Buffered IOs

TEchnologies Internet (TEI)

Olivier Liechti

heig-vd

Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Foundations

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

W1

Introduction to Java IO

Byte and character streams, dealing with files, the decorator pattern, custom reader/writers classes, buffered IOs

W2

Socket API - TCP

Client and server programming, sockets and streams, multi-threaded servers

W3

Application-level Protocol

How to specify your own communication protocol? Implement the client and the server.

W4

Socket API - UDP

Client and server programming, broadcast/multicast, service discovery protocols

This Week

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- **Monday**
  - **08:30 - 08:45** : General introduction
  - **08:45 - 09:00** : Introduction to the Java IO API
  - **09:00 - 10:00** : Exercise 1 (computing stats about text data)
    5' intro, 10' individual analysis, 15' discussion, 30' individual work
  - **10:30 - 10:50** : Java IO & the Decorator Pattern
  - **10:50 - 12:00** : Exercise 2 (custom writers & decorator pattern)
    5' intro, 10' individual analysis, 15' discussion, 40' individual work
- **Homework**
  - Read selected Java Tutorial sections (see next slide)
  - Finish exercises 1 and 2
- **Wednesday**
  - **08:30 - 08:45** : Presentation of finished exercises 1 and 2
  - **08:45 - 09:00** : Buffered IOs in Java
  - **09:00 - 10:00** : Exercise 3 (measuring the performance impact of buffered IOs in Java)

Exercise 1

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- Write a java program that **reads a text file** and produces **computes basic statistics** about its content.
- **Functional requirements**
  - It should be possible to **call the program from the command line** and to pass the **file name** as an argument.
  - When running the program, the user should see at least the following information: 1) **number of characters**, 2) **number of uppercase characters**, 3) **number of lowercase characters**, 4) **number of vowels**, 5) **number of consonants**, 6) **number of digits**.
  - When running the program, the user should also see a **table**, **showing every character appearing in the text and its number of occurrences**.
- **Design requirements**
  - Apply object-oriented principles to have a modular and extensible solution.

Exercise 2

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- Write a java program that **reads a text file**, and **applies various transformations** on the content and **writes the result in another text file**.
- **Functional requirements**
    - It should be possible to **call the program from the command line** and to pass the **file name** as an argument. The program will write results to a file that has the same name with the **“.out” extension** (e.g. running the program on “data.txt” will produce “data.txt.out”.
    - The **first transformation** consists of passing all characters to uppercase. The **second transformation** consists of replacing all ‘a’ and ‘A’ characters with the ‘@’ character. The **third transformation** consists of inserting the number of characters at the end of each line (e.g. **Hello -> Hello [5]**)
- **Design requirements**
    - Apply the decorator pattern, by extending the FilterReader and/or the FilterWriter classes.

Buffered IOs

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Do you know what happens when you do a

```
int c = read();
```

?

Buffered IOs

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

It's a bit like when you are thirsty and feel like **drinking** something...

!

Buffered IOs

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Buffered IOs

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

It takes you 56' to sip a beer

20 min

5 min

10 min

20 min

1 min

Buffered IOs

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Thirsty again...

?

Buffered IOs

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

It takes you 56' again to sip the next beer

20 min 5 min 10 min 20 min 1 min

Buffered IOs

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Can we do better...

?

Buffered IOs

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

It still takes you 56' to bring back a pack of beers...

20 min

5 min

10 min

20 min

1 min

Buffered IOs

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

And only 2 minutes to sip the next one!

1 min

1 min

Buffered IOs

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Coming back to

```
int c = read();
```

- If you don't use buffered IOs, calling read() will issue **one system call** to retrieve **one single byte**... which is not efficient.

- With buffered IOs, calling read() will **pre-fetch "several" bytes** and store it in a **temporary memory space** (i.e. in a **buffer**). "several" defines the **buffer size**.

- Subsequent calls to read() will be able to **fetch bytes directly from the buffer**, which is very fast.

What about

```
write(c);
```

?

Buffered IOs

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

It's the same thing! There is on gotcha:

Sometimes, you want to immediately send the content of the buffer to the output stream.

```
os.flush();
```

Buffered IOs in Java

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- Remember the **Decorator** Design Pattern?
- Using buffered IOs is **as simple as decorating any of your byte or character streams** (don't forget about flushing buffered output streams when required!).

```
InputStream slow;
BufferedInputStream fast = new BufferedInputStream(slow);

OutputStream slow;
BufferedOutputStream fast = new BufferedOutputStream(slow);

Reader slow;
BufferedReader fast = new BufferedReader(slow);

Writer slow;
BufferedWriter fast = new BufferedWriter(slow);
```

Exercise 3

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

How can we assess the performance impact of buffered IOs in Java?

How should we **design an experiment** to answer this question?

What should we **measure**?
What **parameters** could impact the
How should we **measure** it?
How do we **present** and **analyze** the **results**?

Exercise 3

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

10’ to think about your strategy.

10’ to discuss it with others.

40’ to start the implementation.