Input/output

Java I/O is a powerful concept, which provides the all input and output operations. Most of the classes of I/O streams are available in java.io package.

Stream

Stream is the logical connection between Java program and file. In Java, stream is basically a sequence of bytes, which has a continuous flow between Java programs and data storage.

Types of Stream

Stream is basically divided into following types based on data flow direction.

Input Stream
Input stream is represented as an input source. It is used to read the binary data from the source.

Output Stream
Output stream represent a destination source. It is basically used to send out/write the data to destination.

Byte Streams
Byte stream is used to input and output to perform 8-bits bytes. It has the classes like **FileInputStream** and **FileOutputStream.**

Character Streams
Character stream basically works on 16 bit-Unicode value convention. This stream is used to read and write data in the format of 16 bit Unicode characters.

Read and Write the Files

In Java, **InputStream** and **Reader classes** have read () method which is used to read the data from a source.
OutputStream and **Writer classes** have write () method which is used to write the data on the destination.

FileInputStream

FileInputStream class is used to read the data from file. It is meant for reading streams of raw byte. The FileInputStream class provides the connection to a disk file.

FileInputStream Constructors

Constructors	Description
FileInputStream(File file)	It creates a FileInputStream by opening a connection to a file.
FileInputStream(FileDescriptor fdobj)	Creates a FileInputStream by using the FileDescriptor object fdobj.
FileInputStream(String name)	Creates a FileInputStream by opening a connection to a file.

FileInputStream Methods

Method	Description
public int available() throw IOException	Returns an approximation of the number of bytes that can be read from this file input stream.
public void close() throws IOException	Close the input stream file and releases the

	system resources.
protected void finalize ()	Make sure that close () method of this input stream is called properly.
public FileChannel getChannel()	Return the unique object of FileChannel.
public final FileDescriptor getFD() throws IOException	Returns the FileDescriptor object that represents the connection to the actual file.
public int read(byte[] b) throws IOException	Read the data from the given input stream in the form of an array of bytes.
public log skip(long n) throws IOException	Skip or discard the n bytes of data from the input stream.

Example: Reading the data from the File using FileInputStream

//abc.txt

ABCabc

//FISDemo.java

```
import java.io.*;
public class FISDemo
{
   public static void main(String args[])throws FileNotFoundException,
IOException
  {
      FileInputStream fi = new FileInputStream("abc.txt");
       int i;
       System.out.println("ASCII value of the character:");
        while((i=fi.read()) != -1)
        {
           System.out.print(i+" : ");
           System.out.println((char)i);
        }
   }
}
```

Output:

```
ASCII value of the character:
65 : A
66 : B
67 : C
97 : a
98 : b
99 : c
```

Example: Reading the data from the File using FileInputStream

```
import java.io.*;
class FileInputStreamDemo
{
   public static void main(String args[])throws IOException
   {
      FileInputStream fis = new
FileInputStream("FileInputStreamDemo.java");
      System.out.println("Available bytes: "+(fis.available()));
      byte arr[] = new byte[50];
      if (fis.read(arr) != 50)
      {
        System.out.println("Could not get 50 bytes");
       }
       System.out.println(new String(arr, 0, 50));
       fis.skip(50);
       if (fis.read(arr) != 50)
       {
          System.out.println("Could not get 50 bytes:");
        }
        System.out.println(new String(arr, 0, 50));
        fis.close();
   }
}
```

FileOutputStream

The FileInputStream class is sub class of OutputStream class. This class is used for writing the data to a File.

FileOutputStream Constructors

Constructors	Descriptions
FileOutputStream (File file)	Creates a file output stream to write to the file by specified file object.
FileOutputStream (File file, boolean append)	It creates file output stream to write to the file, represent by File object.
FileOutputStream (FileDescriptor, fd)	Creates a file input to write to the specified file descriptor.
FileOutputStream (String name)	Creates a file output stream with specified name.

FileOutputStream Methods

Method	Description
public void write(int b) throws IOException	Writes the specified byte in the OutputStream.
public void write(byte[] b) throws IOException	Writes the specified byte in the OutputStream.
public void close() throws IOException	This method is used to close the output stream.
protected void finalize() throws IOException	This method provides the cleanup to the file.
public void getChannel()	It returns the unique FileChannel object associated with this file output stream.

Example: Reading the data from a file and writing it to same file

```
import java.io.*;
public class FileDemo
{
   public static void main(String args[])
   {
      int i;
      FileInputStream fis;
      FileOutputStream fos;
      try
      {
         fis = new FileInputStream(args[0]);
         fos = new FileOutputStream(args[1]);
         do
         {
            i = fis.read();
            if(i!=-1)
               fos.write(i);
          }
         while(i!=-1);
      }
       catch(Exception exp)
      {
       exp.getMessage();
      }
  }
}
```