

BAB VIII EXCEPTION

Kita mengetahui bahwa tiada program yang sempurna, dan tiada pengguna program yang juga sempurna. Oleh karena itu, diperlukan suatu mekanisme yang membantu menangani error atau kesalahan yang terjadi, baik saat pembuatan maupun implementasi program. Java menyediakan mekanisme dalam pemrograman untuk menangani hal-hal tersebut yang disebut dengan *exception*.

Exception adalah event yang terjadi ketika program menemui kesalahan pada saat instruksi program dijalankan. Banyak hal yang dapat menimbulkan event ini, misalnya crash, harddisk rusak dengan tiba-tiba, sehingga program-program tidak bisa mengakses file-file tertentu. Programmer pun dapat menimbulkan event ini, misalnya dengan melakukan pembagian dengan bilangan nol, atau pengisian elemen array melebihi jumlah elemen array yang dialokasikan dan sebagainya.

Exception terdiri dari dua macam kelompok, yaitu :

- Exception yang merupakan subclass *RuntimeException*
- Exception yang bukan subclass *RuntimeException*

RuntimeException biasanya disebabkan oleh kesalahan program atau pada desain program. Misalnya *NullPointerException* yang disebabkan oleh proses inisialisasi program yang tidak sempurna dan *ArrayIndexOutOfBoundsException* yang disebabkan akses array yang melebihi kapasitas array yang ada.

Dalam bahasa Java, ketika terjadi kesalahan, otomatis akan dilemparkan sebuah objek yang disebut *exception*, yang kemudian dapat diproses lebih lanjut oleh fungsi-fungsi yang siap menangani kesalahan tersebut. Proses pelemparan exception tersebut sering dikenal dengan istilah *throwing exception*, sedangkan proses penerimaan exception yang bersangkutan dikenal dengan istilah *catch exception*.

A. Blok Try – Catch

Untuk penanganan exception, dalam Java digunakan blok *try* dan *catch*. Blok *try* digunakan untuk menempatkan kode-kode program Java yang mengandung kode program yang mungkin melemparkan exception. Blok *catch* digunakan untuk menempatkan kode-kode program Java yang digunakan untuk menangani sebuah exception tertentu.

Setelah kita tambahkan blok try – catch untuk mengatasi error yang terjadi, maka program akan menampilkan pesan error bahwa ada error yang terjadi pada konsol. Sintaks blok try – catch adalah sebagai berikut :

Try

```
{  
... kode program yang mungkin menghasilkan exception  
}  
Catch {exception xx}{...}  
Catch {exception xx}{...}
```

Latihan 34. TryCatch.java

```
import java.io.*;  
class TryCatch {  
    public static void main(String[] args) {  
        try{  
            System.out.println("Masuk 1 dari try catch");  
            File test = new File("c:\\test.txt");  
            System.out.println("Masuk 2 dari try catch");  
            test.createNewFile();  
            System.out.println("Masuk 3 dari try catch");  
        } catch (java.io.IOException e) {  
            //code untuk menangani ekpsesi  
            System.out.println("Keluar dari try catch");  
        }  
    }  
}
```



```
Command Prompt  
G:\JAVAA>javac TryCatch.java  
G:\JAVAA>java TryCatch  
Masuk 1 dari try catch  
Masuk 2 dari try catch  
Keluar dari try catch
```

B. Objek Exception

Objek exception yang dihasilkan dapat kita manfaatkan untuk mengetahui lebih lanjut mengenai error atau exception yang terjadi. Exception merupakan subclass dari class *Throwable* yang mendefinisikan beberapa method yang juga diwarisi oleh exception.

Tiga method yang penting adalah :

- getMessage()

Method ini mengembalikan isi pesan untuk menggambarkan exception yang terjadi.

- `printStackTrace()`

Method ini menampilkan pesan error dan stack trace ke standard error output stream yang biasanya merupakan konsol windows apabila program merupakan program konsol.

- `printStackTrace(PrintStream s)`

Method ini menampilkan pesan error ke objek `PrintStream` yang dijadikan parameter. Apabila ingin menampilkan pesan ke konsol, kita dapat menggunakan `System.out` sebagai parameter.

C. Blok Try – Catch – Finally

Selain try – catch, kita dapat mendefinisikan blok try – catch dan finally yang memiliki proses yang lebih lengkap, karena pada finally kita dapat mendefinisikan kode program yang selalu dieksekusi, baik ada exception yang terjadi maupun bila tidak terjadi exception sama sekali.

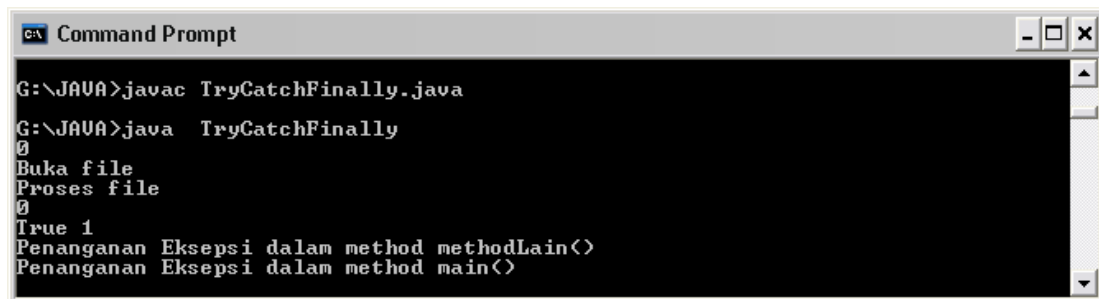
Latihan 35. TryCatchFinally.java

```
class TryCatchFinally {  
    public static void methodLain(int i)  
        throws java.io.CharConversionException {  
        System.out.println("Buka file");  
        try{  
            System.out.println("Proses file");  
            System.out.println(i);  
            if(i==0)  
            {  
                System.out.println("True 1");  
                throw new java.io.CharConversionException("Test Eksepsi");  
                //System.out.println("True 2"); error  
            }  
            else  
                System.out.println("False");  
        }catch(java.io.CharConversionException e) {  
            System.out.println("Penanganan Eksepsi dalam method methodLain()");  
            throw e; //Eksepsi dilempar ke luar method  
        }  
        System.out.println("Tutup file");  
    }  
}
```

```

}
public static void main(String[] args) {
    try{
        System.out.println(args.length);
        methodLain(args.length);
    }catch(java.io.CharConversionException e) {
        System.out.println("Penanganan Eksepsi dalam method main()");
    }
}
}
}

```



```

G:\JAVUA>javac TryCatchFinally.java
G:\JAVUA>java TryCatchFinally
0
Buka file
Proses file
0
True 1
Penanganan Eksepsi dalam method methodLain()
Penanganan Eksepsi dalam method main()

```

D. Membuat Class Exception Baru

Kita dapat membuat class baru yang mewarisi class exception dari java.lang.Exception. Kadangkala kita perlu mendefinisikan class exception yang lebih spesifik untuk keperluan tertentu, supaya penanganan exception dapat lebih baik.

LATIHAN

1. Buatlah program Java yang memiliki/menggunakan blok catch lebih dari satu !
2. Jelaskan tentang keyword 'throw' dan 'throws' , dan buatlah program java untuk keduanya !
3. Apa perbedaan antara keyword throw dengan keyword throws !
4. Jelaskan fungsi pembuatan subclass eksepsi sendiri , dan berikan contoh pemrograman java-nya !