

Exception (OHOHOH)

Hsuan-Tien Lin

Department of CSIE, NTU


OOP Class, May 03-04, 2010

Exception Class Hierachy

```
1 public class java.lang.Throwable;  
2 public class java.lang.Exception  
3     extends Throwable;  
4 public class java.lang.Error  
5     extends Throwable;  
6 public class java.lang.RuntimeException  
7     extends Exception;  
8 public class java.lang.NullPointerException  
9     extends RuntimeException;  
10 public class java.lang.ClassNotFoundException  
11     extends Exception;  
12 public abstract class java.lang.VirtualMachineError  
13     extends Error;
```

Throwable
Exception
Error
RTE
CFE
VME

try-throw-catch-finally

```
1      try{  
2          line_1;   
3          throw new RuntimeException("ohohohohohoh");  
4          line_2;  
5      }  
6      catch(NullPointerException e){  
7          line_3;  
8      }  
9      catch(RuntimeException e){  
10         line_4;  
11     }  
12     finally{  
13         line_5;  
14     }
```

line-6

- if NPE happens at line 1: [1, 3, 5]
- if RTE happens at line 1: [1], 4, 5

What is the difference between the `finally` block and just putting lines afterwards?

try-throw-catch-finally

```
1      try{
2          line_1;
3          throw new Exception("ohohohohohoh");
4          line_2; ← unreachable
5      }
6      catch (NullPointerException e){
7          line_3;
8      }
9      catch (RuntimeException e){
10         line_4;
11     }
12     finally{
13         line_5;
14     }
15     line_6;
```

- if NPE happens at line 1: [1], 3, 5, 6
- if nothing at line 1: 1, 5, caller-exception-handlers

A More Complicated Example

```
1   for(int i = 0; i < 2; i++){
2       try{
3           System.out.println("try");
4           if (i > 0) throw new Exception();
5       }
6       catch(Exception e){
7           System.out.println("catch");
8           return; //continue? break?
9       }
10      finally{
11          System.out.println("finally");
12      }
13
14      System.out.println("afterwards");
15  }
```

Putting It All Together (from Java Tutorial)

```
1     public void WriteList(){
2         PrintWriter out = null;
3
4         try {
5             System.out.println("try");
6             out = new PrintWriter(
7                 new FileWriter("output.txt");
8             );
9             for(i=0;i<10;i++)
10                out.println(vector.elementAt(i));
11        }
12        catch (ArrayIndexOutOfBoundsException e){
13            System.out.println(e);
14        }
15        catch (IOException e){
16            System.out.println("IO_Exception:" + e);
17        }
18        finally{
19            if (out != null)
20                out.close();
21        }
```

throws

```
1 public double divide(int a, int b)
2     throws DivisionByZeroException
3     {
4         if (b == 0)
5             throw new DivisionByZeroException("...");
6     }
7
8     try {
9         divide(m, n);
10    }
11    catch (DivisionByZeroException e) {
12    }
```

- “warn” the component user that an exception may be thrown
- compiler helps check if someone handles it (hahaha if not)
- what gets checked? everything other than RuntimeException and Error

throws

```
1 public double divide(int a, int b)
2     throws DivisionByZeroException, LalalaException
3 {
4     if (b == 0)
5         throw new DivisionByZeroException("...");
6 }
7
8 public void caller() throws LalalaException{
9     try{
10        divide(m, n);
11    }
12    catch(DivisionByZeroException e){
13    }
14 }
```

- if the exception needs checking: caller needs to handle it (catch) or “warn” people (throws)