**Java -write and read Files:**

There different ways to write and read files using Java.

**Commonly used Classes to write a file:**

1. BufferedWriter
2. PrintWriter
3. FileOutputStream
4. DataOutputStream
5. RandomAccessFile
6. FileChannel

**1.**Write with **BufferedWritter: ( write a String to a new file).**

String s=”Hello World”;

BufferedWritter writer=new BufferedWriter(new FileWriter(filename));

writer.write(s);

writer.close();

**Append a String to a existing file:**

String s=”Java Program”;

BufferedWritter writer=new BufferedWriter(new FileWriter(filename, true));

writer.append(‘ ‘);

writer.append(s);

writer.close();

**2.**Write with **PrintWriter: ( write a String to a new file).**

FileWriter fileWriter = new FileWriter(filename);

PrintWriter printWriter = new PrintWriter(fileWriter);

printWriter.print(“hello”);

printWriter.printf(“today is %s and is %d”,”Sunday”,20); // to write formatted text

printWriter.close();

**3.**Write with **FileOutputStream: ( write binary data to a file- converts String into bytes and write bytes to file).**

String s=”Hello”;

FileOutputStream fos=new FileOutputStream(filename);

byte[] sBytes=s.getBytes();

fos.write(sBytes);

fos.close();

**4.**Write with **DataOutputStream: ( write a String to file).**

String s=”Hello”;

FileOutputStream fos=new FileOutputStream(filename);

DataOutputStream dos=new DataOutputStream(new BufferedOutputStream(fos));

dos.writeUTF(s);

dos.close();

**Reading the file:**

String res;

FileInputStream fis=new FileInputStream(filename);

DataInputStream dis=new DataInputStream(fis);

res=dis.readUTF();

sout(res);

**5.** Write with **RandomAccessFile:(can write, edit and also write at a specific position in an existing file).**

int i=40;

RandomAccessFile raf=new RandomAccessFile(filename, “rw”);

raf.seek(20); // at a position 20

raf.writeInt(i);

raf.close();

**Reading the int stored at specific location:**

int result = 0;

RandomAccessFile raf = new RandomAccessFile(filename, "r");

raf.seek(20);

result = reader.readInt();

raf.close();

return result;

**6.**Write with **FileChannel: ( dealing with large file, faster than standard io is FileChannel).**

RandomAccessFile raf = new RandomAccessFile(fileName, "rw");

FileChannel ch = raf.getChannel();

String value = "Hello";

byte[] sBytes = value.getBytes();

ByteBuffer bb = ByteBuffer.allocate(sBytes.length);

bb.put(sBytes);

bb.flip();

chl.write(buffer);

raf.close();

ch.close();

**Reading the file:**

String res;

RandomAccessFile raf = new RandomAccessFile(fileName, "r");

res=raf.readLine());

sout(res);

raf.close();

**7.Wite with Files Class ( Java 7 introduces a new way of working with filesystem with Files Class).**

String str = "Hello";

Path path = Paths.get(fileName);

byte[] strToBytes = str.getBytes();

Files.write(path, strToBytes);

**Reading the file:**

String res = Files.readAllLines(path).get(0);

sout(res);

**Commonly used Classes to read a file:**

1. BufferedReader
2. Scanner
3. StreamTokenizer
4. DataInputStream
5. SequenceInputStream
6. FileChannel

**1.**Read with **BufferedReader: ( to read a String)**

String file ="src/test/resources/test\_read.txt";

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

String s = reader.readLine(); // will return null when the end of file is occured

reader.close();

sout(s);

**2.**Read with **Scanner:**

String file = "src/test/resources/test\_read.txt";

Scanner scanner = new Scanner(new File(file));

scanner.useDelimiter(" ");

String s;

while(scanner.hasNext()){

s=s+scanner.next()

}

sout(s);

scanner.close();

**3.**Read with **StreamTokenizer: ( read a text file into tokens using StreamTokiners).**

String file = "src/test/resources/test\_read.txt";

FileReader reader = new FileReader(file);

StreamTokenizer tokenizer = new StreamTokenizer(reader);

// token 1

tokenizer.nextToken();

assertEquals(StreamTokenizer.TT\_WORD, tokenizer.ttype);

assertEquals("Hello", tokenizer.sval);

// token 2

tokenizer.nextToken();

assertEquals(StreamTokenizer.TT\_NUMBER, tokenizer.ttype);

assertEquals(1, tokenizer.nval, 0.0000001);

// token 3

tokenizer.nextToken();

assertEquals(StreamTokenizer.TT\_EOF, tokenizer.ttype);

reader.close();

**4.**Read with **DataInputStreamr: ( to read binay or primitive data types from a file).**

String expectedValue = "Hello";

String file ="src/test/resources/test\_read.txt";

DataInputStream reader = new DataInputStream(new FileInputStream(file));

String result = reader.readUTF();

reader.close();

assertEquals(expectedValue, result);

**5.**Read with **FileChannel: ( when dealing with large file and faster can be faster that standard IO).**

String expected\_value = "Hello world";

String file = "src/test/resources/test\_read.txt";

RandomAccessFile reader = new RandomAccessFile(file, "r");

FileChannel channel = reader.getChannel();

int bufferSize = 1024;

if (bufferSize > channel.size()) {

bufferSize = (int) channel.size();

}

ByteBuffer buff = ByteBuffer.allocate(bufferSize);

channel.read(buff);

buff.flip();

assertEquals(expected\_value, new String(buff.array()));

channel.close();

reader.close();

**6.**Read with **StringBuilder: (read the entire content of a file into a String).**

String expected\_value = "Hello world n Test line n";

String file = "src/test/resources/test\_read.txt";

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

StringBuilder builder = new StringBuilder();

String s = reader.readLine();

while (s != null) {

builder.append(s);

builder.append("n");

s = reader.readLine();

}

reader.close();

assertEquals(expected\_value, builder.toString());

**7.**Read with **File using Java 7: (read the entire content of a file into a String – small file).**

String expected\_value = "Hello world";

Path path = Paths.get("src/test/resources/test\_read.txt");

String s= Files.readAllLines(path).get(0);

// Or also sing readAllByes()

String s=Files.readAllBytes(path).get(0);

assertEquals(expected\_value, s);

**8.**Read with **File using Java 7: (read the entire content of a file into a String – large file).**

String expected\_value = "Hello world";

Path path = Paths.get("src/test/resources/test\_read.txt");

BufferedReader reader = Files.newBufferedReader(path);

String s = reader.readLine();

assertEquals(expected\_value, s);