Database Connectivity with JDBC

Q1: What is JDBC in Java?

- **A:** JDBC (Java Database Connectivity) is an API in Java that defines how a client can access a database, enabling the creation and execution of SQL statements.

Q2: How do you create a JDBC connection in Java?

- **A:** You can create a JDBC connection using `DriverManager.getConnection()` with parameters for the database URL, username, and password.

Q3: What SQL operations can JDBC support?

- **A:** JDBC supports SQL operations such as CREATE, INSERT, UPDATE, DELETE, and SELECT.

Q4: What class do you use to execute SQL statements in JDBC?

- **A:** You use the `Statement` class to execute SQL statements, such as `executeUpdate` for updating data.

File I/O in Java

Q5: What is `FileInputStream` used for in Java?

- **A:** `FileInputStream` is used for reading data from a file as a sequence of bytes.

Q6: How would you read data from a file and write it to another file in Java?

- **A:** You can use `FileInputStream` to read from a file and `FileOutputStream` to write data to another file within a try-catch block for error handling.

Q7: Why do we use `finally` when working with file streams?

- **A:** The `finally` block is used to close streams to release resources and avoid memory leaks.

BufferedReader and InputStreamReader

Q8: What is the role of `BufferedReader` in Java?

- **A:** `BufferedReader` is used to read text from an input stream efficiently by buffering characters to optimize input operations.

Q9: How do you create a `BufferedReader` object to read from the keyboard?

- **A:** You create a `BufferedReader` object by wrapping it around an `InputStreamReader`, which reads from `System.in`.

Q10: Why is `BufferedReader` preferred over `InputStreamReader` alone for reading text?

- **A:** `BufferedReader` provides the `readLine` method, which reads a whole line of text, making it more efficient than reading character by character.

Standard Streams in Java

Q11: What are the standard input, output, and error streams in Java?

- **A:** The standard streams in Java are `System.in` (input), `System.out` (output), and `System.err` (error).

Q12: How do you print output to the console in Java?

- **A:** You use `System.out.println()` to print output to the console.

Q13: How is `System.err` typically used in Java applications?

- **A:** `System.err` is used to output error messages or exceptions, often displayed in red in many IDEs and consoles.

I/O Operations in Java

Q14: What is an I/O stream in Java?

- **A:** An I/O stream is a sequence of data used to perform input and output operations, where `InputStream` reads data, and `OutputStream` writes data.

Q15: What package contains classes for I/O operations in Java?

- **A:** The `java.io` package contains classes for performing input and output operations.

Certainly! Here are additional questions and answers based on the Java I/O operations and database concepts:

Advanced JDBC Operations

Q16: How can you execute an SQL query that returns a `ResultSet` in JDBC?

- **A:** Use the `executeQuery()` method of the `Statement` object to execute an SQL query that returns a `ResultSet`.

Q17: What is the purpose of the `ResultSet` object in JDBC?

- **A:** `ResultSet` is used to store the data retrieved from a database query, allowing navigation and access to each row and column of the result.

Q18: How do you prevent SQL injection in JDBC when inserting data?

- **A:** Use `PreparedStatement` instead of `Statement` to insert data, as `PreparedStatement` safely handles parameterized queries and prevents SQL injection attacks.

File Handling in Java

Q19: What is `FileOutputStream` used for in Java?

- **A:** `FileOutputStream` is used to write data to a file as a sequence of bytes.

Q20: How can you check if a file exists before reading it in Java?

- **A:** Use the `File` class and call its `exists()` method to check if a file exists before reading it.

Q21: What happens if `FileInputStream` or `FileOutputStream` cannot find the specified file?

- **A:** `FileInputStream` throws a `FileNotFoundException` if the file doesn't exist, while `FileOutputStream` creates a new file if it doesn't exist (if it's used in write mode).

BufferedReader and BufferedWriter

Q22: What is `BufferedWriter` used for in Java?

- **A:** `BufferedWriter` is used to write text to an output stream efficiently by buffering characters, which reduces the number of I/O operations.

Q23: How do you write a line of text to a file using `BufferedWriter`?

- **A:** You can use the `write()` method followed by `newLine()` to write a line of text, or use `write()` and add a newline character manually.

Q24: Why is it important to close `BufferedReader` and `BufferedWriter` after use?

- **A:** Closing these objects releases the resources associated with the streams, and any buffered output is flushed to the destination, preventing resource leaks.

Standard I/O Streams

Q25: How can you redirect `System.out` to write to a file instead of the console?

- **A:** Use `System.setOut(new PrintStream(new FileOutputStream("output.txt")))` to redirect `System.out` to a file.

Q26: What is `System.in` typically connected to?

- **A:** `System.in` is typically connected to the keyboard (standard input) in a console application.

Q27: Can you change the destination of `System.err`? If so, how?

- **A:** Yes, you can change `System.err` by using `System.setErr(new PrintStream(new FileOutputStream("error.log")))` to redirect error output to a file.

Serialization and Deserialization

Q28: What is serialization in Java?

- **A:** Serialization is the process of converting an object into a byte stream, allowing it to be easily saved to a file or transferred over a network.

Q29: Which interface must a class implement to be serializable?

- **A:** A class must implement the `Serializable` interface to be eligible for serialization.

Q30: How do you deserialize an object in Java?

- **A:** Use `ObjectInputStream` and its `readObject()` method to deserialize an object from a byte stream.

File and Directory Operations

Q31: How can you create a new directory in Java?

- **A:** Use the `mkdir()` or `mkdirs()` method of the `File` class to create a new directory.

Q32: What is the difference between `mkdir()` and `mkdirs()`?

- **A:** `mkdir()` creates a single directory, while `mkdirs()` creates the specified directory and any necessary parent directories.

Q33: How can you list all files in a directory in Java?

- **A:** Use the `listFiles()` method of the `File` class to get an array of `File` objects representing the files and directories within a directory.

InputStream and OutputStream

Q34: What is the difference between `InputStream` and `Reader` in Java?

- **A:** `InputStream` reads raw bytes, which is ideal for binary data, while `Reader` reads characters, making it suitable for text data.

Q35: What is `DataOutputStream` used for in Java?

- **A:** `DataOutputStream` is used to write primitive Java data types (e.g., int, double) to an output stream in a machine-independent way.

Q36: Can you read a file line by line using `FileInputStream`?

- **A:** No, `FileInputStream` reads raw bytes. To read a file line by line, you should use `BufferedReader` wrapped around a `FileReader`.

Error Handling in I/O Operations

Q37: Why is it important to handle exceptions in I/O operations?

- **A:** I/O operations are prone to errors like `FileNotFoundException` or `IOException`, which must be handled to ensure the program runs smoothly and avoids crashing.

Q38: What exception is commonly thrown when file reading fails?

- **A:** `FileNotFoundException` is thrown when the specified file cannot be found.

Q39: How can you ensure that a file stream is always closed, even if an error occurs during reading?

- **A:** Use a `try-with-resources` statement or a `finally` block to ensure the file stream is closed, regardless of whether an error occurs.

Properties and Configurations in I/O

Q40: How can you read configuration data from a properties file in Java?

- **A:** Use the `Properties` class and its `load()` method to read configuration data from a properties file.

Q41: What method is used to write key-value pairs to a properties file?

- **A:** Use the `store()` method of the `Properties` class to write key-value pairs to a file.

Q42: How can you retrieve a specific property from a `Properties` object?

- **A:** Use the `getProperty()` method, passing the key as an argument to retrieve the corresponding value.
