Pertemuan 7

File pada Java

Elia Dania Serepina
PJ AP2A

Java IO 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.

Package Java IO

Program Java melakukan pemrosesan file dengan menggunakan class-class dari *packagejava.io*.

Package java.io ini berisikan class-class streams seperti:

1. **FileInputStream** : Untuk input berupa byte dari suatu file
2. **FileOutputStream**: Untuk output berupa byte kepada suatu file
3. **FileReader** : Untuk input berupa karakter dari suatu file
4. **FileWriter** : Untuk output berupa karakter kepada suatu file

Input Stream

Subclass-subclass dari inputStream adalah :

- AudioInputStream
- ByteArrayInputStream
- FileInputStream
- FilterInputStream
- PipedInputStream
- SequenceInputStream
- StringBufferInputStream.

Dua method utama dari InputStream adalah :

- **Read** : Method ini digunakan untuk membaca stream.
- **Close** : Method ini digunakan untuk menutup koneksi input stream.

Output Stream

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 :

- **Voidclose()** : 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

Class dan Method File

Class File adalah kunci dalam pemrosesan File atau Direktori. Objek File merepresentasikan single file atau directory. **Class File** berguna untuk mengambil informasi mengenai suatu file atau direktori dari disk.

Constructor class File tediri dari :

a. public File(String name)
b. public File(String pathToName, String name)
c. public File(File directory, String name)
d. public File(URI uri)

Membuat Objek File

Ada 3 cara membuat objek File, yaitu :

1. Menggunakan objek String sebagai argument yang menginformasikan path untuk file atau direktori. Contoh :

   File direktori = new File("c:\\my documents\\java\\");
   File fileku = new File("c:\\my documents\\java\\dokumen.txt");

2. Menggunakan dua langkah dimana yang pertama untuk mendefinisikan direktori dan yang kedua untuk file. Contoh :

   File dirku = new File("c:\\my documents\\java");
   File filenya = new File(dirku, "dokumennya.txt");

3. Menggunakan dua argument dimana yang pertama adalah argument String yang mendefinisikan direktori, dan yang kedua adalah argument String yang mendefinisikan nama file. Contoh :

   File filesaya = new File("c:\\my documents\\java\\","dokumennya.txt");

Method pada Class File

Method	Keterangan
exists()	Mengembalikan nilai true apabila file atau direktori itu ada
isDirectory()	Mengecek apakah objek file menunjuk pada direktori atau file
getName()	Mendapatkan nama file atau direktori dalam string(tanpa path) dari objek File
getPath()	Mendapatkan path dalam String dari objek File,termasuk nama File / direktori
getAbsolutePath()	Mendapatkan path absolute dari direktori / file yang direferensi oleh objek file
list()	Bila objek File mewakili direktori maka akan mengembalikan array String yang mengandung nama dari isi direktori. Bila objek File merupakan file maka akan mengembalikan nilai null
listFiles()	Bila objek file berupa direktori maka akan mengembalikan objek File yang ada dalam direktori
length()	Mengembalikan nilai bertipe long yang merupakan panjang bytes dari file yang diwakili oleh objek File, bila berupa direktori akan mengembalikan nilai 0
lastModified()	Mengembalikan nilai long yang mewakili waktu terakhir objek File terakhir kali dimodifikasi.

Teknik Operasi File

Berikut ini adalah macam-macam teknik operasi file pada Java antara lain :

1. Membuat File
2. Menampilkan nama File dan Direktori
3. Me-rename File
4. Menghapus File
5. Menghapus non-empty Direktori

Membuat Objek File

Untuk membuat object File, kita cukup memanggil salah satu constructor-nya.

```java
import java.io.File;

public class file {
    public static void main(String[] args) {
        String path = "D:\\Belajar\\Belajar Java";
        File file = new File(path);

        if (file.exists()) {
            System.out.println("File name: " + file.getName());
            System.out.println("Absolute path: " + file.getAbsolutePath());
            System.out.println("Writeable: " + file.canWrite());
            System.out.println("Readable: " + file.canRead());
            System.out.println("File size in bytes: " + file.length());
        } else {
            System.out.println("The file does not exist.");
        }
    }
}

```

Menampilkan Isi Direktori

Kode berikut ini menampilkan nama-nama file yang ada dalam suatu direktori:

```java
import java.io.File;

public class file {
    public static void main(String[] args) {
        String path = "D:\\Belajar\\Belajar Java";
        File dir = new File(path);

        if (dir.isDirectory()) {
            System.out.println("Directory of " + path);
            String[] dirContent = dir.list();

            for (int i = 0; i < dirContent.length; i++) {
                File file = new File(path + "\\" + dirContent[i]);

                if (file.isDirectory()) {
                    System.out.println("<DIR> " + dirContent[i]);
                } else {
                    System.out.println(" " + dirContent[i]);
                }
            }
        } else {
            System.out.println(path + " is not a directory");
        }
    }
}
```

Menampilkan Hanya File

Menampilkan hanya file saja, tidak menampilkan subdirektori maupun hidden files

```java
import java.io.File;

public class file {
    public static void main(String[] args) {
        String path = "D:\\Belajar\\Belajar Java";
        File dir = new File(path);

        if (dir.isDirectory()) {
            System.out.println("Directory of " + path);
            File[] files = dir.listFiles();

            for (File file : files) {
                if (file.isFile() && !file.isHidden()) {
                    System.out.println(file.getName() + " " + file.length() + " bytes");
                }
            }
        }
    }
}
```

Me-rename File

```java
import java.io.File;

public class file {
    public static void main(String[] args) {
        String path = "D:\\Belajar\\Belajar Java\\test.txt";
        File file = new File(path);

        if (file.renameTo(new File("test1.txt")))
            System.out.println("File renamed successfully");
        else
            System.out.println("Failed to rename file");
    }
}
```

Menghapus File

```java
import java.io.File;

public class file {
    public static void main(String[] args) {
        String path = "D:\\Belajar\\Belajar Java\\test1.txt";
        File file = new File(path);

        if (file.delete())
            System.out.println("File deleted successfully");
        else
            System.out.println("Failed to delete file");
    }
}
```

Untuk menghapus sebuah folder termasuk file dan subdirektori di dalamnya, kita cukup memanggil method tersebut:

deleteFile(newFile("nama_folder_yg_dihapus");

BufferReader

Class **BufferedReader** dapat “membungkus” class FileReader untuk menyediakan proses input yang lebih efisien. Class ini menambahkan suatu “buffer” kepada input stream sehingga input tersebut dibaca dalam “potongan besar” dari harddisk daripada byte-per-byte. Hal ini menghasilkan peningkatan performance. Class BufferedReader juga memungkinkan kita untuk membaca data secara per-karakter atau per-baris.

Langkah-langkah:
- Buat object File
- Buat objek FileReader
- Buat objek BufferedReader

Contoh membuat objek BufferedReader untuk membaca file “movie.txt”:

```
File f = newFile("movies.txt");
BufferedReader in = newBufferedReader(new FileReader(f));
```

Read dan ReadLine

Kita dapat menggunakan methods *read* dan *readLine* untuk membaca isi objek BufferedReader.

int read()
- Membaca satu karakter dari file dan me-return suatu angka.
- Menghasilkan -1 apabila end-of-file telah dicapai.
- Throws IOException

String readLine()
- Membaca satu baris dan me-return-nya sebagai String.
- Me-return *null* apabila end-of-file telah dicapai.
- Throws IOException

FileWriter

FileWriter merupakan subclass dari *OutputStreamWriter* dimana class *OutputStreamWriter* adalah subclass dari class abstrak Writer.

Class Writer 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);

PrintWriter

PrintWriter merupakan subclass dari class abstrak Writer yang digunakan melakukan output dari berbagai macam tipe data yang kemudian dikonversi ke bentuk karakter.

Penggunaan PrintWriter dengan FileWriter :

```
PrintWriter fileoutput = new PrintWriter (
        New FileWriter(File objekfile);
);
```

Activity

```java
import java.io.*;
import java.util.Scanner;

public class la7 {
    public static void main(String[] args) throws FileNotFoundException, IOException {
        String nama, npm, grade, keterangan;
        int pilih, nilai;

        PrintStream diskWriter = new PrintStream("D:\\Belajar\\Belajar Java\\Elia_50420416.txt");
        Scanner input = new Scanner(System.in);

        do {
            System.out.println("========== Universitas Gunadarma ==========");
            System.out.println("1. Input Nilai");
            System.out.println("2. Keluar");
            System.out.println("==========================================");
            System.out.print("Pilih: ");
            pilih = input.nextInt();
            System.out.println("");
```

```
        switch(pilih) {
            case 1:
                System.out.print("Masukkan Nama: ");
                nama = input.next();
                System.out.print("Masukkan NPM: ");
                npm = input.next();
                System.out.print("Masukkan Nilai: ");
                nilai = input.nextInt();

                if (nilai >= 80) {
                    grade = "A";
                    keterangan = "Lulus";
                } else if (nilai >= 70) {
                    grade = "B";
                    keterangan = "Lulus";
                } else if (nilai >= 60) {
                    grade = "C";
                    keterangan = "Lulus";
                } else if (nilai >= 50) {
                    grade = "D";
                    keterangan = "Tidak Lulus";
                } else {
                    grade = "E";
                    keterangan = "Tidak Lulus";
                }

                System.out.println("\nNama: " + nama + "\nNPM: " + npm + "\nGrade: " + grade + "\nKeterangan: " + keterangan + "\n");
                diskWriter.println(nama +  ", " + npm + ", " + nilai + ", " + grade + ", " + keterangan);
                diskWriter.println("");
                break;

            case 2:
                System.out.println("Terima Kasih");
                break;

```

```
56                  default:
57                      System.out.println("Pilihan tidak tersedia");
58                      break;
59              }
60          } while (pilih != 2);
61      }
62  }
```

Thanks!