Step 1: Create the `Student` Class

This class implements the `Serializable` interface, allowing objects of this class to be serialized.

```java
import java.io.Serializable;

class Student implements Serializable {
    private static final long serialVersionUID = 1L; // Serial version ID for compatibilit
    private String name;
    private int age;

    public Student(String name, int age) {
        this.name = name;
        this.age = age;
    }

    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }

    @Override
    public String toString() {
        return "Student{name='" + name + "', age=" + age + "}";
    }
}
```

Step 2: Serialization and Deserialization

The following `SerializationTest` class demonstrates serializing a `Student` object to a file and deserializing it back.

```
import java.io.FileOutputStream;
import java.io.FileInputStream;
import java.io.ObjectOutputStream;
import java.io.ObjectInputStream;
import java.io.IOException;

public class SerializationTest {
    public static void main(String[] args) {
        Student student = new Student("Alice", 20);

        // Serialize the student object
        try (FileOutputStream fileOut = new FileOutputStream("student.ser");
             ObjectOutputStream out = new ObjectOutputStream(fileOut)) {
            out.writeObject(student);
            System.out.println("Serialization successful! Object written to student.ser");
        } catch (IOException e) {
            e.printStackTrace();
        }

        // Deserialize the student object
        try (FileInputStream fileIn = new FileInputStream("student.ser");
             ObjectInputStream in = new ObjectInputStream(fileIn)) {
            Student deserializedStudent = (Student) in.readObject();
            System.out.println("Deserialization successful! Object read from student.ser")
            System.out.println("Deserialized Student: " + deserializedStudent);
        } catch (IOException | ClassNotFoundException e) {
            e.printStackTrace();
        }
    }
}
```

Explanation

1. **Serialization**: The `ObjectOutputStream` writes the `student` object to the file `student.ser`.
2. **Deserialization**: The `ObjectInputStream` reads the serialized object from `student.ser` and casts it back to a `Student` object.

Output

Running this program will generate an output similar to:

```
Serialization successful! Object written to student.ser
Deserialization successful! Object read from student.ser
Deserialized Student: Student{name='Alice', age=20}
```

This code should help you understand the basics of serialization and deserialization in Java. Let me know if you'd like to expand on this example!