How to get input in java?

1. Using Input Stream Reader:

It will take input as String only ,which we need to cast:

InputStreamReader i=**new** InputStreamReader(System.***in***);

BufferedReader bu=**new** BufferedReader(i);

**float** s=Float.*parseFloat*(bu.readLine());

1. Using scanner:

We can give functions for input nextLine(),nextInt()

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

String i=sc.nextLine();

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

1. Using System.in:

It will print in asci format

int i=System.in.read();

System.out.println((char)i);

//to read a string

String str="";

while((i=System.in.read())!=48)

{

str=str+(char)i;

}

System.out.println(str);

Copying a value:

There are 3 ways to copy a value.

Shallow copy:

Both the reference will copy the same address,change in one of the reference will affect the other.

**class** Abc

{

**int** i;

**int** j;

@Override

**public** String toString() {

**return** "Abc [i=" + i + ", j=" + j + "]";

}

}

**public** **class** Shallowcopy {

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

{

Abc obj1=**new** Abc();

obj1.i=5;

obj1.j=10;

Abc obj2;

obj2=obj1;

System.***out***.println("obj2"+obj2);

System.***out***.println("obj1"+obj1);

//change obj1 value

obj1.i=20;

System.***out***.println("change obj1 value");

System.***out***.println("obj2"+obj2);

System.***out***.println("obj1"+obj1);

//change obj2 value

obj2.i=2000;

System.***out***.println("change obj2 value");

System.***out***.println("obj2"+obj2);

System.***out***.println("obj1"+obj1);

}

}

o/p:

obj2Abc [i=5, j=10]

obj1Abc [i=5, j=10]

change obj1 value

obj2Abc [i=20, j=10]

obj1Abc [i=20, j=10]

change obj2 value

obj2Abc [i=2000, j=10]

obj1Abc [i=2000, j=10]

Deep Copy:

We will copy each value,change in 1 reference will not affect the other

Abc obj1=**new** Abc();

obj1.i=5;

obj1.j=10;

Abc obj2=**new** Abc();

obj2.i=obj1.i;

obj2.j=obj1.j;

System.***out***.println("obj2"+obj2);

System.***out***.println("obj1"+obj1);

//change obj1 value

obj1.i=20;

System.***out***.println("change obj1 value");

System.***out***.println("obj2"+obj2);

System.***out***.println("obj1"+obj1);

//change obj2 value

obj2.i=2000;

System.***out***.println("change obj2 value");

System.***out***.println("obj2"+obj2);

System.***out***.println("obj1"+obj1);

o/p:

obj2Abc [i=5, j=10]

obj1Abc [i=5, j=10]

change obj1 value

obj2Abc [i=5, j=10]

obj1Abc [i=20, j=10]

change obj2 value

obj2Abc [i=2000, j=10]

obj1Abc [i=20, j=10]

cloneable:

It is a marker interface to give permission for clone method.The clone is a protected method which does shallow copy

**class** bcd **implements** Cloneable

{

**int** i;

**int** j;

@Override

**public** String toString() {

**return** "bcd [i=" + i + ", j=" + j + "]";

}

**public** Object clone() **throws** CloneNotSupportedException

{

**return** **super**.clone();

}

}

**public** **class** Clone {

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

{

bcd obj1=**new** bcd();

obj1.i=5;

obj1.j=6;

bcd obj2=**new** bcd();

obj2=(bcd)obj1.clone();

System.***out***.println("obj1:"+obj1);

System.***out***.println("obj2"+obj2);

obj2.i=20;

System.***out***.println("after change in obj2");

System.***out***.println("obj1:"+obj1);

System.***out***.println("obj2"+obj2);

}

}

To write to a file:

1. Using File Input stream

FileInputStream fi=**new** FileInputStream(F);

DataInputStream dis=**new** DataInputStream(fi);

String str=dis.readUTF();

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

1. Using File Writeer

  FileWriter fw=new FileWriter("text");

        // read character wise from string and write

        // into FileWriter

        for (int i = 0; i < str.length(); i++)

            fw.write(str.charAt(i));

Property file:

To write or read:

Properties pr=**new** Properties();

OutputStream os=**new** FileOutputStream("prop.properties");

//set the properties

pr.setProperty("name", "uma");

pr.setProperty("passwordd", "131191");

pr.setProperty("state", "tn");

//good is comment and store is used to pust teh value

pr.store(os, "good");

//Read teh properties file

Properties pr1=**new** Properties();

InputStream is=**new** FileInputStream("prop.properties");

pr1.load(is);

System.***out***.println(pr1.getProperty("name"));

pr1.list(System.***out***);//to list the properties

Different ways to read a file:

1. Using BufferedReader:
   * provides buffering of data for fast reading

BufferedReader br = new BufferedReader(new FileReader(file));

String st;

while ((st = br.readLine()) != null)

System.out.println(st);

1. Using FileReader class(use file name and file class)
   * reading character files.(need to cast to char or asci value will be printed

FileReader fr =

new FileReader("C:\\Users\\pankaj\\Desktop\\test.txt");

int i;

while ((i=fr.read()) != -1)

System.out.print((char) i);

1. Using Scanner class:

* Scanner provides parsing ability.
* its input into tokens using a delimiter pattern, which by default matches whitespace.

Scanner sc = new Scanner(file);

while (sc.hasNextLine())

System.out.println(sc.nextLine());

1. Using Scanner class but without using loops( uses Fileclass)

Scanner sc = new Scanner(file);

// we just need to use \\Z as delimiter

sc.useDelimiter("\\Z");

System.out.println(sc.next());

1. Reading the whole file in a List(uses name of file)

List<String> lines = Collections.emptyList();

try

{

lines =

Files.readAllLines(Paths.get(“text”), StandardCharsets.UTF\_8); //where text is name of the file

}

1. Using file output Stream:

java.io.File F=**new** java.io.File("demo.txt");

FileOutputStream fo=**new** FileOutputStream(F);

DataOutputStream dos=**new** DataOutputStream(fo);

dos.writeUTF("hel");

Changing file permissions in java:

We can change file permissions using the below and they return boolean result based on sucess

* setExecutable
* setReadable
* setWritable

  File file = new File("C:\\Users\\Mayank\\Desktop\\1.txt");

  file.setExecutable(true);

  file.setReadable(true);

  file.setWritable(false);

File Misclanneous:

     //create File object with dirpath and dname

        File f = new File(dirpath, dname);

//get the contents into arr[]

            //now arr[i] represent either a File or Directory

            String arr[]=f.list();

Moving a file from one directory to another using Java

1. Using Files.Path move() method:

**public static Path move(Path source, Path target, CopyOption..options)**

**throws IOException**

**Parameters:**

**source** - the path to the file to move

**target** - the path to the target file

(may be associated with a different provider to the source path)

**options** - options specifying how the move should be done

**Returns:** the path to the target file

Path temp = Files.*move*

(Paths.*get*("D:\\File temp\\first.txt"),

Paths.*get*("D:\\practice java\\practise\\first1.txt"));

1. Using Java.io.File.renameTo() and Java.io.File.delete() methods:

Copying the file and deleting the original file using these two methods.  
**Syntax of renameTo():**

**public boolean renameTo(File dest)**

**Description:** Renames the file denoted by this abstract path name.

**Parameters:** dest - The new abstract path name for the named file

**Returns:** true if and only if the renaming succeeded; false otherwise

**Syntax of delete():**

**public boolean delete()**

**Description:** Deletes the file or directory

denoted by this abstract path name.

**Returns:** true if and only if the file or

directory is successfully deleted; false otherwise

 File file = new File("C:\\Users\\Mayank\\Desktop\\1.txt");

        // renaming the file and moving it to a new location

        if(file.renameTo

           (new File("C:\\Users\\Mayank\\Desktop\\dest\\newFile.txt")))

        {

            // if file copied successfully then delete the original file

            file.delete();

            System.out.println("File moved successfully");

        }

To append contents to a file:

BufferedWriter out = new BufferedWriter( new FileWriter(file, true));

out.write("hello");

out.close();

Copying a file using java

FileInputStream fis = **new** FileInputStream("text");

/\* assuming that the file exists and need not to be

checked \*/

FileOutputStream fos = **new** FileOutputStream("text1");

**int** b;

**while** ((b=fis.read()) != -1)

fos.write(b);

Serialization:

We need to convert the file to object and object to file we should use marker interface Serializable

**class** save **implements** Serializable

{

**int** i;

}

**public** **class** serialdemo {

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

// **TODO** Auto-generated method stub

save obj1=**new** save();

obj1.i=23;

java.io.File F=**new** java.io.File("demo.txt");

FileOutputStream fo=**new** FileOutputStream(F);

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

os.writeObject(obj1);

FileInputStream fi=**new** FileInputStream(F);

ObjectInputStream dis=**new** ObjectInputStream(fi);

save obj2=(save)dis.readObject();

System.***out***.println(obj2.i);

}

}