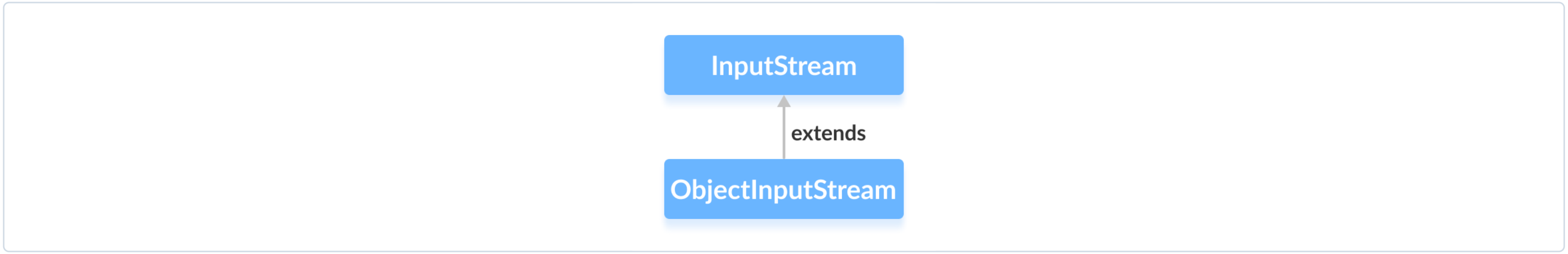
Java ObjectInputStream Class

In this tutorial, we will learn about Java ObjectOutputStream and its methods with the help of examples.

The `ObjectInputStream` class of the `java.io` package can be used to read objects that were previously written by `ObjectOutputStream`.

It extends the `InputStream` abstract class.



Before you learn about the `ObjectInputStream` class, make sure you know about the ObjectOutputStream Class.

Working of ObjectInputStream

The `ObjectInputStream` is mainly used to read data written by the `ObjectOutputStream`.

Basically, the `ObjectOutputStream` converts Java objects into corresponding streams. This is known as serialization. Those converted streams can be stored in files or transferred through networks.

Now, if we need to read those objects, we will use the `ObjectInputStream` that will convert the streams back to corresponding objects. This is known as deserialization.

Create an ObjectInputStream

In order to create an object input stream, we must import the `java.io.ObjectInputStream` package first. Once we import the package, here is how we can create an input stream.

```
// Creates a file input stream linked with the specified file
FileInputStream fileStream = new FileInputStream(String file);

// Creates an object input stream using the file input stream
ObjectInputStream objStream = new ObjectInputStream(fileStream);
```

In the above example, we have created an object input stream named `objStream` that is linked with the file input stream named `fileStream`.

Now, the `objStream` can be used to read objects from the file.

Methods of ObjectInputStream

The `ObjectInputStream` class provides implementations of different methods present in the `InputStream` class.

read() Method

- `read()` - reads a byte of data from the input stream
- `readBoolean()` - reads data in boolean form
- `readChar()` - reads data in character form

- `readInt()` - reads data in integer form
- `readObject()` - reads the object from the input stream

Example 1: Java ObjectInputStream

Let's see how we can use the `ObjectInputStream` class to read objects written by the `ObjectOutputStream` class.

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

class Main {
    public static void main(String[] args) {

        int data1 = 5;
        String data2 = "This is programiz";

        try {
            FileOutputStream file = new FileOutputStream("file.txt");
            ObjectOutputStream output = new ObjectOutputStream(file);

            // Writing to the file using ObjectOutputStream
            output.writeInt(data1);
            output.writeObject(data2);

            FileInputStream fileStream = new FileInputStream("file.txt");
            // Creating an object input stream
            ObjectInputStream objStream = new ObjectInputStream(fileStream);

            //Using the readInt() method
            System.out.println("Integer data :" + objStream.readInt());

            // Using the readObject() method
            System.out.println("String data: " + objStream.readObject());
```

Output

```
Integer data: 5
String data: This is programiz
```

In the above example, we have used the `readInt()` and `readObject()` method to read integer data and object data from the file.

Here, we have used the `ObjectOutputStream` to write data to the file. We then read the data from the file using the `ObjectInputStream`.

Example 2: Java ObjectInputStream

Let's see another practical example,

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

class Dog implements Serializable {

    String name;
    String breed;

    public Dog(String name, String breed) {
        this.name = name;
        this.breed = breed;
    }
}

class Main {
    public static void main(String[] args) {

        // Creates an object of Dog class
        Dog dog = new Dog("Tyson", "Labrador");

        try {
            FileOutputStream file = new FileOutputStream("file.txt");

            // Creates an ObjectOutputStream
            ObjectOutputStream output = new ObjectOutputStream(file);
```

Output

```
Dog Name: Tyson
Dog Breed: Labrador
```

In the above example, we have created

- `ObjectOutputStream` named `output` using the `FileOutputStream` named `file`
- `ObjectInputStream` named `input` using the `FileInputStream` named `fileStream`
- An object `dog` of the `Dog` class

Here, we have then used the object output stream to write the object to the file. And, the object input stream to read the object from the file.

> **Note**: The `Dog` class implements the `Serializable` interface. It is because the `ObjectOutputStream` only writes the serializable objects to the output stream.

Other Methods Of ObjectInputStream

Methods	Descriptions
`available()`	returns the available number of bytes in the input stream
`mark()`	marks the position in input stream up to which data has been read

`reset()`	returns the control to the point in the input stream where the mark was set
`skipBytes()`	skips and discards the specified bytes from the input stream
`close()`	closes the object input stream