- import java.io.\*;

- file related things available here

- File f = new File(“abc.txt”);

- it is used to create file object reference

- f.exists();

- it gives true if file exists or gives false if file not available

- f.createNewFile();

- it is used to create new file if file not available

- java file i/o concept implemented based on unix operating system

- java file object used for directories also

- File f = new File(“practice”);

- it is used to create directory object reference

- f.mkdir();

- It is used to create new directory if not available

- File f = new File(“practice”, “abc.txt”);

Or

File f = new File(“C:\\Users\\siva”,”abc.txt”);

Or

File f = new File(f1,”abc.txt”); where f1 is File f1=new File(“practice”);

- it is used to create file object reference in particular subfolder

- f.isFile();

- it is used to check pointing to file or not

- f.isDirectory();

- it is used to check pointing to directory or not

- f.list()

- it is used to list all files and directories present in current working directory

- ex: String[] s= f.list();

- f.length();

- it is used to find no.of characters in a file

- long l= f.length();

- f.delete();

- it is used to delete file or directory

**Writing data into file**

- FileWriter fw = new FileWriter(“abc.txt”);

- it creates filewriter object reference

- if we do this then data will be overrides the data in file

- if specified file not available then it creates that file

- FileWriter fw = new FileWriter(f);

- it creates filewriter object reference

- if we do this then data will be overrides the data in file

- if specified file not available then it creates that file

- FileWriter fw = new FileWriter(“abc.txt”, true);

- it creates filewriter object reference

- if we do this then data will be appends to data in file

- if specified file not available then it creates that file

- FileWriter fw = new FileWriter(f, true);

- it creates filewriter object reference

- if we do this then data will be appends to data in file

- if specified file not available then it creates that file

- fw.write(‘s’); or fw.write(char c);

- this method used to write single character into a file

- we can also give integer value as input, it adds corresponding character value

- fw.write(char[ ] ch);

- this method used to write character array into a file

- fw.write(“siva”); or fw.write(String s);

- this method used to write string into a file

- flush() method

- it gives guarantee that all given data is added into file

- after .write() method it is highly recommended to use this method

- close() method

- it is used to close the writer after completing

**Read data from file**

- FileReader fr = new FileReader(“abc.txt”);

- it is used to create filereader oject

- FileReader fr = new FileReader(f);

- fr.read();

- it is used to read data from file

- it gives Unicode value of character after we need to convert Unicode value to char value. It give one Unicode value at a time

- if it reach end of file then it give -1 as output

- fr.read(char[ ] ch);

- it is used to read character array

- fr.close();

- it is used to close

**Bufferedwriter**

- it don’t communicate with file directly. It can communicate via writer object only

- it is same as FileWriter except in case of newline printing

- in FileWriter for new line we use \n .but, in BufferedReader for new line we use .newLine() method

- ex: BufferedWriter bw = new BufferedWriter(writer w);

BufferedWriter bw = new BufferedWriter(writer w, int buffersize);

- ex: BufferedWriter bw = new BufferedWriter(new FileWriter(“abc.txt”));

- we have all methods same as FileWriter and in this one extra method added that is .newLine() method

- bw.newLine();

- it is used for newline

**BufferedReader**

- it don’t communicate with file directly. It can communicate via reader object only

- it is same as FileReader .it has all methods of FileReader and contain one extra method readLine();

- ex: BufferedReader br = new BufferedReader(Reader r);

BufferedReader br = new BufferedReader(Reader r, int buffersize);

- ex: BufferedReader br = new BufferedReader(new FileReader(“abc.txt”));

- we have all methods same as FileWriter and in this one extra method added that is .readLine() method

- br.readLine();

- it is used for getting one line at a time from file

- if readLine() reaches to end of file then it give null as output

**PrintWriter**

- it is most enhanced and most powerful writer

- we can communicate with file directly or we can use writer reference also to communicate

- using this we can write any type of data into file. But, in FileWriter anf BufferedWriter we can able to write only char, String types

- it also contain all methods of FileWriter and added below methods extra

- pw.print(‘s’);

- this method used to write data into file

- using this we can write any type of data into file

- pw.println(‘s’);

- this method used to write data with line separator(goto next line after writing)

- using this we can write any type of data into file

- ex: PrintWriter pw = new PrintWriter(“abc.txt”);

PrintWriter pw = new PrintWriter(f);

PrintWriter pw = new PrintWriter(new FileWriter(“abc.txt”));

**Extra points:**

- if we want to use character data files then we should use reader/writer

- if we want to use video/audio/image data then we should use streams

- InputStream for reading data and OutputStream for writing data we use

- character data means text data

- binary data means video, audio, images