PRAKTIKUM 26

INPUT DAN OUTPUT 2

A. TUJUAN PEMBELAJARAN

1. Memahami konsep Input dan Output di Java
2. Mengenal kelas – kelas yang berhubungan dengan IO.
3. Mampu membuat program yang menerapkan konsep Input Output.

B. DASAR TEORI

Dalam pemrograman Java, Stream and File bukanlah istilah yang asing didengar. Stream and File merupakan proses penulisan dan pembacaan data sering yang di sebut dengan proses input dan output, dimana penulisan data berarti mengalirkan data ke output dan menerima atau mendapatkan data dari input.

Penerapan dalam pemrograman java ini selalu melakukan proses input dan output yaitu memindahkan byte data dari satu system ke sistem lain. Data yang dibaca dari server yang mengirim data tidak berbeda dengan membaca data dari sebuah file. Java mengangani I/O secara berbeda dari bahasa-bahasa pemrograman yang lainnya.

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. Dalam Stream ini terdapat proses input dan output sebagai berikut :

1. Input Stream

Kelas java.io.InputStream adalah:

```
public abstract class InputStream
```

Adapun 2 method utama dari InputStream adalah :

- **Read**

  Method ini digunakan untuk membaca stream.

- **Close**

  Method ini digunakan untuk menutup koneksi input stream.

Dalam proses Penginputan Stream ini pun terdapat pembagian dalam kelasnya, yaitu :

a) Byte Stream

Merupakan kelas dan interface ini digunakan untuk menangani data biner.

b) Character Stream

Merupakan kelompok kelas ini digunakan untuk menangani proses baca tulis karakter Unicode. Kelas ini merupakan pengembangan dari kelas Byte Stream sehingga lebih efisien.

Data input dalam Stream ini berfungsi untuk saling melengkapi dengan DataOutputStream, yaitu untuk mendapatkan data yang telah ditulis.

2. 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.

Data Output dalam stream ini merupakan class yang menyediakan cara praktis untuk menuliskan tipe data primitif ke output stream yang lebih mudah digunakan dalam penyelesaian program dalam java.

FILE

File merupakan data yang siap diinput dan diproses dalam Stream yang merupaka data operasi dalam pemrograman. Keterkaitan antara keduanya, proses Input dan Ouput tetap dilakukan walau dengan cara yang berbeda, dari subclass maupun method yang digunakan.

File Input Stream dan File Output Stream

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.

Dalam file pun terdapat subclass – subclass dan method, sama halnya dengan Stream, seperti :

1. 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.

2. File Writer

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

a) FileWriter (File objekfile);

b) FileWriter (String pathkefile);

c) FileWriter (String pathkefile, boolean append);

3. File Reader

File Reader 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);

C. TUGAS PENDAHULUAN

Pelajari konsep IO Java pada Java API documentation

D. PERCOBAAN

Percobaan 1 : Mencari file dengan ekstensi Tertentu

```
import java.io.*;

public class FindCertainExtension {
    private static final String FILE_DIR = "c:\\folder";
    private static final String FILE_TEXT_EXT = ".jpg";

    public static void main(String args[]) {
        new FindCertainExtension().listFile(FILE_DIR, FILE_TEXT_EXT);
    }

    public void listFile(String folder, String ext) {

        GenericExtFilter filter = new GenericExtFilter(ext);

        File dir = new File(folder);

        if(dir.isDirectory()==false){
            System.out.println("Directory does not exists : " + FILE_DIR);
            return;
        }

        // list out all the file name and filter by the extension
        String[] list = dir.list(filter);

        if (list.length == 0) {
            System.out.println("no files end with : " + ext);
            return;
        }

        for (String file : list) {
            String temp = new StringBuffer(FILE_DIR).append(File.separator)
                    .append(file).toString();
            System.out.println("file : " + temp);
        }
    }

    // inner class, generic extension filter
    public class GenericExtFilter implements FilenameFilter {

        private String ext;

        public GenericExtFilter(String ext) {
            this.ext = ext;
        }

        public boolean accept(File dir, String name) {
            return (name.endsWith(ext));
        }
    }
}
```

Percobaan 2 : Membaca ukuran file

```
import java.io.File;
public class FileSizeExample {
    public static void main(String[] args)
    {
        File file =new File("c:\\java_xml_logo.jpg");

        if(file.exists()){

            double bytes = file.length();
            double kilobytes = (bytes / 1024);
            double megabytes = (kilobytes / 1024);
            double gigabytes = (megabytes / 1024);
            double terabytes = (gigabytes / 1024);
            double petabytes = (terabytes / 1024);
            double exabytes = (petabytes / 1024);
            double zettabytes = (exabytes / 1024);
            double yottabytes = (zettabytes / 1024);

            System.out.println("bytes : " + bytes);
            System.out.println("kilobytes : " + kilobytes);
            System.out.println("megabytes : " + megabytes);
            System.out.println("gigabytes : " + gigabytes);
            System.out.println("terabytes : " + terabytes);
            System.out.println("petabytes : " + petabytes);
            System.out.println("exabytes : " + exabytes);
            System.out.println("zettabytes : " + zettabytes);
            System.out.println("yottabytes : " + yottabytes);
        }else{
             System.out.println("File does not exists!");
        }

    }
}
```

Percobaan 3 : Contoh menggunakan BufferedInputStream

```
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.io.LineNumberReader;

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

        try{

                File file =new File("c:\\ihave10lines.txt");

                if(file.exists()){

                    FileReader fr = new FileReader(file);
                    LineNumberReader lnr = new LineNumberReader(fr);

                    int linenumber = 0;

                    while (lnr.readLine() != null){
                        linenumber++;
                    }

                    System.out.println("Total number of lines : " + linenumber);

                    lnr.close();


                }else{
                         System.out.println("File does not exists!");
                }

        }catch(IOException e){
                e.printStackTrace();
        }

    }
}
```

Asumsi menggunakan file dengan nama ihave10lines.txt

Percobaan 4 : Membaca file enkode UTF-8

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;

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

        try {
                File fileDir = new File("c:\\temp\\test.txt");

                BufferedReader in = new BufferedReader(
                   new InputStreamReader(
                      new FileInputStream(fileDir), "UTF8"));

                String str;

                while ((str = in.readLine()) != null) {
                    System.out.println(str);
                }

                in.close();
            }
            catch (UnsupportedEncodingException e)
            {
                        System.out.println(e.getMessage());
            }
            catch (IOException e)
            {
                        System.out.println(e.getMessage());
            }
            catch (Exception e)
            {
                        System.out.println(e.getMessage());
            }
        }
}
```

Percobaan 5 : Menulis Enkode UTF-8 ke File

```
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStreamWriter;
import java.io.UnsupportedEncodingException;
import java.io.Writer;

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

          try {
                File fileDir = new File("c:\\temp\\test.txt");

                Writer out = new BufferedWriter(new OutputStreamWriter(
                        new FileOutputStream(fileDir), "UTF8"));

                out.append("Website UTF-8").append("\r\n");
                out.append("?? UTF-8").append("\r\n");
                out.append("??????? UTF-8").append("\r\n");

                out.flush();
                out.close();

            }
           catch (UnsupportedEncodingException e)
           {
                System.out.println(e.getMessage());
           }
           catch (IOException e)
           {
                System.out.println(e.getMessage());
            }
           catch (Exception e)
           {
                System.out.println(e.getMessage());
           }
        }
}
```

Percobaan 6 : Mengubah String ke InputStream

```
import java.io.BufferedReader;
import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;

public class StringToInputStreamExample {
    public static void main(String[] args) throws IOException {
        String str = "This is a String ~ GoGoGo";

        // convert String into InputStream
        InputStream is = new ByteArrayInputStream(str.getBytes());

        // read it with BufferedReader
        BufferedReader br = new BufferedReader(new InputStreamReader(is));

        String line;
        while ((line = br.readLine()) != null) {
                System.out.println(line);
        }

        br.close();
    }
}
```

Percobaan 7 : Membaca space Harddisk

```
import java.io.File;

public class DiskSpaceDetail
{
    public static void main(String[] args)
    {
        File file = new File("c:");
        long totalSpace = file.getTotalSpace(); //total disk space in bytes.
        long usableSpace = file.getUsableSpace(); ///unallocated / free disk space in bytes.
        long freeSpace = file.getFreeSpace(); //unallocated / free disk space in bytes.

        System.out.println(" === Partition Detail ===");

        System.out.println(" === bytes ===");
        System.out.println("Total size : " + totalSpace + " bytes");
        System.out.println("Space free : " + usableSpace + " bytes");
        System.out.println("Space free : " + freeSpace + " bytes");

        System.out.println(" === mega bytes ===");
        System.out.println("Total size : " + totalSpace /1024 /1024 + " mb");
        System.out.println("Space free : " + usableSpace /1024 /1024 + " mb");
        System.out.println("Space free : " + freeSpace /1024 /1024 + " mb");
    }
}
```

E. LATIHAN

Latihan 1 : Modifikasi program berikut agar dapat dikompile

```
public static void cat(File file) {
    RandomAccessFile input = null;
    String line = null;

    try {
        input = new RandomAccessFile(file, "r");
        while ((line = input.readLine()) != null) {
            System.out.println(line);
        }
        return;
    } finally {
        if (input != null) {
            input.close();
        }
    }
}
```

F. TUGAS

Buatlah tabel mengenai kelas Input Output Java

G. LAPORAN RESMI

Kumpulkan hasil percobaan di atas dan tambahkan analisa untuk tiap percobaan, latihan, dan tugas yang telah dibuat.