Java I.O

- **Java I/O** (Input and Output) is used to process the input and produce the output based on the input.
- Java uses the concept of stream to make I/O operation fast. The java.io package contains all the classes required for input and output operations.
- We can perform **file handling in java** by java IO API.

Java I.O

Stream

- A stream is a sequence of data. In Java a stream is composed of bytes. It's called a stream because it's like a stream of water that continues to flow.
- In java, 3 streams are created for us automatically. All these streams are attached with console.

  1) System.out: standard output stream

  2) System.in: standard input stream

  3) System.err: standard error stream

Java I.O

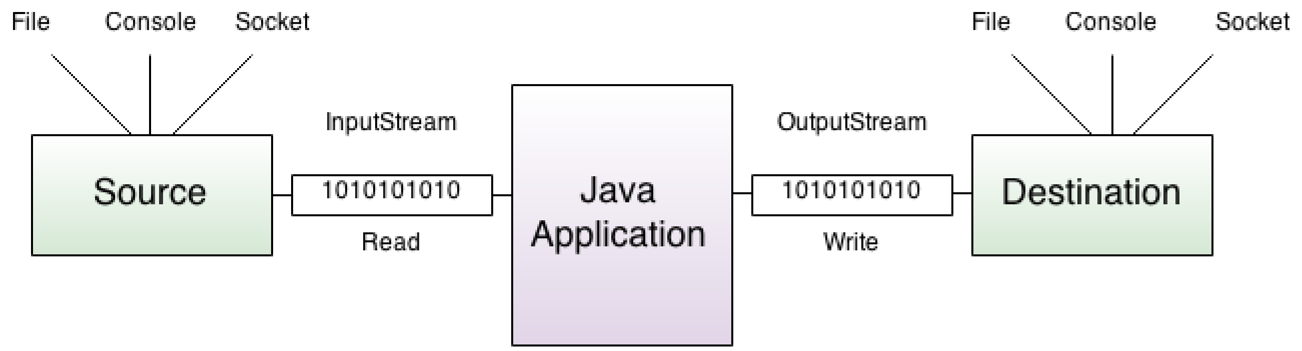
OutputStream

Java application uses an output stream to write data to a destination, it may be a file,an array,peripheral device or socket.

InputStream

Java application uses an input stream to read data from a source, it may be a file,an array,peripheral device or socket.

Java I.O


File
Console
Socket
Source
InputStream
1010101010
Read
Java Application
OutputStream
1010101010
Write
File
Console
Socket
Destination

Java I.O

FileInputStream and FileOutputStream (File Handling)

In Java, FileInputStream and FileOutputStream classes are used to read and write data in file. In another words, they are used for file handling in java.

Java I.O

Java FileOutputStream class

Java FileOutputStream is an output stream for writing data to a file.

```
import java.io.*;
class Test{
  public static void main(String args[]){
   try{
     FileOutputstream fout=new FileOutputStream("abc.txt");
     String s="Sachin Tendulkar is my favourite player";
     byte b[]=s.getBytes();//converting string into byte array
     fout.write(b);
     fout.close();
     System.out.println("success...");
    }catch(Exception e){system.out.println(e);}
  }  }
```

Java I.O

Java FileInputStream class

Java FileInputStream class obtains input bytes from a file.It is used for reading streams of raw bytes such as image data.

It should be used to read byte-oriented data for example to read image, audio, video etc.

Java I.O

```
import java.io.*;
class SimpleRead{
 public static void main(String args[]){
  try{
    FileInputStream fin=new FileInputStream("abc.txt");
    int i=0;
    while((i=fin.read())!=-1){
     System.out.println((char)i);
    }
    fin.close();
  }catch(Exception e){system.out.println(e);}
 }
}
```

Java I.O

Example of Reading the data of current java file and writing it into another file

```
import java.io.*;
class C{
public static void main(String args[])throws Exception{
FileInputStream fin=new FileInputStream("C.java");
FileOutputStream fout=new FileOutputStream("M.java");
int i=0;
while((i=fin.read())!=-1){
fout.write((byte)i);
}
fin.close();
}
}
```

Java I.O

ByteArrayOutputStream

- Java ByteArrayOutputStream class is used to write data into multiple files. In this stream, the data is written into a byte array that can be written to multiple stream.
- The ByteArrayOutputStream holds a copy of data and forwards it to multiple streams.
- The buffer of ByteArrayOutputStream automatically grows according to data.

Java I.O

```
import java.io.*;
class S{
 public static void main(String args[])throws Exception{
  FileOutputStream fout1=new FileOutputStream("f1.txt");
  FileOutputStream fout2=new FileOutputStream("f2.txt");
  ByteArrayOutputStream bout=new ByteArrayOutputStream();
  bout.write(139);
  bout.writeTo(fout1);
  bout.writeTo(fout2);
  bout.flush();
  bout.close();
  System.out.println("success...");
 }  }
```

Java I.O

SequenceInputStream

Java SequenceInputStream class is used to read data from multiple streams. It reads data of streams one by one.

```
import java.io.*;
class Simple{
  public static void main(String args[])throws Exception{
   FileInputStream fin1=new FileInputStream("f1.txt");
   FileInputStream fin2=new FileInputStream("f2.txt");

   SequenceInputStream sis=new SequenceInputStream(fin1,fin2);
```

Java I.O

```
int i;
   while((i=sis.read())!=-1){
    System.out.println((char)i);
   }
   sis.close();
   fin1.close();
   fin2.close();
  }
}
```

Java I.O

Example that reads the data from two files and writes into another file

```
import java.io.*;
class Simple{
  public static void main(String args[])throws Exception{
   FileInputStream fin1=new FileInputStream("f1.txt");
   FileInputStream fin2=new FileInputStream("f2.txt");
   FileOutputStream fout=new FileOutputStream("f3.txt");

   SequenceInputStream sis=new SequenceInputStream(fin1,fin2);
   int i;
   while((i.sisread())!=-1)
   {
     fout.write(i);
   }
```

Java I.O

```
 sis.close();
   fout.close();
   fin1.close();
   fin2.close();

}

}
```

Java I.O

FileWriter and FileReader (File Handling in java)

- Java FileWriter and FileReader classes are used to write and read data from text files. These are character-oriented classes, used for file handling in java.
- Java has suggested not to use the FileInputStream and FileOutputStream classes if you have to read and write the textual information.

Java I.O

FileWriter:

```
import java.io.*;
class Simple{
 public static void main(String args[]){
  try{
   FileWriter fw=new FileWriter("abc.txt");
   fw.write("my name is sachin");
   fw.close();
  }catch(Exception e){System.out.println(e);}
  System.out.println("success");
 }
}
```

Java I.O

FileReader:

```
import java.io.*;
class Simple{
 public static void main(String args[])throws Exception{
  FileReader fr=new FileReader("abc.txt");
  int i;
  while((i=fr.read())!=-1)
  System.out.println((char)i);

  fr.close();
 }
}
```

Java I.O

CharArrayWriter

The CharArrayWriter class can be used to write data to multiple files. This class implements the Appendable interface. Its buffer automatically grows when data is written in this stream. Calling the close() method on this object has no effect.

Java I.O

```
import java.io.*;
class Simple{
 public static void main(String args[])throws Exception{
  CharArrayWriter out=new CharArrayWriter();
  out.write("my name is");
  FileWriter f1=new FileWriter("a.txt");
  FileWriter f2=new FileWriter("b.txt");
  FileWriter f3=new FileWriter("c.txt");
  FileWriter f4=new FileWriter("d.txt");
```

Java I.O

```
out.writeTo(f1);
  out.writeTo(f2);
  out.writeTo(f3);
  out.writeTo(f4);
  f1.close();
  f2.close();
  f3.close();
  f4.close();
 }
}
```

Java I.O

Reading data from keyboard

There are many ways to read data from the keyboard. For example:

1. InputStreamReader
2. Console
3. Scanner

Java I.O

InputStreamReader class

InputStreamReader class can be used to read data from keyboard.It performs two tasks:

- Connects to input stream of keyboard
- Converts the byte-oriented stream into character-oriented stream

Java I.O

Example of reading data from keyboard by InputStreamReader and BufferdReader class

```
import java.io.*;
class G5{
public static void main(String args[])throws Exception{
InputStreamReader r=new InputStreamReader(System.in);
BufferedReader br=new BufferedReader(r);
System.out.println("Enter your name");
String name=br.readLine();
System.out.println("Welcome "+name);
 }
}
```

Java I.O

Console class

- The Java Console class is be used to get input from console. It provides methods to read text and password.
- If you read password using Console class, it will not be displayed to the user.
- The java.io.Console class is attached with system console internally. The Console class is introduced since 1.5.

Java I.O

```
import java.io.*;
class ReadStringTest{
public static void main(String args[]){
Console c=System.console();
System.out.println("Enter your name: ");
String n=c.readLine();
System.out.println("Welcome "+n);
}
}
```

Java I.O

```
import java.io.*;
class ReadPasswordTest{
public static void main(String args[]){
Console c=System.console();
System.out.println("Enter password: ");
char[] ch=c.readPassword();
String pass=String.valueOf(ch);//converting char array into string
System.out.println("Password is: "+pass);
}
}
```