

MINGGU 10

Praktikum Pemrograman Berbasis Objek
oleh Asisten IF2210 2019/2020

OUTLINE

1. Exception
2. Design Pattern

EXCEPTION

- Mirip dengan C++, Java juga memiliki Exception yang dapat di throw dan catch.
- Namun, object yang di-throw harus merupakan anak dari kelas Throwable. Anak kelas Throwable yang paling mudah digunakan adalah Exception

CONTOH KELAS EXCEPTION

```
class IndexOutOfBoundsException extends Exception {  
    public IndexOutOfBoundsException(String msg) {  
        super(msg);  
    }  
}
```

CONTOH PENGGUNAAN EXCEPTION

```
class Vector {  
    ...  
    public void push(int val) { ... }  
  
    public int get(int idx) throws  
    IndexOutOfBoundsException {  
        if (idx < 0 || idx >= this.size) {  
            String msg = String.format("ERROR: The  
index %d you're trying to access is  
inaccessible", idx);  
            throw new IndexOutOfBoundsException(msg);  
        }  
        return this.data[idx];  
    }  
}
```

```
class Main {  
    public static void main(String[] args) {  
        Vector v = new Vector(10);  
        v.push(19);  
        v.push(29);  
  
        try {  
            System.out.println(v.get(1));  
            System.out.println(v.get(2));  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
}
```

EXCEPTION

- Pada bagian try ... catch, perhatikan bahwa kelas yang di-catch adalah kelas Exception.
- Anda tidak harus membuat catch untuk setiap exception yang ada, melainkan bisa langsung melakukan catch terhadap kelas Exception untuk menangkap semua jenis exception yang mungkin muncul. (**ingat kembali materi polymorphism**)

DESIGN PATTERN

- Ada banyak design pattern yang dapat dipelajari. Karena itu, silakan baca slide kuliah mengenai design pattern (dapat diakses di Olympia)

Memandang design pattern (2)

- Once you have identified a potentially useful pattern, **try it out**.
- **Read the full text** of the pattern so you get a sense of its **limitations** and **important features**.
- Try to make the pattern fit, but **don't try too hard**.
- Even if the pattern wasn't the right one, you **have not** wasted your time
 - You have **learned something** about the pattern and/or about the problem.
- If a pattern does not fit exactly, **modify it**.
 - Patterns are **suggestions**, not prescriptions.

(Fowler, 1996)

Di-capture dari slide kuliah.



ADDITIONAL RESOURCES

- Referensi slide kuliah sudah banyak, tapi mungkin beberapa link ini bisa jadi tambahan bacaan sebelum tidur :)
 - <https://sourcemaking.com/>
 - <https://github.com/kamranahmedse/design-patterns-for-humans>