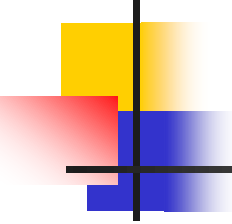


Exception

Oleh:

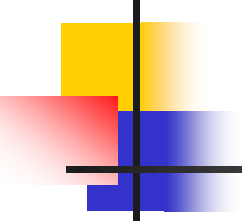
Mike Yuliana

Mochammad Zen Samsono Hadi



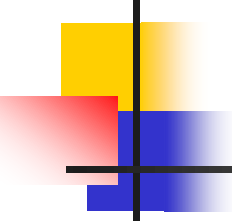
Topik

- Penanganan Eksepsi
- Menangkap Eksepsi
- Catch Secara bertingkat
- Melontarkan Eksepsi
- Melontarkan kembali Eksepsi
- Klausas Throws



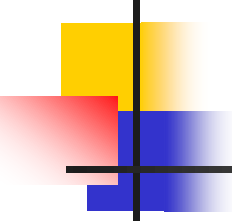
Eksepsi

- Suatu kondisi yang terjadi ketika program menemui kesalahan saat instruksi program dijalankan.
- 2 pilihan yang bisa dilakukan:
 - Menangani sendiri eksepsi tersebut
 - Meneruskannya ke luar dengan cara membuat objek yang menjelaskan eksepsi tersebut dan melemparkannya(throw) keluar agar ditangani oleh code yang memanggil method tersebut

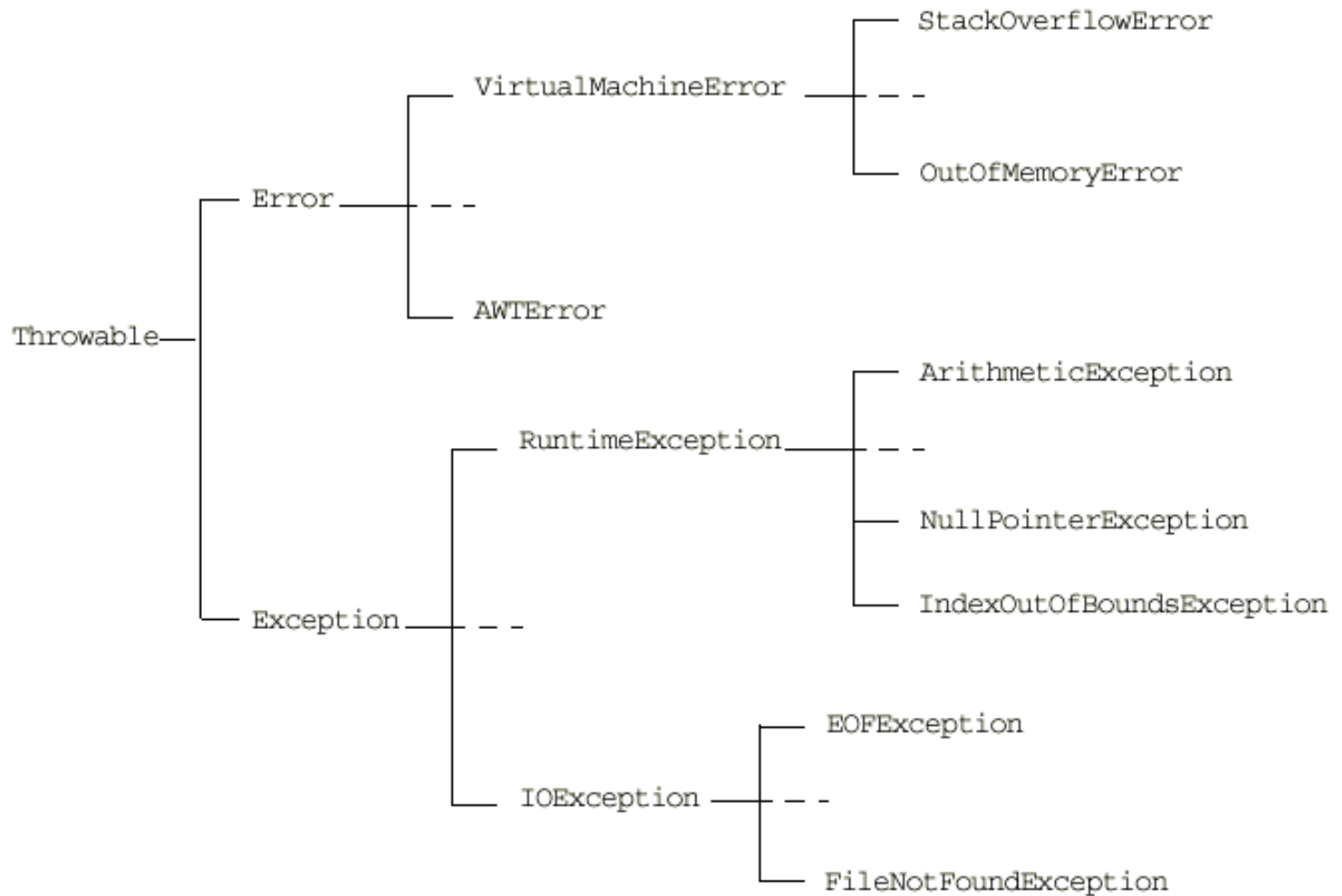


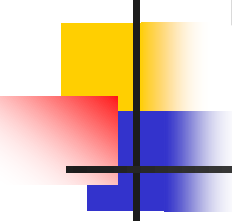
Contoh kesalahan yang terjadi:

- Pembagian bilangan dengan 0
- Pengisian elemen array diluar ukuran array
- Kegagalan koneksi database
- File yang akan dibuka tidak exist



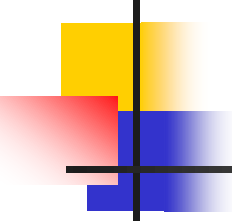
Tipe-Tipe Eksepsi





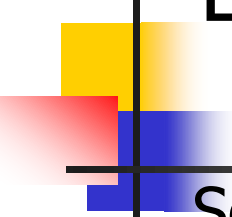
Tujuan masing-masing eksepsi

- **Error** : mengindikasikan bahwa error yang terjadi adalah fatal error (severe problem) dimana proses recovery sangat sulit dilakukan bahkan tidak mungkin dilakukan.
 - Contoh : program running out of memory
- **RuntimeException** : mengindikasikan kesalahan implementasi atau desain program.
 - Contoh : `ArrayIndexOutOfBoundsException`
- **Other exception** : mengindikasikan kesalahan environment.
 - Contoh : file not found, invalid URL exception



Common Exception

- **ArithmeticException**
 - Hasil dari operasi divide-by-zero pada integer
 - Misal : `int i = 12/0;`
- **NullPointerException**
 - Mencoba mengakses atribut atau method suatu object padahal object belum dibuat.
 - Misal : `Date d = null;`
`System.out.println(d.toString());`
- **NegativeArraySizeException**
 - Mencoba membuat array dengan ukuran negatif.
- **ArrayIndexOutOfBoundsException**
 - Mencoba mengakses elemen array dimana index nya melebihi ukuran array.
- **SecurityException**
 - Biasanya dilempar ke browser, class security manager melempar exception untuk applet yang mencoba melakukan:
 - Mengakses lokal file
 - Open socket ke host yg berbeda dgn host yg di open oleh applet



Eksepsi yang Tidak Dicek

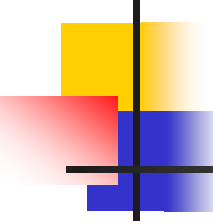
Semua eksepsi yang bertipe RuntimeException dan turunannya tidak harus secara eksplisit ditangani dalam program

Contoh program

```
class DemoEksepsi{  
    public static void main(String args[ ]){  
        int[ ] arr=new int[1];  
        System.out.println(arr[1]);  
    }  
}
```

Hasil eksekusi:

```
C:\Program Files\Java\jdk1.5.0_10\bin\java.exe -classpath "D:\modulajar\dasarpemrograma:  
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 1  
        at DemoEksepsi.main(DemoEksepsi.java:4)  
Finished executing
```

Eksepsi yang Tidak Dicek

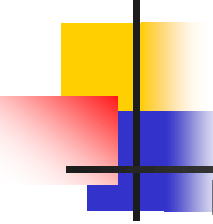
Contoh program:

```
public class BagiNol{  
    public static void main(String args[]){  
        System.out.println("sebelum pembagian");  
        System.out.println(5/0);  
        System.out.println("sesudah pembagian");  
    }  
}
```

Hasil eksekusi:

```
C:\Program Files\Java\jdk1.5.0_10\bin\java.exe -classpath "D:\modulajar\d  
sebelum pembagian  
Exception in thread "main" java.lang.ArithmeticException: / by zero  
        at BagiNol.main(BagiNol.java:4)  
Finished executing
```

Penanganan Eksepsi



Beberapa keyword yang dipergunakan untuk menangani eksepsi antara lain try, catch, catch bertingkat.

```
try{
    /* code yang mungkin mengakibatkan eksepsi*/
}
Catch(TipeEksepsi1 objekEksepsi){
    /*code untuk menangani eksepsi yang cocok dengan tipeEksepsi1*/
}
/..
Catch(tipe EksepsiN objekEksepsi){
    /* code untuk menangani eksepsi yang cocok dengan tipeEksepsiN*/
}
```

Penggunaan Blok try catch



```
class demoeksepsiok{
    public static void main(String args[]){
        try{
            int[] arr=new int[1];
            System.out.println(arr[1]);
            System.out.println("baris ini tidak akan pernah dieksekusi");
        }
        catch(ArrayIndexOutOfBoundsException e){
            System.out.println("terjadi eksepsi karena indeks di luar kapasitas array");
        }
        System.out.println("setelah blok try catch");
    }
}
```

Hasil eksekusi:

```
C:\Program Files\Java\jdk1.5.0_10\bin\java.exe -classpath "D:
terjadi eksepsi karena indeks di luar kapasitas array
setelah blok try catch
Finished executing
```

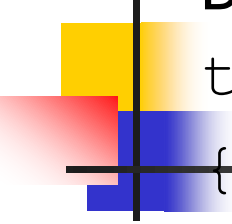
Penggunaan Blok try catch



```
public class BagiNolok{
    public static void main(String args[]){
        System.out.println("sebelum pembagian");
        try{
            System.out.println(5/0);
        }
        catch(ArithmeticException e){
            System.out.println("Terjadi eksepsi karena pembagian dengan nol");
        }
        System.out.println("sesudah pembagian");
    }
}
```

Hasil eksekusi:

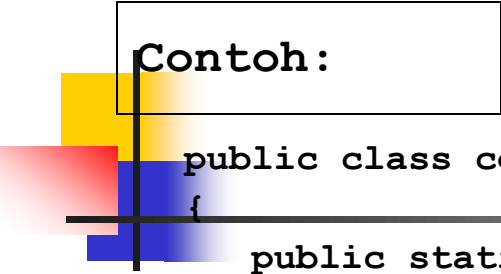
```
C:\Program Files\Java\jdk1.5.0_10\bin\java.exe -cla
sebelum pembagian
Terjadi eksepsi karena pembagian dengan nol
sesudah pembagian
Finished executing
```



Bentuk kedua pernyataan try berupa:

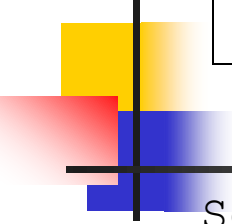
```
try
{
    // blok yang akan ditangkap sekiranya terjadi
    eksepsi
}
finally
{
    // blok yang akan dijalankan terakhir kali
}
```

Bagian finally akan dijalankan , tak tergantung apakah bagian blok try Mengalami eksepsi atau tidak



Contoh:

```
public class cobaFinal
{
    public static void main(String args[])
    {
        double bilangan=100.0;
        System.out.println("sebelum pembagian");
        for(int i=5;i>=0;i--)
        {
            try{
                System.out.println(bilangan+"/"+i+"=");
                System.out.println(bilangan/i);
            }
            finally
            {
                System.out.println("bagian finally dijalankan");
            }
        }
        System.out.println("selesai");
    }
}
```



tampilan:

Sebelum pembagian

$100.0/5=20.0$

Bagian finally dijalankan

$100.0/4=25.0$

Bagian finally dijalankan

$100.0/3=33.33333333$

Bagian finally dijalankan

$100.0/2=50.0$

Bagian finally dijalankan

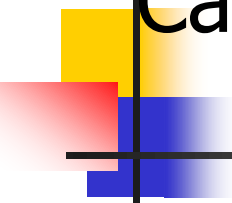
$100.0/1=100.0$

Bagian finally dijalankan

$100.0/0=\text{infinity}$

Bagian finally dijalankan

selesai



Catch Secara Bertingkat

```
try{  
    //blok yang ditangkap sekiranya terjadi eksepsi  
}  
catch(Runtime Exception e){  
    //blok yang akan dijalankan kalau terjadi eksepsi RuntimeException  
}  
catch(Exception e){  
    //blok yang akan dijalankan kalau terjadi eksepsi Exception  
}  
Catch(Throwable e){  
    //blok yang akan dijalankan kalau terjadi eksepsi lain  
}
```



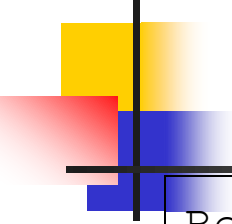

```
public class BagiNoloktingkat{
    public static void main(String args[]){
        System.out.println("sebelum pembagian");

        try{
            System.out.println(5/0);
        }
        catch(RuntimeException e){
            System.out.println("RuntimeException");
        }
        catch(Exception e){
            System.out.println("Exception");
        }
        catch(Throwable e){
            System.out.println("Throwable");
        }

        System.out.println("sesudah pembagian");
    }
}
```

```
C:\Program Files\Java\jdk1.5.0_10\bin\java.exe
sebelum pembagian
RuntimeException
sesudah pembagian
Finished executing
```

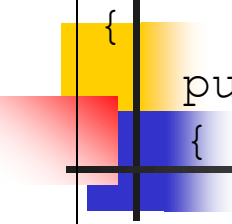
Melontarkan Eksepsi



Bentuk pernyataan:

throw variabel objek

Variabel objek menyatakan variabel objek yang merujuk ke suatu kelas Eksepsi.



```
public class efekthrow
{
    public static void main(String args[])
    {
        RuntimeException r;
        r=new RuntimeException("Eksepsi RuntimeException");
        System.out.println("sebelum throw");
        throw(r);
    }
}
```

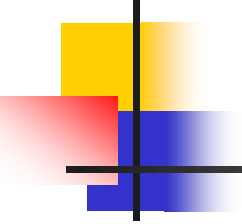
Hasil Eksekusi:

```
sebelum throw
Exception in thread "main" java.lang.RuntimeException:
Eksepsi RuntimeException
    at efekthrow.main(efekthrow.java:12)
```

Melontarkan Kembali Eksepsi

Adakalanya dalam suatu program, kita perlu melontarkan Eksepsi kembali. Kegunaannya antara lain memungkinkan metode lain untuk menangkap eksepsi tersebut.

```
public class efekthrow2
{
    public static void main(String args[])
    {
        int[] larik=new int[10];
        try
        {
            larik[50]=77; //penyebab eksepsi
        }
        catch(ArrayIndexOutOfBoundsException a){
            a=new ArrayIndexOutOfBoundsException("harus berkisar
            antara 0 sampai dengan 9");
            throw(a);
        }
    }
}
```



Hasil Eksekusi:

Output - JavaApplication2 (run-single)

init:

deps-jar:

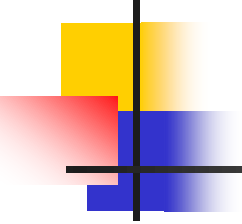
compile-single:

run-single:

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: harus berkisar antara 0 sampai dengan 9
| [at coba.main\(coba.java:12\)](#)

Java Result: 1

BUILD SUCCESSFUL (total time: 0 seconds)

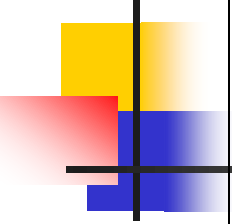


Klausu throws

klausu throws berfungsi untuk memberitahukan bahwa suatu metode ada kemungkinan menghasilkan eksepsi.

Bentuk throws:

```
public tipe1 namametode(tipe2 x,tipe3 y) throws jeniseksepsi
{
    .....
}
```

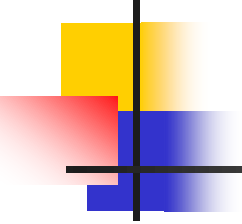


```
import java.io.*;

class coba{
    public void metodeA(){
        System.out.println("metode A");
    }
    public void metodeB()throws IOException{
        System.out.println("metode B");
    }
}

public class efekthrow{
    public static void main(String args[]){
        coba obj=new coba();
        obj.metodeA();
        obj.metodeB();
    }
}
```

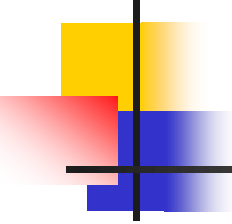
```
Error!!
Harusnya
try
{
    obj.metodeB();
}catch(IOException i) {
    System.out.println("Error");
}
```



Hasil Eksekusi:

Output - JavaApplication2 (compile-single)

```
init:
deps-jar:
Compiling 1 source file to D:\modulgenap2010\java2010\nyoba\JavaApplication2\build\classes
D:\modulgenap2010\java2010\nyoba\JavaApplication2\src\coba.java:17: unreported exception java.io.IOException; must be caught or declared to be thrown
    obj.metodeB();
    ^
1 error
BUILD FAILED (total time: 0 seconds)
```

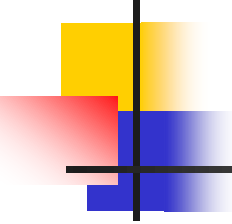



Tugas 1

- Masukkan input berupa nilai angka untuk proses pembagian berikut:
- $Y = 100/\text{angka}$

```
Scanner input = new Scanner(System.in);  
System.out.print("Masukkan angka: ");  
int angka = input.nextInt();
```

```
Masukkan angka: 0  
Tidak bisa membagi dengan nol!  
Masukkan angka: ze  
Input harus berupa angka!  
Masukkan angka: &*<br>Input harus berupa angka!  
Masukkan angka: 9  
Hasil: 11  
Data sudah benar.
```



Tugas 2

- Dari login username, hanya boleh memasukkan angka dan huruf saja

Masukkan username : zen&

Error: Username hanya boleh berisi huruf dan angka!

Masukkan username : (8)

Error: Username hanya boleh berisi huruf dan angka!

Masukkan username : zen25

Username OK: zen25