Sabancı Üniversitesi

Object Oriented Programming with Java

06 - Exception Handling

Sabancı Üniversitesi

What is an Exception?

- There are two major error types in Java:
  - Compiler Error : Errors catchable before compile
    - Ex: case errors, object name errors, syntax errors…
  - Exception: Errors that occur at runtime
    - Ex: System errors, device errors, file not found errors…

Sabancı Üniversitesi

try - catch Block

- When an exceptional event occurs in Java, an exception is said to be “thrown”
- The code that is responsible for dealing with the exception is “exception handler”
- Exception handling works by transferring the execution of a program to an appropriate exception handler when exception occurs

```
try{
//put code here that might cause exception
}
catch(MyFirstException){
//Put code here that handles this exception
}
catch(MySecondException){
//Put code here that handles this exception
}
```

Guarded Region

Exception Handler

Exception Handler

Sabancı Üniversitesi

Using `finally`

- A finally block encloses code that is always executed at some point after try block
- It executes even an exception thrown or not
- Used for cleaning resources after exception thrown

```
try{
//put code here that might cause exception
}
catch(MyFirstException){
//Put code here that handles this exception
}
finally{
//Put code here to clear resources
}
```

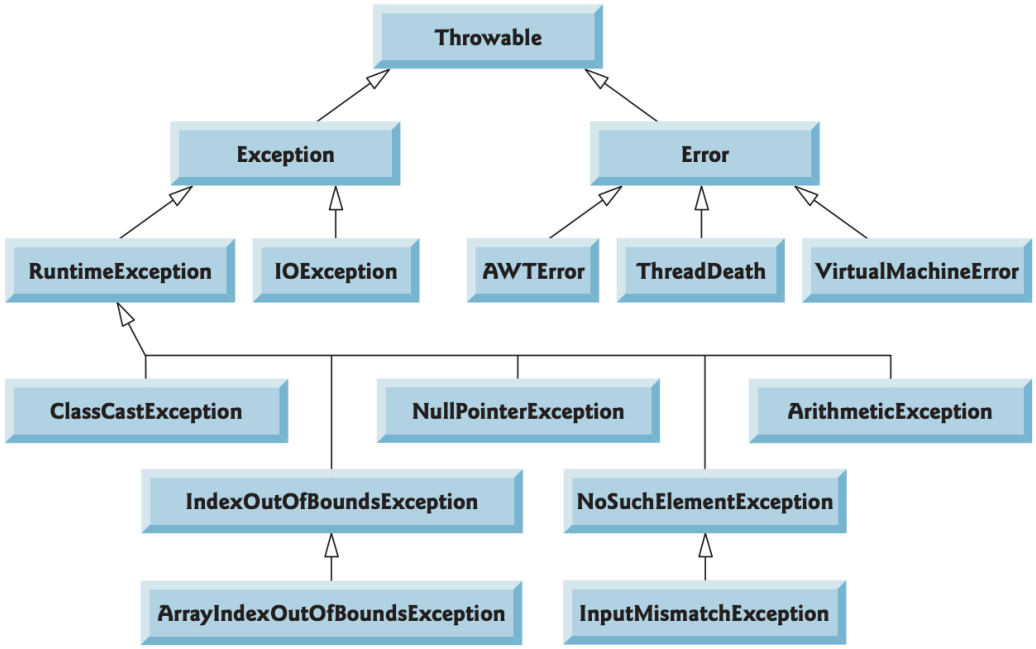
Also legal:

```
try{
//do stuff
}
finally{
//clean up
}
```

Sabancı Üniversitesi

Exception Hierarchy

- All errors and exceptions inherit from the Throwable class
  - Error: typically unrecoverable low level system problems (Ex, Hardware problems)
  - Exceptions are used for programming errors



Checked vs Unchecked Exceptions

- Unchecked → all subtypes of RuntimeException
- Checked → all other throwable types
- Compiler requires that all *checked* exceptions be handled

```
public void doSomething() {
        try{
                anExceptionalMethod();
        }catch(SomeException e){}
}
```

Unchecked exceptions are programmer errors, impossible to recover, ex. NullPointerException, ArithmeticException

- or can declare that exception may go up the caller

```
public void doSomething() throws SomeException{
        anExceptionalMethod();
}
```

We don't have to handle the exception in try-catch

Sabancı Üniversitesi

Order of Catching Exceptions

- A catch block can specify a super type exception
- Catch block sequence must start with subtypes

```
try{
//put code here that might cause exception
}
catch(IndexOutOfBoundsException ie){
ie.printStackTrace();
}
catch(Exception ex){
ex.printStackTrace();
}
```

catch both ArrayIndexOutOfBoundsException and StringIndexOutOfBoundsException which are subtypes

catch general exception after specific one

Sabancı Üniversitesi

Exception Declaration

- The `throws` keyword is used as follows to list the exceptions that a method can throw:

```
void myFunction() throws MyException1, MyException2{
if(somethingWentWrong)
        throw new MyException1("Something went wrong");
}
```

- If an exception is thrown from a method it should also be declared by the method (if it is a checked exception)
- If declared and checked it should be handled in try-catch

Sabancı Üniversitesi

Creating Your Own Exceptions

- Since `Exception` is a class, we can subclass it or any of its subclasses to create our own exceptions

```
public class MyException extends Exception{



}
```

Sabancı Üniversitesi

Re-throwing an Exception

- An exception handler may re-throw the exception it just caught

```
public void myMethod() throws SomeException{
        try{
                myOtherMethod(); //also throw SomeException
        }catch(SomeException ex){
                System.err.println("SomeException occured");
                throw ex;
        }
}
```

Sabancı Üniversitesi

Overriding Methods

- The overriding method can throw any unchecked (runtime) exception, regardless of whether the overridden method declares the exception
- The overriding method must not throw checked exceptions that are new or broader those of the overridden method
- An overriding method doesn't have to declare any exceptions that will never throw, regardless of what the overridden method declared