

BAB X STREAM DAN FILE

Proses penulisan dan pembacaan data sering kita sebut dengan proses *input* dan *output*, dimana penulisan data berarti mengalirkan data ke output dan menerima atau mendapatkan data dari input.

A. Stream

Stream merupakan dasar operasi input-output (I/O) dalam Java yang menggunakan package java.io sebagai package utama. *Stream* adalah representasi abstrak dari input dan output device, dimana aliran bytes akan ditransfer seperti file dalam harddisk, file pada sistem remote atau printer. Kita dapat membaca data dari input stream, yang dapat berupa file, keyboard atau komputer remote. Sedangkan untuk operasi penulisan berarti menulis data pada output stream. Package java.io mendukung dua tipe stream, yaitu *binari* dan *karakter* stream. *Binari* merupakan data berupa bit atau data binari, sedangkan *karakter* adalah tipe khusus untuk pembacaan dan penulisan teks/karakter.

B. Input Stream

Subclass-subclass dari inputStream adalah : AudioInputStream, ByteArrayInputStream, FileInputStream, FilterInputStream, PipedInputStream, SequenceInputStream, dan StringBufferInputStream.

Dua method utama dari InputStream adalah :

- Read

Method ini digunakan untuk membaca stream.

- Close

Method ini digunakan untuk menutup koneksi input stream.

Latihan 38. InputStream.java

```
import java.io.*;

class InputStream {

    public static void main(String[] args) throws IOException {
        byte[] data = new byte[10];
        System.out.print("Ketik 10 buah karakter :");
        System.in.read(data);
        System.out.print("Karakter yang Anda ketik yaitu : ");
        for(int i=0;i<data.length;i++) {
            System.out.print((char)data[i]);
        }
    }
}
```



```
G:\JAVAA>javac InputStream.java
G:\JAVAA>java InputStream
Ketik 10 buah karakter :asep hs
Karakter yang Anda ketik yaitu : asep hs
```

C. Output Stream

Subclass-subclass dari outputStream adalah :

- `ByteArrayOutputStream` : digunakan untuk menuliskan stream menjadi byte array.
- `FileOutputStream` : digunakan untuk menulis pada file
- `FilterOutputStream` : merupakan superclass dari subclass-subclass seperti `DataOutputStream`, `BufferOutputStream`, `PrintStream`, `CheckedOutputStream`
- `ObjectOutputStream` : digunakan untuk menuliskan objek pada `OutputStream`.
- `PipedOutputStream` : digunakan untuk menjadi output dari `PipedInputStream`.

Sebagian method-method `OutputStream` adalah :

- `Void close()`
Menutup output stream yang aktif dan melepaskan sumber daya terkait dengan stream tersebut
- `Void flush()`
Melakukan flush output stream dan memaksa semua byte buffer untuk dituliskan keluar
- `Void write(byte[] b)`
Menulis sebanyak `b.length` dari byte array ke output stream
- `Void write(byte[] b, int off, int len)`
Menuliskan sebanyak `len` byte dari byte array `b` dimulai dari index `off`

Latihan 39. OutputStream.java

```
import java.io.*;

class OutputStream {
    public static void main(String[] args) throws IOException {
        byte[] data = {'a','b','c','d','e','f','g'};
        System.out.write(data,3,4);
        //pindah baris
        System.out.write('\n');
        //tuliskan semua isi array data
        System.out.write(data);
    }
}
```



```
C:\ Command Prompt
G:\JAVA>javac OutputStream.java
G:\JAVA>java OutputStream
defg
abcdefg
```

D. DataOutputStream

DataOutputStream merupakan class yang menyediakan cara praktis untuk menuliskan tipe data primitif ke output stream. Sebagian method `DataOutputStream` adalah :

- `writeDouble` : berfungsi menuliskan data bertipe double ke output stream
- `writeInt` : berfungsi menuliskan data bertipe integer ke output stream
- `writeBoolean` : berfungsi menuliskan data boolean ke output stream
- `writeUTF` : berfungsi menuliskan data string menggunakan encoding UTF-8 yang tidak tergantung pada mesin.

E. DataInputStream

`DataInputStream` berfungsi untuk saling melengkapi dengan `DataOutputStream`, yaitu untuk mendapatkan data yang ditulis dengan `DataOutputStream`. Sebagian method `DataInputStream` adalah :

- `ReadDouble()` : membaca data bertipe double
- `readInt()` : membaca data bertipe integer
- `readBoolean()` : membaca data boolean
- `readUTF()` : membaca data dengan encoding UTF-8

F. FileInputStream dan FileOutputStream

`FileInputStream` digunakan untuk membaca data dari file yang merupakan turunan langsung dari class `InputStream` dan `FileOutputStream` untuk menuliskan data ke file merupakan turunan langsung dari class `OutputStream`.

Latihan 40. FileInStream.java

```
import java.io.*;

class FileInStream {
    public static void main(String[] args){
        if(args.length==0) {
            System.out.println("Anda harus memasukkan nama file sebagai parameternya.");
            return;
        }
    }
}
```

```

byte data;
FileInputStream fin=null;
try{
    fin = new FileInputStream(args[0]);
    do{
        data = (byte)fin.read();
        System.out.print((char)data);
    }while(data!=-1);
    }catch(FileNotFoundException e) {
        System.out.println("File : " + args[0] + " tidak ditemukan.");
    }catch(IOException e) {
        System.out.println("Eksepsi tidak diketahui : " + e);
    }finally {
        //tutup file
        if(fin!=null) {
            try{
                fin.close();
            }catch(IOException err) {
                System.out.println("Eksepsi tidak diketahui : " + err);
            }
        }
    }
}

```

```

C:\> Command Prompt
G:\JAVUA>javac FileInStream.java
G:\JAVUA>java FileInStream
Anda harus memasukkan nama file sebagai parameternya.
G:\JAVUA>java FileInStream input.txt
contoh untuk program FileInStream.java?

```

Latihan 41. FileOutputStream

```

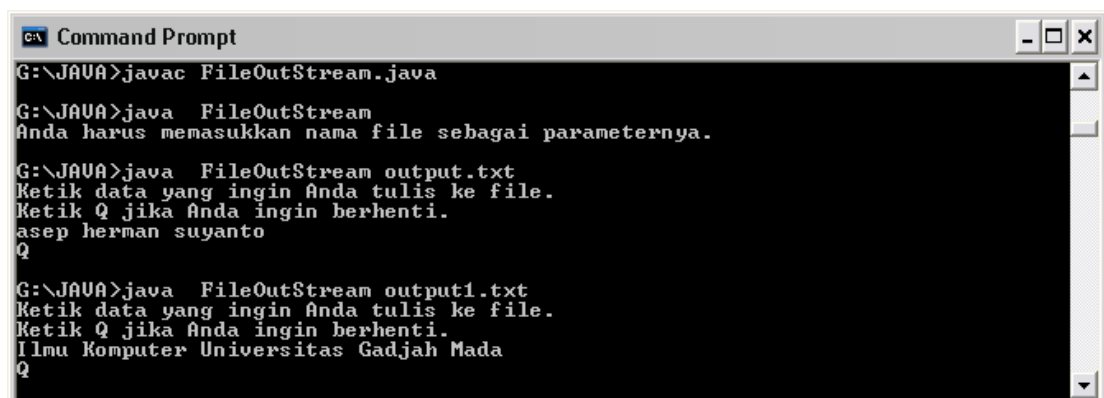
import java.io.*;
class FileOutputStream {
    public static void main(String[] args){
        if(args.length==0) {
            System.out.println("Anda harus memasukkan nama file sebagai parameternya.");
            return;
        }
    }
}

```

```

byte data;
FileOutputStream fout=null;
try{
    fout = new FileOutputStream(args[0]);
    System.out.println("Ketik data yang ingin Anda tulis ke file.");
    System.out.println("Ketik Q jika Anda ingin berhenti.");
    data = (byte)System.in.read();
    while(data!=(byte)'Q'){
        fout.write(data);
        data = (byte)System.in.read();
    }
}catch(FileNotFoundException e) {
    System.out.println("File : " + args[0] + " tidak dapat dibuka atau dibuat.");
}catch(IOException e) {
    System.out.println("Eksepsi tidak diketahui : " + e);
}finally {
    //tutup file
    if(fout!=null) {
        try{
            fout.close();
        }catch(IOException err) {
            System.out.println("Eksepsi tidak diketahui : " + err);
        }
    }
}
}

```



```

C:\> Command Prompt
G:\JAVUA>javac FileOutputStream.java
G:\JAVUA>java FileOutputStream
Anda harus memasukkan nama file sebagai parameternya.
G:\JAVUA>java FileOutputStream output.txt
Ketik data yang ingin Anda tulis ke file.
Ketik Q jika Anda ingin berhenti.
asep herman suyanto
Q
G:\JAVUA>java FileOutputStream output1.txt
Ketik data yang ingin Anda tulis ke file.
Ketik Q jika Anda ingin berhenti.
Ilmu Komputer Universitas Gadjah Mada
Q

```

Keterangan : file output1.txt sebenarnya tidak ada, dengan perintah diatas akan secara langsung terbuat.

G. Class File

Class **File** merupakan langkah awal dalam mempelajari proses input-output dengan Java, karena **File** merupakan objek yang mewakili path, file, atau direktori pada harddisk. Ada tiga cara membuat objek **File**, yaitu :

- Menggunakan objek string sebagai argumen yang menginformasikan path untuk file atau direktori.
- Menggunakan dua langkah, dimana yang pertama untuk mendefinisikan direktori dan yang kedua untuk file.
- Menggunakan dua argumen, dimana yang pertama adalah argumen string yang mendefinisikan direktori, dan yang kedua adalah argumen string yang mendefinisikan nama file.

H. FileWriter

Di dalam aplikasi web, disamping database, penggunaan file untuk menyimpan data cukup banyak dilakukan karena kebutuhan penyimpanan data yang sederhana cukup dengan menggunakan file. FileWriter merupakan subclass dari OutputStreamWriter yang merupakan subclass dari class abstract Writer. Class FileWriter memiliki konstruktor yang umum seperti berikut :

- FileWriter (File objekfile);
- FileWriter (String pathkefile);
- FileWriter (String pathkefile, boolean append);

Contoh penggunaan :

```
File inifile = (pathdirektori, namafile);  
FileWriter outputnya = new FileWriter (inifile);
```

Latihan 42. MenulisFile.java

```
import java.io.*;  
class MenulisFile {  
    public static void main(String[] args){  
        if(args.length==0) {  
            System.out.println("Anda harus memasukkan nama file sebagai  
parameternya.");  
            return;  
        }  
        String data;  
        FileWriter fout=null;  
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
```

```

try{
    fout = new FileWriter(args[0]);
    System.out.println("Ketik data yang ingin Anda tulis ke file.");
    System.out.println("Ketik BERHENTI jika Anda ingin berhenti.");
    data = br.readLine();
    while(!data.equals("BERHENTI")){
        // \r\n digunakan untuk pindah baris
        fout.write(data + "\r\n");
        data = br.readLine();
    }
} catch(FileNotFoundException e) {
    System.out.println("File : " + args[0] + " tidak dapat dibuka atau dibuat.");
} catch(IOException e) {
    System.out.println("Eksepsi tidak diketahui : " + e);
} finally {
    //tutup file
    if(fout!=null) {
        try{
            fout.close();
        } catch(IOException err) {
            System.out.println("Eksepsi tidak diketahui : " + err);
        }
    }
}
}
}

```

```

G:\JAVUA>javac MenulisFile.java
G:\JAVUA>java MenulisFile
Anda harus memasukkan nama file sebagai parameternya.
G:\JAVUA>java MenulisFile Menulis.txt
Ketik data yang ingin Anda tulis ke file.
Ketik BERHENTI jika Anda ingin berhenti.
Ilmu Komputer
Matematika dan Ilmu Pengetahuan Alam
Universitas Gadjah Mada
BERHENTI

```

I. **FileReader**

FileReader merupakan class yang dapat digunakan untuk membaca file teks.

Konstruktor dari **FileReader** :

- `FileReader(File objekfile);`

- `FileReader(String pathkefile);`

Method yang digunakan :


- `Read(char[] array);`
- `Read(char[] array, int offset, int length);`

Contoh penggunaan :

```
File fileteks = new File(direktori, namafile);
FileReader baca = new FileReader(fileteks);
C=baca.read(char[] yang dibaca);
```

Latihan 43. BacaFile.java

```
import java.io.*;
class BacaFile {
    public static void main(String[] args){
        if(args.length==0) {
            System.out.println("Anda harus memasukkan nama file sebagai
parameternya.");
            return;
        }
        String data;
        FileReader fin=null;
        try{
            fin = new FileReader(args[0]);
            //bungkus objek FileReader dengan objek BufferedReader
            BufferedReader br = new BufferedReader(fin);
            do{
                data = br.readLine();
                System.out.println(data);
            }while(data!=null);
        }catch(FileNotFoundException e) {
            System.out.println("File : " + args[0] + " tidak ditemukan.");
        }catch(IOException e) {
            System.out.println("Eksepsi tidak diketahui : " + e);
        }finally {
            //tutup file
            if(fin!=null) {
                try{
                    fin.close();
```

```
G:\JAVA>javac BacaFile.java

G:\JAVA>java BacaFile
Anda harus memasukkan nama file sebagai parameternya.

G:\JAVA>java BacaFile Baca.txt
Program Studi Ilmu Komputer
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Gadjah Mada
null
```

LATIHAN

1. Jelaskan tentang Byte Stream dan Character Stream, beserta dengan perbedaan utamanya ?
2. Berikan contoh program untuk `DataInputStream` dan `DataOutputStream` !
3. Buatlah program untuk mengecek keberadaan file/isi pada direktori !
4. Buatlah program untuk mengkopi suatu file teks ! Nama file yang hendak dikopi dan hasil kopinya harus dilewatkan sebagai parameter pada waktu eksekusi.

Contoh eksekusi program :

Java kopi file1.txt file2.txt