)

* (Eg) is it mandatory to use an API, to communicate b/w two applications?
→ It is not mandatory to use an API to communicate one application to another, but the result is tight coupling.

(2) what are the 6, standard steps of JDBC?
① Load & Register the Drivers
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 @ platform.

platform can be created by using

* Statement
  PreparedStatement
  CallableStatement

} Interfaces of JDBC API
↓
java.sql.

java.sql.Statement = Con.CreateStatement();

- Statement → Interface → JDBC API
- Con → Connection Interface → JDBC API
- CreateStatement → Helper method → returns → only one Implementation object of Statement Interface
  return type → Statement(I)

CreateStatement(); → Declared in Connection Interface
- Type of method → Non-static
  → Since interface can't have static method.

java.sql.PreparedStatement = Con.prepareStatement(Qry);

- PreparedStatement → Interface → JDBC API
- Con → Connection Interface → JDBC API
- prepareStatement → Helper method → returns only one imple object of PreparedStatement Interface
  return type → PreparedStatement

· prepareStatement (qry);
↳ Declared in Connection Interface
↳ Type of method → Non-static
↳ Since Interface can't have static method.

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

(*) java.sql. CallableStatement = con. prepareCall (qry);
CallableStatement → Interface → JDBC API
con → Connection Interface → JDBC API
prepareCall → Helper method → returns only one impl object of CallableStatement
return type → CallableStatement Interface.

~~prepareStatement~~
prepareCall (qry) → Declared in Connection Interface
↳ Type of method → Non-static
→ Since Interface can't have static method.

(4) Execute Sql queries @ Statement

+ boolean execute (SQL qry); → Can Execute any kind of SQL queries.
↳ Generic method
↳ return type → Boolean → True → DQL
False → DML, DDL

** Declared in Statement (I)
↓

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 @ DQL qry is passed.

② + int executeUpdate (only DML)

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

↳ return type → integer

↳ throws SQLException if DDL @ DQL qry is passed.

③ + 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.Execute();
pstmt.ExecuteUpdate();
pstmt.executeQuery();
} only Execution happens 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 or buffer memory which can fetch with the help
* of ResultSet Interface.

(5)

java.sql.ResultSet = Stmt.getResultSet();

ResultSet → Interface → JDBC API
Stmt → Interface → JDBC API
getResultSet() → Helper method → returns an Empty object of ResultSet (I) → return type → ResultSet

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 NullPointerException

* Costly resources are closed to enhance 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 number & fully qualified class name given by different DB Servers.

		Port No
(1) MS-SQL	→ Sql-Server Driver	→
(2) My-SQL	→ Driver	→ 3306
(3) Derby	→ Embedded Driver	→ 1527
(4) Oracle	→ Oracle Driver	→ 1521

* Advantages of JDBC?

(1) To achive 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)

Edited

Difference b/w Statement PreparedStatement, CallableStatement

* what is JDBC Transaction? mention the need & advantages of JDBC Transaction?

→ JDBC Transaction is considered as a single Buisness unit which is used execute multiple Sql statements
→ To maintain data consistency we go with JDBC Transaction.
→ Advantages: we can achive ACID properties

* Define Autonocity & relate DBC Transactions?

Autonocity → Do everything (or) Do nothing means autonocity.

Do everything reffers to → complete transaction (or) succeesfull transaction, where data consistency is maintained.

Do nothing → reffers to incomplete (or) unsuccessfull 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 begining again.

* whenever we use rollback() with ~~Savepoint~~ savepoint, the operation gets rolled back only after the savepoint 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 parameter's
3. Register the OUT parameter types.
4. Execute the procedure
5. process the resultant data
6. close the callable Statement implementation object.

(*) 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	prepared statement	callable statement
doesn't support's placeholder	support's placeholder	support's
doesn't support EP	support EP	support's EP
disadvantageous with Batch update	disadvantageous with Batch update	disadvantageous
createStatement	prepareStatement	prepareCall
Execute(qry);	prepareStatement(qry);	prepareCall(qry);

Differences

Statement	Prepared Statement	Callable Statement
1) Statement Compiles & Execute the query on every Instance of Execution.	1) preparedStatement Compile once and Execute many times (EP)	1) Callable Statement compiles once & Execute many time.
2) wherever we use Batch with Statement then a single batch can contain any kind of DML queries. (Advantages)	2) whenever we use Batch with prepared Statement then a Single batch can contain multiple similar kind of queries (disadvantageous	2) — " —
3) doesnot Support's placeholders	3) Supports placeholder So, value can be external by the user dynamically at the Run time.	3) Supports placeholder. — " —
4) query's are parsed at the time of execution.	4) query are parsed at the time of implementation object creation	4) — " —
5) doesnot Supports Stored procedure	5) Supports Supports Stored procedure	5) Supports storedprocedure.

1) what are different types of servers?

The different types of Servers as follows

1) DB Server :- Oracle, mysql, Sybase, Informix etc..

2) Application Server :- JBoss, IBM Websphere, oracle weblogic.

3) Web Server :- ApacheTomcat, OracleGlassFish

4) what is the port no for ApacheTomcat & can the portnumber be change.

→ 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

5) write the difference a Server & a Servlet

Server	Servlet
Server is a Software which is used to process the client request & serves the client request. (manages the resources)	Servlet :- Servlet is a Server side java program. Servlet is used to perform all the 3 type's of logic (presentation persistence, & buisness logic) and also is used 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 type's of logic

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

Servlet life cycle represent's the phases of a Servlet object from the creation until the destruction.
There are 4 different phases of Servlet lifecycle
① Instantiation phase ⑤ Object creation phase
② Initialization phase
③ Service phase
④ destruction phase.

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 instantiation & initialization

Type a Servlet is Single Threaded (or) Multi Threaded?
→ Servlet is MultiThreaded & can be make SingleThreaded in 2 diff ways

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

SingleThreadedModel is a depricated 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

javax.servlet.GenericServlet
① Not specific to any protocol
② Doesn't support's Session
↓
One Abstract method
↓
Service()
↓
mandatory to override

Two concrete method
↓
Init()
destroy()
↓
optional to override

javax.servlet.http.HttpServlet
HttpServlet → 8 types of http Request
- Get, Post, Put, Delete, Trace, option, Head, Connect
(1) Specific to http protocol
(2) supports Session
↓
only concrete method without any abstract-class
↓
doXxx - protected.

* Post Request maintains data security since the data's are carried as a part of HttpRequest body to the server which is not displayed to the enducer.
* Post Request is used to deal with Unlimited data

Get

* Get request deals with only limited data ie 1024 characters
* In case of Get Request, the data's are carried to the server as a part of url which is displayed to the enduser. hence not secure.

NOTE 100% asked
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 - ServletContext
5. page - Object
6. session - HttpSession
7. Exception - Throwable
8. pageContext - pageContext
9. out - JSPWriter

There are 3 different classes used w.r.t Servlet which are follows

javax.servlet.GenericServlet
javax.servlet.http.HttpServlet
java.io.PrintWriter

ServletConfig

javax.Servlet.ServletConfig → Interface

Impln Object stores Init parameter

Scope of ServletConfig → particular to Servlet Object

Helper method → getServletConfig();

ServletConfig → To fetch Init-parameter By using

String name = config.getInitParameter(String key)

Declare Init-parameters in → web.xml → within a servlet tag

② ServletContext

javax.Servlet.ServletContext → Interface

Impln Objects stores Content-parameters & Attribute

Scope of ServletContext → throughout the application

Helper method → getServletContext();

ServletContext → By using ContentgetInitParameter(String key);

Content-parameters → Data in the form of K & V pair → both are String

Attribute → Data in the form of K & V pair → unique String, obj

'n' no of parameter can be

Declare Content-parameters → web.xml → Outside the servlet tag

Declare Init-parameters → web.xml → within a Servlet Tag

declared.

+ void setAttribute(String key, Object obj) → to add object into Scope

+ object getAttribute(String key) → Fetch the object back from Scope

Request Dispatcher

Impln Object

javax.Servlet.RequestDispatcher → (VI @ form data @ user data) → helper method → getRequestDispatcher()

RequestDispatcher

↓ Interface

↓ 2 methods

+ void forward (req, resp)
+ void include (req, resp)

No of Servlet Objet's for an Application → n

No of Config objects → depends on Servlet Object

No of Context Object for an application → Only One.

No of web.xml → only web.xml

HttpSession → scope → throughout the application

javax.Servlet.HttpSession → (stores user specific data & SessionId) → Helper method → getSession();

↓ Interface

↓ Specific to http protocol

Session
- Created by Server
- Stored on client side
- Expired when Session ends

(*) Define Servlet chaining, mention the need for Servlet chaining and when it is 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 avail 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 as where data's are involved.

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

Cookies & Session :

(*) Define Jsp & Explain in brief
(*) what are the life cycle phases & methods of JSP?
(*) Difference b/w Servlet & JSP?
(*) Difference the lifecycle methods of Servlet & Jsp?

(*) Define Servlet Chaining, mention the need for Servlet chaining and when it is 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 extract 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 on where data's are involved.

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

Cookies & Session :

(*) Define Jsp & Explain in brief
(*) what are the life cycle phases & methods of JSP?

(*) Difference b/w Servlet & JSP?
(*) Difference the 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

most asked

write all the 9 Implicit Object's of Jsp

(*) what are JSTL tags? mention few.

NOTE :- By default JSP Extends HttpJsp. &
By default all the Translated Servlets
extends a class HttpJspBase

~~Statement~~ Prepared Statement

(*) Define Jsp & explain in brief?

* JSP is a higel level Abstraction of Servlet
* Each & every JSP internally acts as a Servlet
* whenever a first-client request is made to Jsp the JEE Containers converts the JSP into ~~Translated~~ Servlet which is known as Translated-Servlets (or) 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 Successfull compilation a TranslatedServlet .class is generated and the response is rendered back to the client
* ② If the compilation fails then the JEE container throw's an error called 500 error. (Server error) that mean an error with 500 code
* By default JSP Extends Http.

Need: Jsp is used to give only presentation logic

presentation logic is a logic which deals with presenting the contents onto the applicatio

⊛ Lifecycle phases

1) Translation phase
2) Instantiation phase
3) Initialization phase
4) Service phase
5) Destruction phase

Life cycle methods

(1) jspInit()
(2) _jspService(HttpServletRequest, HttpServletResponse)
(3) jspDestroy()

⊛ Difference b/w Servlet & JSP

Servlet	JSP
1) Servlet is a Server Side java program which is used to perform all the 3 type of logic along with which takes the HttpClient Request	JSP is a High level 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	jsp
① only Once	init() is a method which is used to initialize the resources of servlet object by taken ServletConfig as a parameter it can overriden & it is public in nature	* jspInit() is a method which is used to initialize the resources of Jsp object it can be overriden & it is protected in nature
② multiple	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	② _jspService() is a method which is used to process the client request which is protected in nature & can't be overriden
③ only once	destroy() → is used to close the costly resources can be overriden, public in nature	③ jspDestroy() is used to close the costly resources of jsp object can be overriden & protected in nature

Difference b/w Cookie & a Session

Cookie	session.
1) Cookie is class present in javax.servlet Class	Session is an Interface present in javax.servlet.http
2) Cookie is a temporary Storage area which is used to store small amount of data on the client side which is created by servers.	Session is an interface 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	3) 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	4) Session object expires once the browsing session ends and ~~user~~ client can be destroy the ~~Sess~~
5) When cookie age is set with -ve then the type of cookie is non-persistence which ends once the browsing session ends when the type is NPC ends for the first client request where the type is PC Expires only the time -delay given by servlet clarkser	5) 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 Servlet Config object — " —
5) create one req & 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