# **Assignment 7**

Q1. What is the use of JDBC in java?

Before JDBC, ODBC API was the database API to connect and execute the query with the database. But ODBC API uses ODBC driver which is written in C language. That is why Java has defined its own API (JDBC API) that uses JDBC drivers (written in Java language).

We can use JDBC API to handle database using Java program and can perform the following activities:

1. Connect to the database
2. Execute queries and update statements to the database
3. Retrieve the result received from the database.

Q2. What are the steps involved in JDBC?

There are 5 steps to connect any java application with the database using JDBC. These steps are as follows:

* Register the Driver class
* Create connection
* Create statement
* Execute queries
* Close connection

### 1) Register the driver class

|  |
| --- |
| The **forName()** method of Class class is used to register the driver class. This method is used to dynamically load the driver class. |

### **Syntax of forName() method**

**public** **static** **void** forName(String className)**throws** ClassNotFoundException

### 2) Create the connection object

|  |
| --- |
| The **getConnection()** method of DriverManager class is used to establish connection with the database. |

### **Syntax of getConnection() method**

1) **public** **static** Connection getConnection(String url)**throws** SQLException

2) **public** **static** Connection getConnection(String url,String name,String password)

**throws** SQLException

### 3) Create the Statement object

|  |
| --- |
| The createStatement() method of Connection interface is used to create statement. The object of statement is responsible to execute queries with the database. |

### **Syntax of createStatement() method**

**public** Statement createStatement()**throws** SQLException

### 4) Execute the query

|  |
| --- |
| The executeQuery() method of Statement interface is used to execute queries to the database. This method returns the object of ResultSet that can be used to get all the records of a table. |

### **Syntax of executeQuery() method**

**public** ResultSet executeQuery(String sql)**throws** SQLException

### 5) Close the connection object

|  |
| --- |
| By closing connection object statement and ResultSet will be closed automatically. The close() method of Connection interface is used to close the connection. |

### **Syntax of close() method**

**public** **void** close()**throws** SQLException

Q3. What are the types of statement in JDBC in java?

The statement interface is used to create SQL basic statements in Java it provides methods to execute queries with the database. There are different types of statements that are used in JDBC as follows:

* Create Statement
* Prepared Statement
* Callable Statement

**1.** **Create a Statement:** From the connection interface, you can create the object for this interface. It is generally used for general**–**purpose access to databases and is useful while using static SQL statements at runtime.

**Syntax:**

Statement statement = connection.createStatement();

**2. Prepared Statement** represents a recompiled SQL statement, that can be executed many times. This accepts parameterized SQL queries. In this, “?” is used instead of the parameter, one can pass the parameter dynamically by using the methods of PREPARED STATEMENT at run time.

**Illustration:**

Considering in the people database if there is a need to INSERT some values, SQL statements such as these are used:

INSERT INTO people VALUES ("Ayan",25);  
INSERT INTO people VALUES("Kiyra",32);

**3. Callable Statement** are stored procedures which are a group of statements that we compile in the database for some task, they are beneficial when we are dealing with multiple tables with complex scenario & rather than sending multiple queries to the database, we can sendtherequired data to the stored procedure & lower the logic executed in the database server itself. The Callable Statement interface provided by JDBC API helps in executing stored procedures.

**Syntax:** To prepare a CallableStatement

CallableStatement cstmt = con.prepareCall("{call Procedure\_name (?, ?}");

Q4. What is Servlet in Java?

**Servlet** technology is robust and scalable because of java language. Before Servlet, CGI (Common Gateway Interface) scripting language was common as a server-side programming language.

Servlet can be described in many ways, depending on the context.

* Servlet is a technology which is used to create a web application.
* Servlet is an API that provides many interfaces and classes including documentation.
* Servlet is an interface that must be implemented for creating any Servlet.
* Servlet is a class that extends the capabilities of the servers and responds to the incoming requests. It can respond to any requests.
* Servlet is a web component that is deployed on the server to create a dynamic web page.

Q5. Explain the life Cycle of servlet?

The web container maintains the life cycle of a servlet instance. Let's see the life cycle of the servlet:

1 Servlet class is loaded.

2 Servlet instance is created.

3 init method is invoked.

4 service method is invoked.

5 destroy method is invoked.

### 1) Servlet class is loaded

The classloader is responsible to load the servlet class. The servlet class is loaded when the first request for the servlet is received by the web container.

2) Servlet instance is created

The web container creates the instance of a servlet after loading the servlet class. The servlet instance is created only once in the servlet life cycle.

### 3) init method is invoked

|  |
| --- |
| The web container calls the init method only once after creating the servlet instance. The init method is used to initialize the servlet. It is the life cycle method of the javax.servlet.Servlet interface. Syntax of the init method is given below: |

**public** **void** init(ServletConfig config) **throws** ServletException

### 4) service method is invoked

The web container calls the service method each time when request for the servlet is received. If servlet is not initialized, it follows the first three steps as described above then calls the service method. If servlet is initialized, it calls the service method. Notice that servlet is initialized only once. The syntax of the service method of the Servlet interface is given below:

**public** **void** service(ServletRequest request, ServletResponse response) **throws** ServletException, IOException

5) destroy method is invoked

The web container calls the destroy method before removing the servlet instance from the service. It gives the servlet an opportunity to clean up any resource for example memory, thread etc. The syntax of the destroy method of the Servlet interface is given below:

**public** **void** destroy()

Q6. Explain the difference between the RequestDispatcher.forward() and HttpServletResponse.sendRedirect() methods?

The Key **Difference between** **forward() and sendRedirect()** is that **forward()** method works at server-side. whereas **sendRedirect()** method works at client side.

* forward(): This method is declared in ***RequestDispatcher*** Interface.
* sendRedirect(): This method is declared in ***HttpServletResponse*** Interface.

|  |  |
| --- | --- |
| **forward()** | **sendRedirect()** |
| forward() is method of RequestDispatcher interface | sendRedirect() is the method of HttpServletResponse In forward(). |
| In forward() redirect happens at server end and not visible to client. | In sendRedirect(), redirection happens at client end and it’s visible to client. |
| It is faster than the redirect. | It is slower than a forward, since it requires two browser requests (one for actual request and another for redirected request). |
| In case of forwarding () original URL remains unaffected. | In the case of sendRedirect() browser knows that it’s making a new request, so the original URL changes. |
| Transfer the request to the same server. | Transfer the request different server. |
| When forward is called on RequestDispather object we pass request and response object so our old request object is present on a new resource which is going to process our request | In the case of SendRedirect call old request and response, the object is lost because it’s treated as a new request by the browser |
| Syntax: forward (ServletRequest request, ServletResponse response) | Syntax: void sendRedirect(String url) |

Q7. What is the purpose of the doGet() and doPost() methods in a servlet?

**Get method**

Get method main job is asking the server for the resources. Get is one of the simplest HTTP methods. Get method has a size limitation of 1024 characters.

**Post method**

Post method is Provide information. by using post, we can send as well as request data to the server.

|  |  |
| --- | --- |
| **doGet()** | **doPost()** |
| In doget() Parameters not encrypted. | In dopost() Parameters encrypted. |
| doget() allows bookmark. | dopost() disallows bookmark. |
| doget() method is idempotent. | dopost() method does not idempotent. |
| In doget() not change anything on the server. | In dopost() server is expected to remember. |
| doget() is request information. | dopost() is provide information. |
| In doget() parameters are appended to URL and sent with header information. | In dopost(), on the other hand, will send the information through socket back to the webservers and it won’t show in the URL bar. |
| doget() is Faster. | dopost() Slower. |
| In doget() only 1024 characters limit. | In dopost() does not have limit. |

Q8. Explain the JSP Model-View-Controller (MVC) architecture.

# MVC Architecture in Java

The Model-View-Controller (MVC) is a well-known design pattern in the web development field. It is way to organize our code. It specifies that a program or application shall consist of data model, presentation information and control information. The MVC pattern needs all these components to be separated as different objects.

The model designs based on the MVC architecture follow MVC design pattern. The application logic is separated from the user interface while designing the software using model designs.

The MVC pattern architecture consists of three layers:

* **Model:** It represents the business layer of application. It is an object to carry the data that can also contain the logic to update controller if data is changed.
* **View:** It represents the presentation layer of application. It is used to visualize the data that the model contains.
* **Controller:** It works on both the model and view. It is used to manage the flow of application, i.e., data flow in the model object and to update the view whenever data is changed.

The Model contains the simple Java classes, the View used to display the data and the Controller contains the servlets. Due to this separation the user requests are processed as follows:

1 A client (browser) sends a request to the controller on the server side, for a page.

2 The controller then calls the model. It gathers the requested data.

3 Then the controller transfers the data retrieved to the view layer.

4 Now the result is sent back to the browser (client) by the view.

Q9. What are some of the advantages of Servlets?

A servlet can be imagined to be as an applet running on the server side. Some of the other server-side technologies available are Common Gateway Interface (CGI), server-side JavaScript and Active Server Pages (ASP).

## **Advantages of servlets**

**• Persistent:** Servlets remain in memory until explicitly destroyed. This helps in serving several incoming requests. Servlets establishes connection only once with the database and can handle several requests on the same database. This reduces the time and resources required to establish connection again and again with the same database.

**• Portable:** Since servlets are written in Java, they are portable. That is, servlets are compatible with almost all operating systems. The programs written on one operating system can be executed on another operating system.

**• Server-independent:** Servlets are compatible with any web server available today. Most of the software vendors today support servlets within their web server products.

**• Protocol-independent:** Servlets can be created to support any of the protocols like FTP commands, Telnet sessions, NNTP newsgroups, etc. It also provides extended support for the functionality of HTTP protocol.

**• Extensible:** Servlets being written in Java, can be extended and polymorphed into the objects that suits the user requirement.

**• Secure:** Since servlets are server-side programs and can be invoked by web server only, they inherit all the security measures taken by the web server. They are also safe from the problems related to memory management as Java does not support the concept of pointers and perform garbage collection automatically.

**• Fast:** Since servlets are compiled into bytecodes, they can execute more quickly as compared to other scripting languages. The bytecode compilation feature helps servlets to give much better performance. In addition, it also provides advantage of strong error and type checking.

Q10.What are the limitations of JSP?

JavaServer Pages (JSP) is a technology that allows developers to create dynamic web pages. However, there are some limitations to JSP that can be overcome by integrating it with other technologies, such as servlets. Some of the limitations of JSP include:

* Limited control over the HTML output: With JSP, developers have limited control over the HTML output, as the JSP engine takes care of generating the HTML code. This can make it difficult to fine-tune the layout and design of a web page.
* Limited functionality: JSP is primarily used for generating dynamic content and does not provide as many features as other web development frameworks.
* Limited scalability: JSP is not as scalable as other web development frameworks, as it can consume a lot of resources when handling large requests.

To overcome these limitations, developers can integrate JSP with servlets. Servlets are Java classes that can be used to handle HTTP requests and generate dynamic content. By using servlets in conjunction with JSP, developers can have more control over the HTML output and functionality of a web page.