



Exception Handling

Objectives

- Di akhir session, peserta diharapkan mampu untuk:
 - Mendefinisikan exception
 - Menangani exception dengan menggunakan try-catch-finally block
 - Membuat class Exceptions sendiri

Exceptions

- Sebuah event yang akan menginterupsi alur proses program normal dari sebuah program
- Akan mengakibatkan program terminate abnormally

Exceptions

- Error tidak harus selalu ditangani dengan exception handling (namun exception mempermudah penanganan error)
- Exception handling di Java bekerja dengan cara mengubah alur eksekusi program, sambil melempar suatu objek tertentu sebagai informasi untuk alur yang baru

Runtime exception

- Exception yang diturunkan dari RuntimeException tidak akan diperiksa pada saat kompilasi. Exception jenis ini baru dapat terdeteksi pada saat eksekusi.
- Contoh: NullPointerException, IndexOutOfBoundsException, dsb.

Contoh Exceptions

- Beberapa contoh exceptions
 - **ArrayIndexOutOfBoundsException** exception, akan terjadi jika kita mengakses suatu array di index yang tidak valid
 - **NumberFormatException**, akan terjadi jika kita melakukan passing parameter non-number ke method **Integer.parseInt()**

Penanganan Exceptions

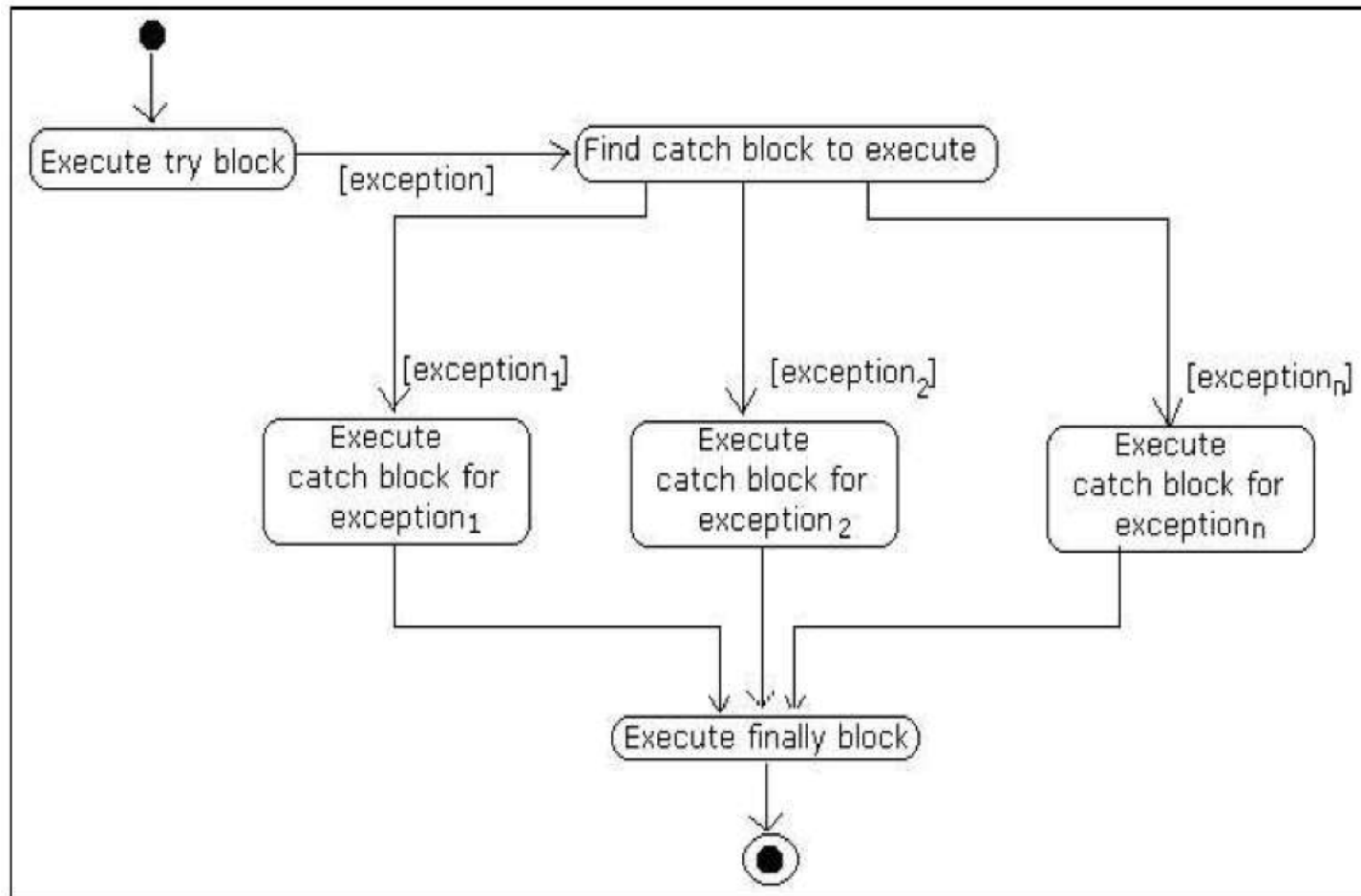
- Untuk menangani exceptions di Java digunakan blok **try-catch-finally**
- Statement program yang memungkinkan terjadinya exceptions harus diletakkan dalam blok **try-catch-finally**

Blok try-catch-finally

- Bentuk umum dari blok try-catch-finally



```
try{  
    //write the statement that can generate an exceptions  
    //in this block  
}  
catch(<exceptionType1> <varName1>){  
    //write the action your program will do if an exception of certain  
    //type occurred  
}  
...  
}  
catch(<exceptionTypen> <varNamen>){  
    //write the action your program will do if an exception of certain  
    //type occurred  
}  
finally{  
    //add more cleanup code here  
}
```


Alur Program



Alur Normal (tanpa exception)

```
try {  
    aksi_1();  
    aksi_2();  
    aksi_3();  
} catch (ExceptionClassName name){  
    //penanganan error (tidak dieksekusi)  
}  
aksi_4();  
aksi_5();
```



Exception Dilempar aksi_2()

```
try {  
    aksi_1();  
    aksi_2();  
    aksi_3(); //tidak dieksekusi  
} catch (ExceptionClassName name){  
    //penanganan error (dieksekusi)  
}  
aksi_4();  
aksi_5();
```



Contoh Penggunaan Exception

```
import java.util.Scanner; /*JDK 1.5*/
class ContohException {
    public static void main (String args[]) {
        Scanner sc = new Scanner(System.in);
        try {
            System.out.print("Masukkan Angka:");
            int num = sc.nextInt();
            if (num>10) throw new Exception();
            System.out.println("Angka kurang dari atau sama
                dengan 10");
        } catch (Exception s) {
            System.out.println( "Angka Lebih dari 10");
        }
        System.out.println( "Selesai");
    }
}
```

Catatan

- Dalam statement

- `throw new Exception()`, sebuah instans dari kelas `Exception()` diciptakan (konstruktor defaultnya dipanggil)

- Jika kelas memiliki konstruktor yang lain, konstruktor itu dapat dipanggil:

- `throw new Exception("lebih dari 12");`

- Dalam catch, method kelas exception dapat dipanggil:

- `System.out.println(e.getMessage());`

Nested Exception Handling

- Boleh ada blok try-catch lain dalam suatu try-catch, penangkap adalah blok terdekat

```
try {  
    fungsi_a();  
    try {  
        fungsi(b);  
    } catch (NumberFormatException n) {  
    }  
    } catch (FatalException fatal) { }
```

Melempar kembali exception

- Dalam catch, exception bisa dithrow:

```
catch (Exception e) {throw e;}
```

- Exception baru bisa dibuat:

```
catch (FileNotFoundException e) {  
    throw new FatalException();  
}
```

- Gunanya melempar kembali agar exception ditangkap oleh klausa exception terluar, atau method yang memanggil method ini.

Exception pada method

```
void connect_internet() throws connect_exception{  
    //coba melakukan koneksi  
    if (error) throw new connect_exception("error");  
    //....  
}  
  
//...  
//bagian main  
try {  
    connect_internet();  
} catch (connect_exception e) {  
    /*do something*/  
}
```


Catatan

- Exception adalah *nama kelas* exception standar milik Java
- Programmer bisa membuat sendiri kelas Exception dengan mengimplementasikan Throwable, atau (yang lebih disarankan) menurunkan dari `java.lang.Exception`

Membuat Class Exceptions

- Buat class exception yang diturunkan dari class exception lain yang lebih umum
 - Misal class **OutOfDiskSpaceException** bisa diturunkan dari **IOException**
- Sebaiknya turunkan dari class **Exception** karena sudah memiliki method untuk mencatat pesan exception

Contoh: SmallIntExcept

```
class SmallIntExcept extends Exception
{
    private static int num_except;
    SmallIntExcept(String msg) {
        super(msg);
        num_except++;
    }
    static int numException () {
        return num_except;
    }
    void response () {
        System.out.println(getMessage());
    }
};
```

Class SmallInt

```
class SmallInt{
    int value;
    SmallInt(int val){
        value = val;
    }
    void plus(SmallInt X) throws SmallIntExcept{
        value = value + X.value;
        if (value > 10)      throw new SmallIntExcept ("TOO BIG");
        if (value < 0)       throw new SmallIntExcept ("TOO SMALL");
    }
    public String toString() {
        return Integer.toString(value);
    }
    void ReadVal () {
        Scanner s = new Scanner(Sytem.in);
        value = s.nextInt();
    }
}
```

Main Program

```
class SmallIntExample {  
    public static void main (String args[]) {  
        System.out.println("start of smallint ...");  
        SmallInt S1= new SmallInt(1);  
        SmallInt S = new SmallInt();  
        S.ReadVal ();  
        try {  
            S1.plus (S);  
            System.out.println("hasil S1= S1+S =" +S1);  
        }catch (SmallIntExcept e) {  
            e.response ();  
        }  
    }  
}
```

Summary

- Mendefinisikan exception
- Menangani exception dengan menggunakan blok **try-catch-finally**
- Membuat class exception `SmallIntExcept` subclass dari `Exception`