Assignment – 7 (Java)

Q1.What is the use of JDBC in java?

Answer:JDBC (Java Database Connectivity) is a Java API that provides a standard way to interact with databases. It enables Java programs to connect to and manipulate relational databases by providing methods to establish connections, execute queries, retrieve results, and update data. JDBC allows developers to write database-independent code, making it easier to switch between different database systems. It plays a crucial role in Java-based applications that require database access, enabling them to store, retrieve, and manipulate data efficiently and securely.

Q2.What are the steps involved in JDBC?

Answer:

The steps involved in JDBC are as follows:

1. Load the JDBC driver using the `Class.forName()` method.
2. Establish a connection to the database using the `DriverManager.getConnection()` method.
3. Create a statement object using the connection's `createStatement()` method.
4. Execute SQL queries or commands using the statement's `executeQuery()` or `executeUpdate()` methods.
5. Retrieve the results from the query using the ResultSet object.
6. Process and manipulate the data as needed.
7. Close the ResultSet, statement, and connection using their respective `close()` methods to release resources.

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

Answer:In JDBC, there are three types of statements available to execute SQL queries and commands:

1. Statement: It is the simplest type and can execute SQL statements without parameters. However, it is prone to SQL injection attacks.
2. PreparedStatement: It is pre-compiled and allows the execution of parameterized queries. It offers better performance and security as it prevents SQL injection.
3. CallableStatement: It is used to execute stored procedures in the database. It can also pass input and output parameters to the stored procedures and retrieve results from them.

Q4.What is Servlet in Java?

Answer: A Servlet in Java is a server-side component that dynamically generates web content and handles client requests. It operates within a web container, such as Apache Tomcat, and extends the capabilities of a web server. Servlets receive HTTP requests from clients, process them, and generate responses, which can be in various formats like HTML, XML, or JSON. They can handle session management, form processing, database interactions, and other web-related tasks. Servlets are written in Java and provide a scalable and platform-independent solution for building web applications.

Q5.Explain the life Cycle of servlet?

Answer:The life cycle of a servlet consists of several stages:

1. Initialization: The servlet container loads the servlet class and calls its `init()` method to initialize resources.
2. Request Handling: The container invokes the `service()` method to process client requests. The servlet reads request parameters, performs business logic, and generates a response.
3. Multi-threading: Each request is processed in a separate thread by invoking the `service()` method concurrently.
4. Destruction: When the container shuts down or removes the servlet, it calls the `destroy()` method to release resources.

Throughout this life cycle, the servlet handles multiple requests, maintaining its state and responding to clients' needs.

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

Answer:The `RequestDispatcher.forward()` method is used to forward a request from one servlet to another resource (servlet, JSP, or HTML) on the server-side. The original request and response objects are sent to the target resource, allowing them to share the same request attributes and parameters. It is an internal redirection.

On the other hand, `HttpServletResponse.sendRedirect()` method sends a redirect response to the client browser. It instructs the browser to send a new request to a different URL. The client browser receives the redirect response and initiates a new request to the specified URL. It is an external redirection.

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

Answer:The `doGet()` and `doPost()` methods in a servlet are used to handle HTTP GET and POST requests, respectively.

The `doGet()` method is invoked when a client sends an HTTP GET request to the servlet. It is commonly used for data retrieval, where parameters are passed through the URL.

The `doPost()` method is invoked when a client sends an HTTP POST request to the servlet. It is typically used for submitting data to the server, such as form submissions, where parameters are included in the request body.

By implementing these methods, a servlet can handle different types of requests and perform specific operations based on the request type.

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

Answer: The JSP Model-View-Controller (MVC) architecture is a design pattern that separates the concerns of an application into three components. The model represents the data and business logic, the view defines the presentation layer, and the controller handles the communication between the model and view. In this architecture, JSP pages serve as the view layer, JavaBeans or servlets act as the model, and servlets or controllers manage the flow and interaction between the model and view. This separation promotes modular and maintainable code by clearly defining the responsibilities of each component.

Q9.What are some of the advantages of Servlets?

Answer: Some advantages of using servlets are as follows:
1. Platform Independence: Servlets are written in Java, making them platform-independent and capable of running on any platform that supports Java.
2. Performance: Servlets are efficient as they are loaded once and can handle multiple requests concurrently, reducing the overhead of creating a new process for each request.
3. Scalability: Servlets can handle a large number of simultaneous requests, making them suitable for high-traffic applications.
4. Integration: Servlets can easily integrate with other Java technologies like JavaServer Pages (JSP), JavaBeans, and Enterprise JavaBeans (EJB).
5. Security: Servlets provide built-in security mechanisms for handling authentication, authorization, and secure communication.

Q10.What are the limitations of JSP?

Answer: Some limitations of JSP are as follows:
1. Steep Learning Curve: JSP requires a solid understanding of Java and web development concepts, making it challenging for beginners.
2. Mixing Logic and Presentation: JSP can lead to the mixing of business logic and presentation code, making the code less maintainable and harder to debug.
3. Performance Overhead: JSP pages need to be compiled into servlets, which introduces an initial overhead. Additionally, frequent modifications to JSP files can impact performance.
4. Limited Control: JSP provides limited control over the generated HTML output, making it difficult to achieve fine-grained customization.
5. Tight Coupling: JSP can lead to tight coupling between the presentation layer and business logic, making it harder to change either component independently.