RandomAccess File

This class is used for reading and writing to random access file. A random access file behaves like a large array of bytes. There is a cursor implied to the array called file pointer, by moving the cursor we do the read write operations. If end-of-file is reached before the desired number of byte has been read than EOFException is thrown. It is a type of IOException.

Benefits of Using Random Access File

Unlike IO streams that allow reading and writing data sequentially, random access file allows you to read and write a small chunk of data at any position in the file, using a file pointer. By controlling this file pointer, you can move forth and back within the file to perform reading and writing operations, at any position you want, in a random fashion.

Random access file is ideal for manipulating a particular file format at low level, such as reading, creating and updating MP3 files, PDF files, etc and even your own file format.

How to Use the RandomAccessFile Class

You can create a RandomAccessFile object by supplying the file path as a String or a File object, plus the access mode:

- public **RandomAccessFile(File file, String mode)**
- public **RandomAccessFile(String name, String mode)**

Modes of RandomAccessFile

The access mode can be one of the following values:

- **“r”:** reading only mode.
- **“rw”:** reading and writing mode.
- **“rws”:** same as “rw”, plus any changes to the file’s content and its metadata (such as the last modification time) take effect immediately.
- **“rwd”:** same as “rw”, plus any changes to the file’s content, but not its metadata take effect immediately.

Methods of RandomAccessFile

For controlling the file pointer, the RandomAccessFile class provides the following methods:

- **close():** It closes this random access file stream and releases any system resources associated with the stream.
- **write():** It writes the specified byte to this file
- **read():** It reads a byte of data from this file.
- **length():** It returns the length of this file

- **seek(long pos):** moves the file pointer to a specified position in the file. The offset is measured in bytes from the beginning of the file. At this position, the next read or write occurs.
- **skipBytes(int n):** moves the file pointer advance n bytes from the current position. This skips over n bytes of input.
- **getFilePointer():** returns the current position of the file pointer.

For reading and writing, as the RandomAccessFile class implements both the DataOutput and DataInput interfaces, you can use their methods (to name a few):

- **Reading methods: r**ead(byte[]), readByte(), readInt(), readLong(), etc.
- **Writing methods:** write(byte[]), writeByte(), writeInt(), writeLong(), etc.

Java RandomAccessFile read example

We can read byte array from a file using RandomAccessFile in java. Below is the pseudo code to read file using RandomAccessFile.x

```
RandomAccessFile raf = new RandomAccessFile("file.txt", "r");

raf.seek(1);

byte[] bytes = new byte[5];

raf.read(bytes);

raf.close();

System.out.println(new String(bytes));
```

In the first line, we are creating RandomAccessFile instance for the file in read-only mode.

Then in the second line, we are moving the file pointer to index 1.

We have created a byte array of length 5, so when we are calling read(bytes) method, then 5 bytes are read from file to the byte array.

Finally, we are closing the RandomAccessFile resource and printing the byte array to console.

Java RandomAccessFile write example

Here is a simple example showing how to write data to a file using RandomAccessFile in java.

```
RandomAccessFile raf = new RandomAccessFile("file.txt", "rw");

raf.seek(5);

raf.write("Data".getBytes());

raf.close();
```

Since RandomAccessFile treats the file as a byte array, write operation can override the data as well as it can append to a file. It all depends on the file pointer position. If the pointer is moved beyond the file length and then write operation is called, then there will be junk data written in the file. So you should take care of this while using write operation.

RandomAccessFile append example

All we need is to make sure file pointer is at the end of the file to append to a file. Below is the code for append to file using RandomAccessFile.

```
RandomAccessFile raf = new RandomAccessFile("file.txt", "rw");
raf.seek(raf.length());
raf.write("Data".getBytes());
raf.close();
```

Example

```
import java.io.IOException;
import java.io.RandomAccessFile;

public class ExFile
{
        static final String FILEPATH ="myFile.TXT";
   public static void main(String[] args)
   {
          try
          {
         System.out.println(new String(readFromFile(FILEPATH, 0, 18)));
              writeToFile(FILEPATH, "I love my country and my people", 31);
      }
      catch (IOException e)
      {
              e.printStackTrace();
      }
       }
   private static byte[] readFromFile(String filePath, int position, int size)
              throws IOException
       {
```

```
        RandomAccessFile file = new RandomAccessFile(filePath, "r");
            file.seek(position);
        byte[] bytes = new byte[size];
            file.read(bytes);
        file.close();
            return bytes;
    }
         private static void writeToFile(String filePath, String data, int position)
           throws IOException
    {
            RandomAccessFile file = new RandomAccessFile(filePath, "rw");
        file.seek(position);
            file.write(data.getBytes());
        file.close();
         }
}
```

The myFile.TXT contains text "This class is used for reading and writing to random access file." after running the program it will contains
This class is used for reading I love my country and my peoplele.

Java RandomAccessFile Example

Here is a complete java RandomAccessFile example with different read and write operations.

```
package com.journaldev.files;

import java.io.IOException;

import java.io.RandomAccessFile;

public class RandomAccessFileExample { public static void main(String[] args)

{

   try

    {

      // file content is "ABCDEFGH" String filePath =

        "/Users/pankaj/Downloads/source.txt";

      System.out.println(new String(readCharsFromFile(filePath, 1, 5)));

      writeData(filePath, "Data", 5); //now file content is "ABCDEData"

      appendData(filePath, "pankaj"); //now file content is "ABCDEDatapankaj"
```

```
    }
    catch (IOException e)
    {
       e.printStackTrace();
    }
}
private static void appendData(String filePath, String data) throws IOException
{
    RandomAccessFile raFile = new RandomAccessFile(filePath, "rw");
    raFile.seek(raFile.length());
    System.out.println("current pointer = "+raFile.getFilePointer());
    raFile.write(data.getBytes());
    raFile.close();
}
private static void writeData(String filePath,String data, int seek) throws IOException
{
    RandomAccessFile file = new RandomAccessFile(filePath, "rw");
    file.seek(seek);
    file.write(data.getBytes());
    file.close();
}
private static byte[] readCharsFromFile(String filePath, int seek, int chars) throws
IOException
{
    RandomAccessFile file = new RandomAccessFile(filePath, "r");
    file.seek(seek);
    byte[] bytes = new byte[chars];
    file.read(bytes);
    file.close();
    return bytes;
 }
```

```
}
```