innopolis university

Exceptions and testing

Stanislav I. Protasov

30/07/2015

Splitter example

```
package files;
import java.util.Scanner;

public class Splitter {
    public static void main(String[] args) {
        String line = "";
        // create scanner to read text
        Scanner sc = new Scanner(System.in);
        do {
            // read next line
            line = sc.hasNextLine() ? sc.nextLine() : "";
            // convert line to upper case and write to standard output
            System.out.println(line.toUpperCase());
            // convert line to lower case and write to standard error
            System.err.println(line.toLowerCase());
        } while (!line.equals(""));
        sc.close();
    }
}
```

Serializable

```
import java.io.Serializable;

public class Class1 implements Serializable {
    private static final long serialVersionUID = 0x8FFFFFFFFFFFFFFFL;
    int v = 4;
}
```

```
FileOutputStream fs = new FileOutputStream("D:\\file.ser");
ObjectOutputStream os = new ObjectOutputStream(fs);
os.writeObject(new Class1());
os.flush();
os.close();
```

```
AC ED 00 05 73 72 00 06 43 6C 61 73 73 31 8F FF
FF FF FF FF FF FF 02 00 01 49 00 01 76 78 70 00
00 00 04
```

```
¬í ♣sr ♠Class1SS3ÿ
ÿÿÿÿÿÿ☻ ☺I ☺vxp
♦
```

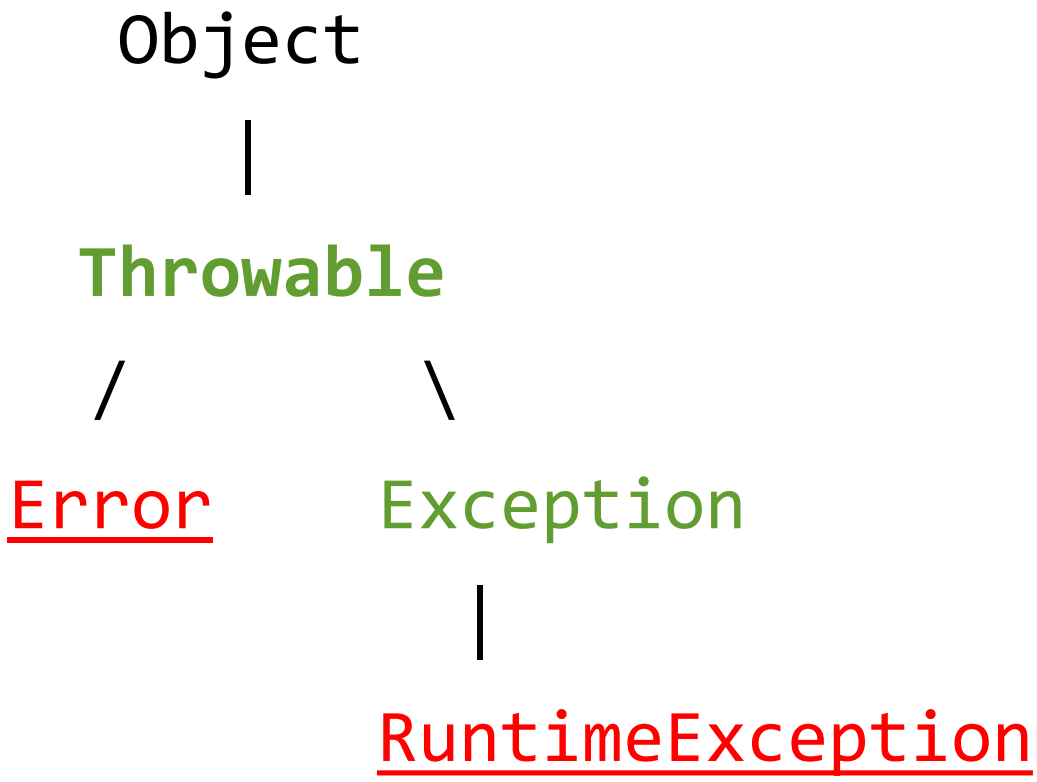
Agenda

- Exceptions
  - Error, Exceptions and handling
- Unit Tests

Problems (errors) vs Exceptions

- **"Problem" – situation when your program behaves not in expected way**
  - **2 + 2 = 5**
  - **2 + 2 = 4 … takes 20 seconds to calculate**
  - **1 / 0**
  - **File.open("???123321\5431`");**
  - **a = null;**
    a.equals(b);
- **Exception – problem, that can be detected as something "not expected to happen" and handled by your code**

Java Exception hierarchy

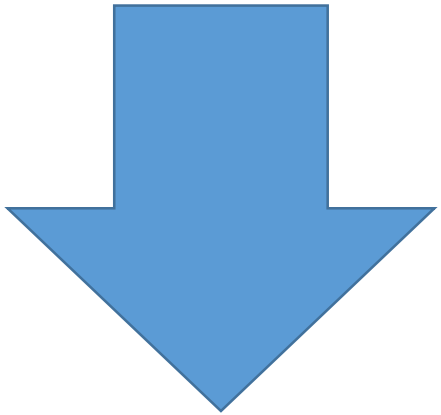


Java Exception paradigm

- Classifications
  - Compilation errors vs. runtime exceptions
  - Errors vs Exception
    - Errors are either compilation errors or serious problems that should not be handled
  - Checked ("predictable") vs Unchecked exceptions
    - All Errors are Unchecked exceptions

Java compile-time errors

```
6        int variable = new Object();
```



```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
        Type mismatch: cannot convert from Object to int

        at Orders.q(Orders.java:6)
        at Orders.main(Orders.java:11)
```

Java compile-time errors

```
4  static int q() {
5      int value = 3;
6  }
```

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
        This method must return a result of type int

        at Orders.q(Orders.java:4)
        at Orders.main(Orders.java:9)
```

Java runtime-time exceptions

```
static int usual() { return 1; }
static int fun() { while (true); }

static int q() { throw new ArithmeticException(); }
static int qGuarded() { return q(); }

static int p() throws Exception { throw new Exception(); }

static int pGuarded() {
    int val = 0;
    try {
        val = p();
        System.out.println("Everything is ok!");
        return val;
    } catch (Exception ex) {
        System.out.println("Ooops!");
        return 0;
    } finally {
        System.out.println("In any case ");
    }
}

public static void main(String[] args) {
    pGuarded();
    qGuarded();
}
```

Unchecked exception

Checked exception

Check

Try block

Catch blocks

Finally block

```
Exception in thread "main" Ooops!
In any case
java.lang.ArithmeticException
        at Orders.q(Orders.java:6)
        at Orders.qGuarded(Orders.java:7)
        at Orders.main(Orders.java:27)
```

Try – catch – finally

```
try {
      // ...
} catch (Exception e) {
      // A1: only if extends Exception
} catch (Throwable e) {
      // A2: cannot switch with A1!
} finally {
      // F : executed after catch
}
```

Try – catch – finally: special cases

```
try {

} finally {

}
```

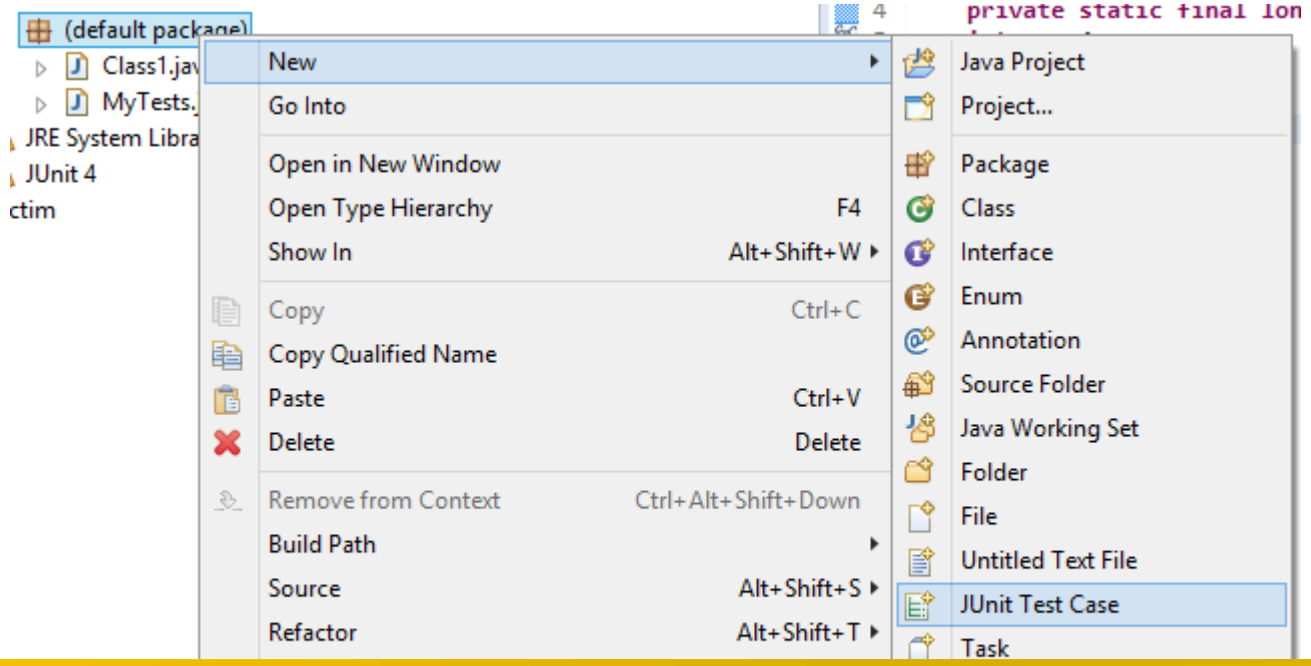
```
static int pGuarded() throws Exception {
    try {
        throw new Exception("A");
    } catch (Exception e) {
        throw new Exception("B");
    } finally {
        throw new Exception("C");
    }
}
```

Checked Exception

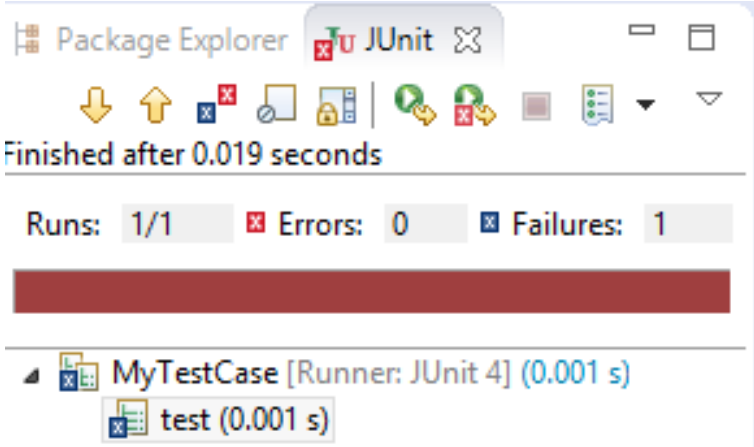
```
class A {
    void a() throws Exception { }
    void b() throws Exception { a(); }
    void c() throws Throwable { a(); }
    void d() throws ArithmeticException { a(); }
}

class B extends A {
    void a() throws Throwable { }
}

class C extends A {
    void a() throws ExportException { }
}
```

Unit testing

- **Unit testing – an approach in programming, that allows to check if parts (units) of your program behaves right in automatic way**
- **There are few frameworks to implement Unit test. *JUnit* is pre-installed for Eclipse.**



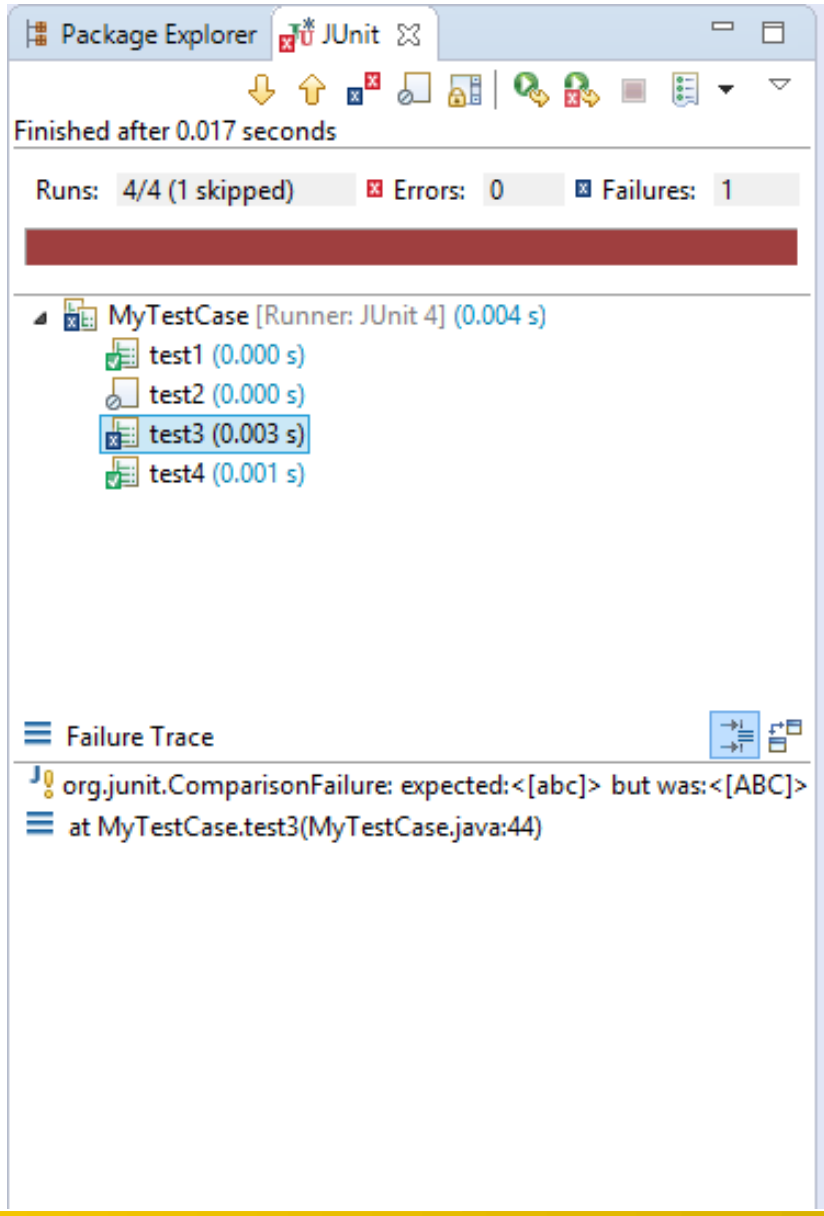
Running test



MyTestCase.java

```
1  import static org.junit.Assert.*;
8
9  public class MyTestCase {
10
11     @BeforeClass
12     public static void setUpBeforeClass() throws Exception {
13     }
14
15     @AfterClass
16     public static void tearDownAfterClass() throws Exception {
17     }
18
19     @Before
20     public void setUp() throws Exception {
21     }
22
23     @After
24     public void tearDown() throws Exception {
25     }
26
27     @Test
28     public void test() {
29         fail("Not yet implemented");
30     }
31
32 }
```

Typical test case



MyTestCase.java

```
22
23      @Test
24      public void test1() {
25          // Arrange
26          String string = "abc";
27          // Act
28          String result = string.toUpperCase();
29          // Assert
30          assertEquals("ABC", result);
31      }
32
33      @Test
34      @Ignore
35      public void test2() { }
36
37      @Test
38      public void test3() {
39          // Arrange
40          String string = "abc";
41          // Act
42          String result = string.toUpperCase();
43          // Assert
44          assertEquals("abc", result);
45      }
46
47      @Test(expected = ArithmeticException.class)
48      public void test4() {
49          // Arrange
50          int a = 0, b = 12;
51          // Act
52          int c = b / a;
53          // Assert
54          fail("Exception expected!");
55      }
56  }
```

INNOPOLIS UNIVERSITY

s.protasov@innopolis.ru