Dasar-Dasar Pemrograman 2

Tutorial 10 Kelas B, D, dan E

Text I/O

Kamis, 2 Mei 2019 - 16:00 WIB

Materi
Pada TP2 kalian sebelumnya sudah ada *sneak-peek* mengenai materi lab kali ini, yaitu text I/O. Nah, kali ini akan diperjelas apa itu text I/O dan bagaimana cara menggunakannya.

Text I/O

Adalah cara untuk membaca Input dari Console atau file lain dan/atau mengeluarkan Output ke sebuah file lain yang pada umumnya ber-tipe data text (seperti .txt). Selain itu disini kita akan membahas bagaimana cara membuat suatu file dapat digunakan di dalam suatu program java.

Ada banyak *class* di java yang menyediakan fungsionalitas untuk *read/write* dari/ke file lain. Untuk sekarang ini, kita akan menggunakan *BufferedReader* untuk membaca, *FileReader* untuk membuat suatu file agar dapat dibaca oleh *BufferedReader*, *File* untuk membuat sebuah file dapat dimanipulasi dalam program dan *PrintWriter* untuk menulis output ke-sebuah file baru.

(Jika kalian kebingungan atau ingin mengetahui lebih banyak method - method dari kelas tersebut, kalian dapat selalu membuka Javadoc yang disediakan oleh java lho!)

Pembaca File

- **FileReader**
  FileReader adalah suatu class di Java yang bisa digunakan untuk membaca karakter dalam file.
  Contoh penggunaan:
  - Buat file dengan nama Hello1.txt berisi "**This is an example**"

  - Buat constructor class FileReader yang menerima parameter String nama

file (dalam contoh ini “Hello1.txt”) yang ada di direktori file Java (dalam contoh ini FileRead.java).

```
import java.io.*;
public class FileRead {

   Run | Debug
   public static void main(String args[])throws IOException {

       // Creates a FileReader Object
       FileReader fr = new FileReader("Hello1.txt");
```

- Kita bisa menyimpan seluruh karakter di file ke dalam array dengan method read(char[] c) dengan parameter array char tempat kita ingin menyimpan hasil file yang di-read.

```
char [] a = new char[50];
fr.read(a);   // reads the content to the array

for(char c : a)
    System.out.print(c);   // prints the characters one by one
fr.close();
```

- Output yang dihasilkan adalah seperti ini.

```
This
 is
 an
 example
```

- **BufferedReader**

  BufferedReader adalah suatu Class di Java yang berfungsi untuk menyederhanakan pembacaan teks dari suatu *input stream* (misalnya *file*). Secara singkat, BufferedReader meminimalisir penggunaan *I/O operation* dengan membaca potongan karakter dan menyimpannya di dalam *internal buffer* yang bisa menghasilkan pembacaan yang lebih cepat, karena tidak perlu berinteraksi lagi dengan *file* yang dibuka. BufferedReader biasanya digunakan dengan membaca suatu objek FileReader dan memasukkannya ke dalam *buffer*.

  Contoh penggunaan:

  - Buat terlebih dahulu file input.txt berisi “**Halo Dunia!**”
  - Selanjutnya buatlah program ReadFile.java berikut untuk membaca file input.txt

```java
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;

public class ReadFile {
    public static void main(String[] args) throws IOException {
        BufferedReader reader = new BufferedReader(new FileReader("input.txt"));
        System.out.println(reader.readLine());
        reader.close();
    }
}
```

- Setelah dijalankan programnya akan menghasilkan sebagai berikut

```
dek.depe@cs.ui.ac.id:~$ ls
ReadFile.java input.txt
dek.depe@cs.ui.ac.id:~$ javac ReadFile.java
dek.depe@cs.ui.ac.id:~$ java ReadFile
Halo Dunia!
dek.depe@cs.ui.ac.id:~$
```

- Perhatikan di program ReadFile.java di atas, terdapat syntax 'throws IOException'. Seperti yang sudah dipelajari sebelumnya tentang materi *Error Handling*, pembacaan *file* input.txt dengan BufferedReader bisa saja terdapat *error* yaitu ketika *file* yang ingin dibaca tidak ditemukan.
-
- Perhatikan contoh keluaran berikut jika *file* input.txt tidak ditemukan.

```
dek.depe@cs.ui.ac.id:~$ ls
ReadFile.java
dek.depe@cs.ui.ac.id:~$ javac ReadFile.java
dek.depe@cs.ui.ac.id:~$ java ReadFile
Exception in thread "main" java.io.FileNotFoundException: input.txt (No such file or directory)
    at java.io.FileInputStream.open0(Native Method)
    at java.io.FileInputStream.open(FileInputStream.java:195)
    at java.io.FileInputStream.<init>(FileInputStream.java:138)
    at java.io.FileInputStream.<init>(FileInputStream.java:93)
    at java.io.FileReader.<init>(FileReader.java:58)
    at ReadFile.main(ReadFile.java:7)
dek.depe@cs.ui.ac.id:~$
```

- Kita juga bisa menggunakan 'try catch' *statement* saat menggunakan BufferedReader.

```java
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;

public class ReadFile {
    public static void main(String[] args) {
        try {
            BufferedReader reader = new BufferedReader(new FileReader("input.txt"));
            System.out.println(reader.readLine());
            reader.close();
        } catch (IOException e) {
            System.out.println("File tidak ditemukan");
        }
    }
}
```

- Lalu saat dijalankan dan *file* input.txt akan memberikan keluaran sebagai berikut.

```
dek.depe@cs.ui.ac.id:~$ ls
ReadFile.java
dek.depe@cs.ui.ac.id:~$ javac ReadFile.java
dek.depe@cs.ui.ac.id:~$ java ReadFile
File tidak ditemukan
dek.depe@cs.ui.ac.id:~$
```

- Kalian juga bisa mencoba bagaimana jika program ReadFile.java tersebut tidak menggunakan ‘throws IOException’ ataupun ‘try catch’ *statement*.

- **File**

File adalah suatu class di java yang merepresentasikan suatu “file” dalam program java. Objek dari kelas File dapat menyimpan berbagai atribut dari “file” dalam komputer itu sendiri seperti path dan sifatnya (apakah hanya bisa dibaca, apakah hanya bisa di-*write* dan sebagainya). Sekali dibuat, suatu *instance* dari class file tidak dapat diubah pathnya.

Contoh penggunaan :

```java
import java.io.File;

class fileboi{
    Run | Debug
    public static void main(String[]args) {
        File f = new File("ini.txt"); //membuka file bernama ini.txt

        System.out.println(f.getName()); //ini.txt
        System.out.println(f.getPath()); //ini.txt
        System.out.println(f.getAbsolutePath()); //C:\Users\arnas\Desktop\Temp\ini.txt
    }
}
```

Untuk info tentang *method-method* yang ada dalam class File silahkan merujuk kepada tautan berikut ini.

Penulis File

- **PrintWriter**
  Printwriter adalah sebuah class yang membantu program java untuk mencetak secara langsung kedalam class File yang telah dibuat. Perbedaan Printwriter dengan beberapa class untuk mencetak lainnya adalah Printwriter ditujukan untuk mencetak sebuah "teks" kedalam file, oleh karena itu PrintWriter dilengkapi dengan method format() dan printf(). PrintWriter mengimplementasikan method yang ada dalam PrintStream, oleh karena itu PrintWriter **harus di close()** ketika sudah selesai penggunaannya. Selengkapnya pada tautan berikut.

  **Perhatikan bahwa PrintWriter dapat memicu adanya FileNotFoundException pada pembuatan *instance* nya. Kira-kira bagaimana cara meng*handle* nya ya ? :)

Contoh penggunaan :

```java
import java.io.File;
import java.io.IOException;
import java.io.PrintWriter;

class fileboi{
    Run | Debug
    public static void main(String[]args) throws IOException {
        File f = new File("ini.txt"); //membuka file bernama ini.txt

        System.out.println(f.getName()); //ini.txt
        System.out.println(f.getPath()); //ini.txt
        System.out.println(f.getAbsolutePath()); //C:\Users\arnas\Desktop\Temp\ini.txt

        PrintWriter pw = new PrintWriter(f);
        pw.println("hai dunia");
        pw.printf("%d", 99);
        pw.close();
    }
}
```

Payday **(Bukan Judul Game)**

It's payday! Sebuah perusahaan yang tidak dapat disebutkan namanya sedang menghitung gaji yang akan dikeluarkan untuk membayar para karyawan yang bekerja di sana.
Setiap karyawan memiliki nama, status pekerjaan, dan *role* pekerjaan. status pekerjaan terdiri dari Intern, Part-time, dan Full-time. Sementara, *role* pekerjaan terdiri dari Web Developer, UI/UX Designer, dan Assembler Programmer. Perusahaan Ini menerapkan peraturan terkait minimal jam kerja per minggu yang berhubungan dengan tipe pekerjaan. Untuk setiap *role* pekerjaan, terdapat perhitungan bayaran per jam yang berbeda pula. Peraturan tersebut dirangkum dalam tabel di bawah ini:

Status Pekerjaan	Minimal Jam Kerja
Intern	30
Part-time	20
Full-time	40

Role	Bayaran Per Jam
Web Developer	40
UI/UX Designer	30
Assembly Programmer	100

Berikut adalah cara perhitungan gaji seorang karyawan:

1. Bila jam kerja yang dihabiskan **sama dengan** minimal jam kerja, maka gaji sementara adalah **jam kerja x bayaran**.
2. Bila jam kerja yang dihabiskan **lebih dari** minimal jam kerja, maka gaji sementara adalah **(minimal jam kerja x bayaran) + (jam ekstra x 150% bayaran)**.
   Note: jam ekstra merupakan selisih dari jam kerja karyawan dengan jam kerja minimal.
3. Bila jam kerja yang dihabiskan **kurang dari** minimal jam kerja, maka gaji sementara adalah **jam kerja x 50% bayaran**.
4. Gaji akhir yang didapat dari langkah 1, 2, atau 3 adalah gaji sementara yang dipotong pajak sebanyak 15%.

Kamu diminta untuk membuat program yang mensimulasikan proses pengolahan data CSV menjadi sebuah file TXT yang kontennya berasal dari pengolahan input.

Comma separated values, atau biasa disebut CSV, adalah sebuah format *file* sederhana yang biasa digunakan untuk menyimpan data yang tersusun dalam beberapa baris, seperti sebuah spreadsheet atau *database*. Satu baris dalam file CSV adalah sebuah *record*, di mana satu *record* dapat memiliki banyak *data fields* yang dibatasi (delimited) dengan tanda koma. Contohnya adalah sebagai berikut:

```
no ktp,no_rekening,kantor_cabang,nama_bank
3274212597827940,0907984212,Jakarta Selatan,Bank BNI
3278582361758580,111991018832,Jakarta Selatan,Bank Mandiri
3274692400892570,0656992156,Jakarta Selatan,Bank BNI
3275954822239780,0681171621,Cimahi,Bank BNI
3271786541831160,0656831317,Jakarta Selatan,Bank BNI
3274860243830410,111988744124,Jakarta Timur,Bank Mandiri
```

Juga terdapat satu baris header yang ada di baris pertama *file* CSV dengan format yang sama seperti baris *record*. *Header* berisi nama yang sesuai dengan *data field* dari *record* yang ada di file.

Spesifikasi Program :

Kamu diminta untuk mengolah konten sebuah file CSV menjadi sebuah file output berformat TXT yang menampilkan info setiap karyawan (id, nama, role, status, dan gaji akhir) beserta info biaya yang harus dikeluarkan oleh perusahaan untuk membayar gaji seluruh karyawannya pada baris terakhir.

Kamu akan diberikan template yang terdiri dari class-class berikut:

1. Main
2. Company
3. CompanyParser
4. Employee
5. AssemblerProgrammer
6. UIUXDesigner
7. WebDeveloper

Di dalam bagian tertentu pada class-class yang disediakan terdapat method yang belum memiliki implementasi penuh dan ditandai dengan comment TODO, **silahkan implementasikan method sesuai dengan ketentuan yang ada di template**. Kamu juga diberikan sebuah Main class yang akan di-*run*, **pastikan kode kalian memenuhi kriteria *file* ini**.

Berikut adalah contoh output jika diberikan sebuah *file* **data_karyawan.csv**.

Input:

[data_karyawan.csv]

```
emp_name,work_hour,job_role,stats
Moses M. Sosa,25,Web Developer,Part-time
Knox I. Huber,21,UI/UX Designer,Full-time
Morgan X. Aguirre,33,Assembler Programmer,Part-time
Imogene V. Hale,27,Web Developer,Part-time
Irene Y. Terrell,20,Web Developer,Intern
```

Output:

[all.txt] - Informasi semua karyawan

```
[Moses M. Sosa] - [Work Hour: 25] - [Job Role: Web Developer] - [Status: Part-time] - [Salary: 935]
[Name: Knox I. Huber] - [Work Hour: 21] - [Job Role: UI/UX Designer] - [Status: Full-time] - [Salary: 267.75]
[Name: Morgan X. Aguirre] - [Work Hour: 33] - [Job Role: Assembler Programmer] - [Status: Part-time] - [Salary: 3357.5]
[Name: Imogene V. Hale] - [Work Hour: 27] - [Job Role: Web Developer] - [Status: Part-time] - [Salary: 1037]
[Name: Irene Y. Terrell] - [Work Hour: 20] - [Job Role: Web Developer] - [Status: Intern] - [Salary: 340]
Total cost: 5937.25
```

[filterSalary.txt] - Misalkan ingin diketahui karyawan-karyawan yang gajinya lebih dari 500

```
[Moses M. Sosa] - [Work Hour: 25] - [Job Role: Web Developer] - [Status: Part-time] - [Salary: 935]
[Name: Morgan X. Aguirre] - [Work Hour: 33] - [Job Role: Assembler Programmer] - [Status: Part-time] - [Salary: 3357.5]
[Name: Imogene V. Hale] - [Work Hour: 27] - [Job Role: Web Developer] - [Status: Part-time] - [Salary: 1037]
Total cost: 5329.5
```

[overperformedEmployees.txt] - Mengetahui karyawan-karyawan yang jam kerjanya melebihi minimal jam kerja yang ditentukan

```
[Moses M. Sosa] - [Work Hour: 25] - [Job Role: Web Developer] - [Status: Part-time] - [Salary: 935]
[Name: Morgan X. Aguirre] - [Work Hour: 33] - [Job Role: Assembler Programmer] - [Status: Part-time] - [Salary: 3357.5]
[Name: Imogene V. Hale] - [Work Hour: 27] - [Job Role: Web Developer] - [Status: Part-time] - [Salary: 1037]
Total cost: 5329.5
```

Komponen Penilaian :

Komponen	Penjelasan	Bobot
Implementasi Class Main	Class Main dapat berjalan sebagai program simulasi yang mengolah CSV input menjadi TXT output yang diharapkan.	15 %
Implementasi Class Company	Class Company diimplementasikan dengan baik dan dapat melakukan behavior yang diinginkan.	20 %
Implementasi Class Employee beserta Subclassnya.	Abstract Class Employee dan subclass-subclassnya (AssemblerProgrammer, UIUIXDesigner, dan WebDeveloper) diimplementasikan dengan baik dan dapat melakukan behavior yang diinginkan.	20 %
Implementasi Class CompanyParser	Class CompanyParser diimplementasikan dengan baik sehingga dapat mengolah CSV input menjadi object-object yang diharapkan.	20 %
Ketepatan Output	Ketepatan konten file-file Output ketika Main dijalankan (Pemenuhan requirement client).	15 %
Kerapian	Penulisan program mengikuti kaidah dan konvensi yang telah diajarkan. Program ditulis dengan rapi, terstruktur, dan disertakan oleh dokumentasi secukupnya.	10 %

Deadline :

Kamis, 2 Mei 2019
Pukul **17:40 WIB**

Persentase Nilai Total Terhadap Lama Keterlambatan :

dari	hingga	persentase
0:00:00	0:00:00	100%
0:00:01	0:10:00	85%
0:10:01	0:20:00	70%
0:20:01	0:30:00	50%
0:30:01	24:00:00	25%

Format Pengumpulan :

Zip semua class yang kamu buat dan kumpulkan di slot pengumpulan yang telah disediakan di SCeLE dengan format :

[Kode Asdos]_[Nama]_[Kelas]_[NPM]_Lab[X].zip

Contoh :

TS_AnthonyEdwardStark_X_19630339_Lab10.zip

Acknowledged Lecturers :

- Fariz Darari, S.Kom, M.Sc., Ph.D.
- Ardhi Putra Pratama Hartono S.Kom, M.Sc
- Dipta Tanaya S.Kom, M.Kom.

Authors :

- LA
- WH
- MDN
- FAA