**\*\* Java Input/Output \*\***

**Java I/O :**

1. **Java I/O** (Input and Output) is used to process the inputandproduce the output.
2. Java uses the concept of a stream to make I/O operation fast.
3. The java.io package contains all the classes required for input and output operations.
4. We can perform **file handling in Java** by Java I/O API.

## Stream :

1. A stream is a sequence of data. In Java, a stream is composed of bytes.
2. It's called a stream because it is like a stream of water that continues to flow.

In Java, 3 streams are created for us automatically. All these streams are attached with the console.

**1) System.out:**standard output stream

**2) System.in:**standard input stream

**3) System.err:**standard error stream

Let's see the code to print **output and an error** message to the console.

### InputStream

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

### OutputStream

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

Java IO

### Useful methods of InputStream

|  |  |  |
| --- | --- | --- |
| **No.** | **Method** | **Description** |
|  | public abstract int read()throws IOException | reads the next byte of data from the input stream. It returns -1 at the end of the file. |
|  | public int available()throws IOException | returns an estimate of the number of bytes that can be read from the current input stream. |
|  | public void close()throws IOException | is used to close the current input stream. |

### InputStream Hierarchy

Java input stream hierarchy

## OutputStream class

1. OutputStream class is an abstract class.
2. It is the superclass of all classes representing an output stream of bytes.
3. An output stream accepts output bytes and sends them to some sink.

### Useful methods of OutputStream

|  |  |  |
| --- | --- | --- |
| **No.** | **Method** | **Description** |
|  | public void write (int) throws IOException | is used to write a byte to the current output stream. |
|  | public void write(byte[]) throws IOException | is used to write an array of byte to the current output stream. |
|  | public void flush() throws IOException | flushes the current output stream. |
|  | public void close() throws IOException | is used to close the current output stream. |

### OutputStream Hierarchy

Java output stream hierarchy

# FileOutputStream Class

Java FileOutputStream is an output stream used for writing data to a [file](https://www.javatpoint.com/java-file-class).

If you have to write primitive values into a file, use FileOutputStream class. You can write byte-oriented as well as character-oriented data through FileOutputStream class. But, for character-oriented data, it is preferred to use [FileWriter](https://www.javatpoint.com/java-filterwriter-class) than FileOutputStream.

# File Class

1. The File class is an abstract representation of file and directory pathname.
2. A pathname can be either absolute or relative.

### [Constructors](https://www.javatpoint.com/java-constructor)

|  |  |  |
| --- | --- | --- |
| **No .** | **Constructor** | **Description** |
|  | File(File parent, String child) | It creates a new File instance from a parent abstract pathname and a child pathname string. |
|  | File(String pathname) | It creates a new File instance by converting the given pathname string into an abstract pathname. |
|  | File(String parent, String child) | It creates a new File instance from a parent pathname string and a child pathname string. |
|  | File(URL) | It creates a new File instance by converting the given file: URI into an abstract pathname. |

### Useful Methods

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Modifier and Type** | **Method** | **Description** |
|  | static File | createTempFile(String  prefix, String suffix) | It creates an empty file in the default temporary-file directory, using the given prefix and suffix to generate its name. |
|  | boolean | createNewFile() | It atomically creates a new, empty file named by this abstract pathname if and only if a file with this name does not yet exist. |
|  | boolean | canWrite() | It tests whether the application can modify the file denoted by this abstract pathname.String[] |
|  | boolean | canExecute() | It tests whether the application can execute the file denoted by this abstract pathname. |
|  | boolean | canRead() | It tests whether the application can read the file denoted by this abstract pathname. |
|  | boolean | isAbsolute() | It tests whether this abstract pathname is absolute. |
|  | | boolean | isDirectory() | It tests whether the file denoted by this abstract pathname is a directory. |
|  | | boolean | isFile() | It tests whether the file denoted by this abstract pathname is a normal file. |
|  | | String | getName() | It returns the name of the file or directory denoted by this abstract pathname. |
|  | | String | getParent() | It returns the pathname string of this abstract pathname's parent, or null if this pathname does not name a parent directory. |
|  | | Path | toPath() | It returns a java.nio.file.Path object constructed from the this abstract path. |
|  | | URI | toURI() | It constructs a file: URI that represents this abstract pathname. |
|  | | File[] | listFiles() | It returns an [array](https://www.javatpoint.com/array-in-java) of abstract pathnames denoting the files in the directory denoted by this abstract pathname |
|  | | long | getFreeSpace() | It returns the number of unallocated bytes in the partition named by this abstract path name. |
|  | | String[] | list(FilenameFilter filter) | It returns an array of strings naming the files and directories in the directory denoted by this abstract pathname that satisfy the specified filter. |
|  | | boolean | mkdir() | It creates the directory named by this abstract pathname. |

**\*\* *File* \*\***

class Q01\_File\_Class

**{**

    public static void main**(** String args**[])**

**{**

File f **=** **new** File**(**"D:\2\_All\_Code\02\_Input\_Output\_File\_Handling(Stream)"**);**

        System**.**out**.**println**();**

**}**

**}**

error: cannot find symbol

                File f = new File("D:\2\_All\_Code\02\_Input\_Output\_File\_Handling(Stream)");

                ^

**import** java**.**io**.\*;**

class Q02\_File\_Class\_length

**{**

    public static void main**(** String args**[])**

**{**

        File f **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo1.txt"**);**

        System**.**out**.**println**(** f**.**length**()** **);**

**}**

**}**

264

.length() return Long type(in Byte).

**import** java**.**io**.\*;**

class Q03\_File\_Class\_exists

**{**

    public static void main**(** String args**[])**

**{**

        File f **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo1.txt"**);**

        System**.**out**.**println**(** f**.**exists**()** **);**

**}**

**}**

true

.exists() return boolean value file exist return true otherwise false.

**import** java**.**io**.\*;**

class Q04\_File\_Class\_isHidden

**{**

    public static void main**(** String args**[])**

**{**

        File f **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo1.txt"**);**

        System**.**out**.**println**(** f**.**isHidden**()** **);**

**}**

**}**

false

.isHidden() return boolean value file is hidden or Not.

If it is hidden Return true otherwise false.

**import** java**.**io**.\*;**

class Q05\_File\_Class\_canWrite

**{**

    public static void main**(** String args**[])**

**{**

        File f **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo1.txt"**);**

        System**.**out**.**println**(** f**.**canWrite**()** **);**

**}**

**}**

true

.canWrite() return boolean value if we can write in file it Return true otherwise false.

**import** java**.**io**.\*;**

class Q06\_File\_Class\_isDirectory

**{**

    public static void main**(** String args**[])**

**{**

        File f1 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo1.txt"**);**

        System**.**out**.**println**(** f1**.**isDirectory**()** **);**

        File f2 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)"**);**

        System**.**out**.**println**(** f2**.**isDirectory**()** **);**

**}**

**}**

true

.isDirectory() return boolean value if File path is file it Return false and when file path is Folder return true.

if file return false.

if folder return true.

**import** java**.**io**.\*;**

class Q07\_File\_Class\_lastModified

**{**

    public static void main**(** String args**[])**

**{**

        File f1 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo1.txt"**);**

        //01

        long time **=** f1**.**lastModified**();**

        System**.**out**.**println**(** time **);**

        // 02

        java**.**util**.**Date date **=** **new** java**.**util**.**Date**(** time **);**

        System**.**out**.**println**(** date **);**

**}**

**}**

// 01

1695983179011

// 02

Fri Sep 29 15:56:19 IST 20231

.lastModified() : return Date when over file is modified.

**import** java**.**io**.\*;**

class Q08\_File\_Class\_list

**{**

    public static void main**(** String args**[])**

**{**

        File f1 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/"**);**

        //01

        String s1**[]** **=** f1**.**list**();**

**for(** String s **:** s1**)**

**{**

            System**.**out**.**println**(** s **);**

**}**

**}**

**}**

Demo1.txt

Input\_Output\_File\_Handling(Stream).docx

Input\_Output\_File\_Handling(Stream).txt

Q01\_File\_Class.java

Q02\_File\_Class\_length.java

Q03\_File\_Class\_exists.java

Q04\_File\_Class\_isHidden.java

Q05\_File\_Class\_canWrite.java

Q06\_File\_Class\_isDirectory.java

Q07\_File\_Class\_lastModified.java

Q08\_File\_Class\_list.class

Q08\_File\_Class\_list.java

.list() : its return String list How many file in Your folder.

**import** java**.**io**.\*;**

class Q09\_File\_Class\_mkdir

**{**

    public static void main**(** String args**[])**

**{**

        File f1 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Vishal Soner"**);**

        boolean b1 **=** f1**.**mkdir**();**

**if(** b1 **)**

**{**

            System**.**out**.**println**(**"Folder Created"**);**

**}**

**else**

**{**

            System**.**out**.**println**(**"Folder Already Exit"**);**

**}**

**}**

**}**

// 01  run      true

Folder Created

// 02 run       false.

Folder Already Exit

D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/(Vishal Soner)

Vishal Soner Folder  :  1st time folder create return true, 2nd time return false,

.mkdir() : its return Boolean. Folder Created return true, otherwise return false.

Use to Create Folder.

**import** java**.**io**.\*;**

class Q10\_File\_Class\_delete

**{**

    public static void main**(** String args**[])**

**{**

        File f1 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Vishal Soner"**);**

        boolean b1 **=** f1**.**delete**();**

**if(** b1 **)**

**{**

            System**.**out**.**println**(**"Folder Deleted"**);**

**}**

**else**

**{**

            System**.**out**.**println**(**"Folder Already Deleted"**);**

**}**

**}**

**}**

// 01  run      true

Folder Deleted

// 02 run       false.

Folder Already Deleted

D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/(Vishal Soner)

Vishal Soner Folder  :  1st time folder Deleted return true, 2nd time return false,

.delete() : its return Boolean. Folder Deleted return true, otherwise return false.

Use to Deleted the Folder.

**import** java**.**io**.\*;**

class Q11\_File\_Class\_renameTo

**{**

    public static void main**(** String args**[])**

**{**

        File f1 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Vishal Soner"**);**

        File f2 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Vishal"**);**

        boolean b1 **=** f1**.**renameTo**(**f2**);**

**if(** b1 **)**

**{**

            System**.**out**.**println**(**"Folder Name Changed"**);**

**}**

**else**

**{**

            System**.**out**.**println**(**"Folder Name Already Changed"**);**

**}**

**}**

**}**

// 01  run      true

Folder Name Changed

// 02 run       false.

Folder Name Already Changed

D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/(Vishal Soner)

Vishal Soner Folder  :  1st time folder rename ho gya to return true, 2nd time return false,

.renameTo() : its return Boolean. Folder Renamed return true, otherwise return false.

Use to rename the Folder.

**import** java**.**io**.\*;**

class Q12\_File\_Class\_renameTo

**{**

    public static void main**(** String args**[])**

**{**

        File f1 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/A.Vishal Soner"**);**

        File f2 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Vishal Soner"**);**

        boolean b1 **=** f1**.**renameTo**(**f2**);**

**if(** b1 **)**

**{**

            System**.**out**.**println**(**"Folder Name Changed"**);**

**}**

**else**

**{**

            System**.**out**.**println**(**"Folder Name Already Changed"**);**

**}**

**}**

**}**

// 01  run      true

Folder Name Changed

// 02 run       false.

Folder Name Already Changed

"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/(A.Vishal Soner");

"E:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Vishal Soner");

Ye Folder ka Name change kar ke Directry v cahange kar degi.( D se E me dal Dega).

**import** java**.**io**.\*;**

class Q13\_File\_Class\_canWrite

**{**

    public static void main**(** String args**[])**

**{**

        File f1 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Vishal Soner"**);**

        boolean b1 **=** f1**.**canWrite**();**

**if(** b1 **)**

**{**

            System**.**out**.**println**(**"Yes "**);**

**}**

**else**

**{**

            System**.**out**.**println**(**"No"**);**

**}**

**}**

**}**

Yes

.canWrite() : its return Boolean. when we can write in file/folder return true otherwise return false.

Use to check we can write or not.

**import** java**.**io**.\*;**

class Q14\_File\_Class\_canRead

**{**

    public static void main**(** String args**[])**

**{**

        File f1 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Vishal Soner"**);**

        boolean b1 **=** f1**.**canRead**();**

**if(** b1 **)**

**{**

            System**.**out**.**println**(**"Yes "**);**

**}**

**else**

**{**

            System**.**out**.**println**(**"No"**);**

**}**

**}**

**}**

Yes

.canRead() : its return Boolean. when we can Read in file/folder return true,  otherwise return false.

Use to check we can write or not.

**import** java**.**io**.\*;**

class Q15\_File\_Class\_isFile

**{**

    public static void main**(** String args**[])**

**{**

        File f1 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Vishal Soner"**);**

        boolean b1 **=** f1**.**isFile**();**

**if(** b1 **)**

            System**.**out**.**println**(**"Yes "**);**

**else**

            System**.**out**.**println**(**"No"**);**

        File f2 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo1.txt"**);**

        boolean b2 **=** f2**.**isFile**();**

**if(** b2 **)**

            System**.**out**.**println**(**"Yes "**);**

**else**

            System**.**out**.**println**(**"No"**);**

**}**

**}**

No

Yes

.isFile() : its return Boolean. To check the file/folder  if path set data is file return true,  otherwise return false.

Use to check the file or not.

**import** java**.**io**.\*;**

class Q16\_File\_Class\_createNewFile

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        File f1 **=** **new** File**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        boolean b1 **=** f1**.**createNewFile**();**

**if(** b1 **)**

            System**.**out**.**println**(**"Yes "**);**

**else**

            System**.**out**.**println**(**"No"**);**

**}**

**}**

// 01 1st time Run

Yes

// 02 2st time Run

No

.createNewFile() : its return Boolean. To Create the file/folder  if path set data file not created then

                   Create a file return true,  if already avilable then return false.

Use to check file is Created or not.

**\*\* FileInputStream \*\***

**import** java**.**io**.\*;**

class Q17\_FileInputSteam\_read

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        FileInputStream fs  **=**  **new** FileInputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo1.txt"**);**

**while(** **true** **)**

**{**

            int n **=** fs**.**read**();**

            System**.**out**.**print**(** n **+** "  " **);**

**}**

**}**

**}**

-1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  infinity

Step - 01 : Open the steram.

FileInputStream fs = new FileInputStream("Path");

Step - 02 : read the data.

fs.read();

Step - 03 : Close the steram.

fs.close();

**import** java**.**io**.\*;**

class Q18\_FileInputSteam\_read

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        FileInputStream fs  **=**  **new** FileInputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo1.txt"**);**

**while(** **true** **)**

**{**

            int n **=** fs**.**read**();**

**if(-**1 **==** n**)**

**break;**

            System**.**out**.**print**(** n **+** "  " **);**

**}**

        fs**.**close**();**

**}**

**}**

86   105  115  104  97  108  32  83  111  110  101  114  32   45  32   48  49   13   10   86  105  115  104  97  108  32  83  111  110 101  114  32   45   32  48

50  13  10   86   105  115  104  97  108  32  83  111  110  101  114

Step - 01 : Open the steram.

FileInputStream fs = new FileInputStream("Path");

Step - 02 : read the data.

fs.read();

Step - 03 : Close the steram.

fs.close();

**import** java**.**io**.\*;**

class Q19\_FileInputSteam\_read

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        FileInputStream fs  **=**  **new** FileInputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo1.txt"**);**

**while(** **true** **)**

**{**

            int n **=** fs**.**read**();**

**if(-**1 **==** n**)**

**break;**

            System**.**out**.**print**(** **(**char**)**n**);**

**}**

        fs**.**close**();**

**}**

**}**

Vishal Soner - 01

Vishal Soner - 02

Vishal Soner - 03

Vishal Soner - 04

Vishal Soner - 05

Vishal Soner - 06

Vishal Soner - 07

Vishal Soner - 08

Vishal Soner - 09

**import** java**.**io**.\*;**

class Q20\_FileInputSteam\_read

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        FileInputStream fs  **=**  **new** FileInputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo1.txt"**);**

        byte b**[]** **=** **new** byte**[**10**];**

        fs**.**read**(**b**);**

**for(** byte i **:** b**)**

**{**

            System**.**out**.**print**(** i **+** "  " **);**

**}**

        fs**.**close**();**

**}**

**}**

86  105  115  104  97  108  32  83  111  110

**import** java**.**io**.\*;**

class Q21\_FileInputSteam\_read

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        FileInputStream fs  **=**  **new** FileInputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo1.txt"**);**

        byte b**[]** **=** **new** byte**[**200**];**

        fs**.**read**(**b**);**

        String s **=** **new** String**(**b**);**

        System**.**out**.**print**(** s **);**

        fs**.**close**();**

**}**

**}**

Vishal Soner - 01

Vishal Soner - 02

Vishal Soner - 03

Vishal Soner - 04

Vishal Soner - 05

**import** java**.**io**.\*;**

class Q22\_FileInputSteam\_available

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        FileInputStream fs  **=**  **new** FileInputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo1.txt"**);**

        int len **=** fs**.**available**();**

        byte b**[]** **=** **new** byte**[**len**];**

        fs**.**read**(**b**);**

        String s **=** **new** String**(**b**);**

        System**.**out**.**print**(** s **);**

        fs**.**close**();**

**}**

**}**

Vishal Soner - 01

Vishal Soner - 02

Vishal Soner - 03

Vishal Soner - 04

Vishal Soner - 05

**.**write**();**           **:**    Add All

**.**write**(** byte**[]** **);**   **:**    Add All

**.**write**(** byte**[]** **,** int start**,** int end**);** Add

    For Reading  : the file must exists.

    For Writeing : 1. if file is available then the contents you are writing will overwrite the existing data.

                   2. if file is not available so new file created And then data will be store to new file.

**import** java**.**io**.\*;**

class Q23\_FileOutputSteam\_write

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        FileOutputStream  fs  **=**  **new** FileOutputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        fs**.**write**(**97**);**

        fs**.**write**(**98**);**

        fs**.**write**(**99**);**

        fs**.**write**(**100**);**

        fs**.**write**(**101**);**

        fs**.**write**(**102**);**

        fs**.**close**();**

        FileInputStream fi **=** **new** FileInputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        byte b**[]** **=** **new** byte**[**fi**.**available**()];**

        fi**.**read**(**b**);**

        String s **=** **new** String**(**b**);**

        System**.**out**.**println**(** s **);**

        System**.**out**.**println**(** "Done" **);**

        fi**.**close**();**

**}**

**}**

abcd

...FileOutputStream("path");

Parameter me true nai denge to data overRide ho jayega.(All Remove new Insert).

...FileOutputStream("path", true);

Parameter me true denge to data store(concat) ho jayega.(Without delete any data).

.write(data);

Step - 01 : Open the steram.

FileOutputStream fs = new FileOutputStream("Path");

Step - 02 : Write the data.

fs.write();

Step - 03 : Close the steram.

fs.close();

**import** java**.**io**.\*;**

class Q24\_FileOutputSteam\_write

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        FileOutputStream  fs  **=**  **new** FileOutputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**,** **true);**

        fs**.**write**(**97**);**

        fs**.**write**(**98**);**

        fs**.**write**(**99**);**

        fs**.**write**(**100**);**

        fs**.**write**(**101**);**

        fs**.**write**(**102**);**

        fs**.**close**();**

        FileInputStream fi **=** **new** FileInputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        byte b**[]** **=** **new** byte**[**fi**.**available**()];**

        fi**.**read**(**b**);**

        String s **=** **new** String**(**b**);**

        System**.**out**.**println**(** s **);**

        System**.**out**.**println**(** "Done" **);**

        fi**.**close**();**

**}**

**}**

// 1st Run

abcdef

Done

// 2st Run

abcdefabcdef

Done

// 3st Run

abcdefabcdefabcdef

Done

// 4st Run

abcdefabcdefabcdefabcdef

Done

// 5st Run

abcdefabcdefabcdefabcdefabcdef

Done

...FileOutputStream("path");

Parameter me true nai denge to data overRide ho jayega.(All Remove new Insert).

...FileOutputStream("path", true);

Parameter me true denge to data store(concat) ho jayega.(Without delete any data).

.write(data);

**import** java**.**io**.\*;**

class Q25\_FileOutputSteam\_write

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        FileOutputStream  fs  **=**  **new** FileOutputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        String s1 **=** "Hello Vishal Soner, This is my Account"**;**

        byte b1**[]** **=** s1**.**getBytes**();**

        fs**.**write**(**b1**);**

        fs**.**close**();**

        FileInputStream fi **=** **new** FileInputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        byte b2**[]** **=** **new** byte**[**fi**.**available**()];**

        fi**.**read**(**b2**);**

        String s2 **=** **new** String**(**b2**);**

        System**.**out**.**println**(** s2 **);**

        System**.**out**.**println**(** "Done" **);**

        fi**.**close**();**

**}**

**}**

Hello Vishal Soner, This is my Account

Done

**import** java**.**io**.\*;**

class Q26\_FileOutputSteam\_write

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        FileOutputStream  fs  **=**  **new** FileOutputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        String s1 **=** "Hello Vishal Soner, This is my Account"**;**

        byte b1**[]** **=** s1**.**getBytes**();**

        fs**.**write**(**b1**,** 6**,** 10**);**

        fs**.**close**();**

        FileInputStream fi **=** **new** FileInputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        byte b2**[]** **=** **new** byte**[**fi**.**available**()];**

        fi**.**read**(**b2**);**

        String s2 **=** **new** String**(**b2**);**

        System**.**out**.**println**(** s2 **);**

        System**.**out**.**println**(** "Done" **);**

        fi**.**close**();**

**}**

**}**

Vishal Son

Done

**\*\* File-Writer \*\***

**import** java**.**io**.\*;**

class Q27\_FileWriter

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        // FileWriter fw = new FileWriter("D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt");

        FileWriter fw **=** **new** FileWriter**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**,** **true);**

        fw**.**write**(**"Hello Vishal Soner\n"**);**

        fw**.**write**(**"Writing the Duplicate Data"**);**

        fw**.**close**();**

        System**.**out**.**println**(**"Done"**);**

**}**

**}**

Done

Ye Data ko Insert ke karega + Old Data ko delete kar dega.

FileWriter fw = new FileWriter("D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt",  true);

Yadi Parameter me true pass kar de to Old data Delete nai hoga + Data insert ho jayega.

**\*\* File-Reader \*\***

**import** java**.**io**.\*;**

class Q28\_FileReader

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        FileReader fr **=** **new** FileReader**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        char c**[]** **=** **new** char**[**30**];**

        fr**.**read**(**c**);**

**for(** char c1 **:** c**)**

**{**

            System**.**out**.**print**(**c1**);**

**}**

        fr**.**close**();**

**}**

**}**

Hello Vishal Soner

Writing the

Ye Data ko Read ke Liye use karte he.

**import** java**.**io**.\*;**

class Q29\_FileReader

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        FileReader fr **=** **new** FileReader**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

**while(** **true** **)**

**{**

            int n **=** fr**.**read**();**

**if(**n **==** **-**1**)**

**break;**

            System**.**out**.**print**(** **(**char**)**n **);**

**}**

        fr**.**close**();**

**}**

**}**

Hello Vishal Soner

Writing the Duplicate Data

To read / write the **Primitive** Data We use ,

**DataInputStream**   :  is given for reading Primitive data.

**DataOutputStream**  :  is given for Writing Primitive data.

To read / write  the **Non** **-** **Primitive** Object We use ,

**ObjectInputStream**   :    is given for reading non-Primitive Object from file.

**ObjectOutputStream**  :    is given for writing non-Primitive Object from file.

DataOutputStream  :  To Write the primitive Data into file.

1. **void writeByte( byte );**
2. **void writeShort( short );**
3. **void writeInt( int );**
4. **void writeLong( long );**
5. **void writeFloat( float );**
6. **void writeDouble( double );**
7. **void writeBoolean(boolean);**

**DataOutputStream** Se data insert to ho jayega but ham eshe file se directly access nai kar sakte he

To read the Data From File.

**DataInputStream** :

1. **readByte( byte );**
2. **readShort( short  );**
3. **readInt ( int );**
4. **readLong( long );**
5. **readFloat( float );**
6. **readDouble( double );**
7. **readBoolean( boolean );**

**\*\* Data-Output-Stream\*\***

**import** java**.**io**.\*;**

class Q30\_DataOutputStream

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        int x **=** 100**;**

        float y **=** 10.50f**;**

        boolean tr **=** **true;**

        FileOutputStream fs **=** **new** FileOutputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        //Store Primitive Type Data Using FileOutputStream (Error)

        fs**.**write**(**x**);**

        fs**.**write**(**y**);**

        fs**.**write**(**tr**);**

        fs**.**close**();**

**}**

**}**

error: no suitable method found for write(float)

    fs.write(y);

      ^

error: no suitable method found for write(boolean)

    fs.write(tr);

      ^

DataOutputStream ds **=** **new** DataOutputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt")**;**

**Error :**

**import** java**.**io**.\*;**

class Q31\_DataOutputStream

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        int x **=** 100**;**

        float y **=** 10.50f**;**

        Boolean tr **=** **true;**

        FileOutputStream fs **=** **new** FileOutputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        //Store Primitive Type Data Using FileOutputStream

        DataOutputStream ds **=** **new** DataOutputStream**(**fs**);**

        ds**.**writeInt**(**x**);**

        ds**.**writeFloat**(**y**);**

        ds**.**writeBoolean**(**tr**);**

        ds**.**close**();**

        fs**.**close**();**

        System**.**out**.**println**(**"Done"**);**

**}**

**}**

Done

**DataOutputStream** Se data insert to ho jayega but ham eshe file se directly access nai kar sakte he

**\*\* Data-Input-Stream\*\***

**import** java**.**io**.\*;**

class Q33\_DataInputStream

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        int x **=** 100**;**

        float y **=** 10.50f**;**

        boolean tr **=** **true;**

        FileInputStream fs **=** **new** FileInputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        //Read Primitive Type Data Using FileInputStream.

        DataInputStream ds **=** **new** DataInputStream**(**fs**);**

        // char c[] = new char[ ds.available() ];

        byte c**[]** **=** **new** byte**[** ds**.**available**()** **];**

        ds**.**readFully**(**c**);**

**for(** byte c1 **:** c**)**

**{**

            System**.**out**.**println**(** c1 **);**

**}**

        ds**.**close**();**

        fs**.**close**();**

**}**

**}**

100

10.5

true

DataOutputStream Se data insert to ho jayega but ham eshe file se directly access nai kar sakte he

All Object Are Store in Heap Memory.

Non-Primitive Class ka use Object ko ek file me Store karne ke liye karte he.

--------------------------------------------------------------------------------------------------------

Serialization : Preserving State of an Objects. (ObjectOutputStream)

                Serialization is a mechanism of writing the state of an object into a byte-stream. It is mainly used in Hibernate, JPA technologies.

De-serialization : Reading State of an Objects. (ObjectInputStream)

                    The reverse operation of serialization is called de-serialization where byte-stream is converted into an object.

                    The serialization and deserialization process is platform-independent,

                    it means you can serialize an object on one platform and deserialize it on a different platform.

--------------------------------------------------------------------------------------------------------

1. if you want to save the Object of a Class, you need to mark it as serialize, That

2. we can do by implement (java.io.serializable interface) the Serializable interface for serializing the object.

--------------------------------------------------------------------------------------------------------

For serializing the object, we call the writeObject() method of ObjectOutputStream class,  and

For deserialization we call the readObject() method of ObjectInputStream class.

--------------------------------------------------------------------------------------------------------

java.io.Serializable interface

1. Serializable is a marker interface (has no data member and method).

2. It is used to "mark" Java classes so that the objects of these classes may get a certain capability.

3. The Cloneable and Remote are also marker interfaces.

The Serializable interface must be implemented by the class whose object needs to be persisted.

The String class and all the wrapper classes implement the java.io.Serializable interface by default.

**(** **\*\*** Folder**/**pkg **:** Game**.**java **)**

package pkg**;**

**import** java**.**io**.\*;**

**import** java**.**util**.**Scanner**;**

public class Game implements Serializable

**{**

    private int id**;**

    private String name**;**

    private int age**;**

    private long contact**;**

    public void play**()**

**{**

        Scanner sc **=** **new** Scanner**(**System**.**in**);**

        System**.**out**.**print**(**"Enter Id : "**);**

        id **=** sc**.**nextInt**();**

        System**.**out**.**print**(**"Enter Name : "**);**

        name **=** sc**.**next**();**

        System**.**out**.**print**(**"Enter Age : "**);**

        age **=** sc**.**nextInt**();**

        System**.**out**.**print**(**"Enter Contact No. : "**);**

        contact **=** sc**.**nextLong**();**

        display\_Data**();**

**}**

    public void display\_Data**()**

**{**

        System**.**out**.**println**(**"-------------------"**);**

        System**.**out**.**println**(**"| Id      : " **+** id      **+** "    |"**);**

        System**.**out**.**println**(**"| Name    : " **+** name    **+** "    |"**);**

        System**.**out**.**println**(**"| Age     : " **+** age     **+** "    |"**);**

        System**.**out**.**println**(**"| Contact : " **+** contact **+** "    |" **);**

        System**.**out**.**println**(**"-------------------"**);**

**}**

**}**

**import** java**.**io**.\*;**

**import** java**.**util**.\*;**

class Q36\_ObjectOutputStream

**{**

    public static void main**(** String args**[])** throws IOException

**{**

        pkg**.**Game g1 **=** **new** pkg**.**Game**();**

        g1**.**play**();**

        FileOutputStream fs **=** **new** FileOutputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        ObjectOutputStream os **=** **new** ObjectOutputStream**(**fs**);**

        os**.**writeObject**(**g1**);**

        System**.**out**.**println**(**"Stored"**);**

        fs**.**close**();**

        os**.**close**();**

**}**

**}**

Stored

**import** java**.**io**.\*;**

class Q37\_ObjectInputStream

**{**

    public static void main**(** String args**[])** throws Exception

**{**

        FileInputStream fs1 **=** **new** FileInputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        ObjectInputStream os1 **=** **new** ObjectInputStream**(**fs1**);**

        pkg**.**Game g2 **=** **(**pkg**.**Game**)**os1**.**readObject**();**

        g2**.**display\_Data**();**

        fs1**.**close**();**

        os1**.**close**();**

**}**

**}**

// 1st Time Compile :

-------------------

| Id      : 10 |

| Name   : 20 |

| Age     : 30  |

| Contac : 40  |

-------------------

**import** java**.**io**.\*;**

class Q38\_ObjectInputStream

**{**

    public static void main**(** String args**[])** throws Exception

**{**

        FileInputStream fs1 **=** **new** FileInputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        ObjectInputStream os1 **=** **new** ObjectInputStream**(**fs1**);**

        pkg**.**Game g2 **=** **(**pkg**.**Game**)**os1**.**readObject**();**

        g2**.**display\_Data**();**

        fs1**.**close**();**

        os1**.**close**();**

        pkg**.**Game g1 **=** **new** pkg**.**Game**();**

        g1**.**play**();**

**//Here Display and New Data Added After Display  (new Data insert Also).**

        FileOutputStream fs2 **=** **new** FileOutputStream**(**"D:/2\_All\_Code/02\_Input\_Output\_File\_Handling(Stream)/Demo2.txt"**);**

        ObjectOutputStream os2 **=** **new** ObjectOutputStream**(**fs2**);**

        os2**.**writeObject**(**g1**);**

        System**.**out**.**println**(**"Stored"**);**

        fs2**.**close**();**

        os2**.**close**();**

**}**

**}**

// 1st Time Compile :

-------------------

| Id      : 10    |

| Name    : 20    |

| Age     : 30    |

| Contact : 40    |

-------------------

Enter Id : 50

Enter Name : 60

Enter Age : 70

Enter Contact No. : 80

-------------------

| Id      : 50    |

| Name    : 60    |

| Age     : 70    |

| Contact : 80    |

-------------------

Stored

// 2st Time Compile :

D:\2\_All\_Code\02\_Input\_Output\_File\_Handling(Stream)>java Q37\_ObjectInputStream

-------------------

| Id      : 50    |

| Name    : 60    |

| Age     : 70    |

| Contact : 80    |

-------------------

Enter Id : 101

Enter Name : 102

Enter Age : 103

Enter Contact No. : 104

-------------------

| Id      : 101    |

| Name    : 102    |

| Age     : 103    |

| Contact : 104    |

-------------------

------------------------------------------------------