

CS-152: Software Testing

Neal Holtschulte

March 16, 2021

Software Testing Outline

- ▶ Terminology
- ▶ Assertions and when to use them
- ▶ Try-catch and when to use them
- ▶ What are Exceptions
- ▶ Further resources
- ▶ Practice

Terminology

Unit testing - testing small pieces of code (such as individual methods or functions).

A collection of unit tests is known as a **test suite**.

Regression testing is the process of repeatedly running tests.

Source: Java An Introduction to Problem Solving and Programming by Savitch. Page 336.

Terminology

A **test case** is an input output pair that we know to be correct. If we give our program (or a method of our program) the input and we do not get the expected output, then the program (or method) has failed the test case and we may have discovered a bug.

Negative test - a test to make sure the system doesn't crash if given an unexpected input.

Tracing

Tracing variables means watching the variables change value while the program is running.

Often there are debugging utilities that will help you do this, but we're just going to use *System.out.println*.

Tracing

Since we don't want all these print statements in the final version of our program we can use a

```
public static final boolean DEBUG = true;
```

Then you can wrap all your print statements in ifs and set DEBUG to false when you're done testing.

```
if (DEBUG){ System.out.println("value's value is " +  
                                value); }
```

Assertions

Assertions can be used to verify that certain conditions are true and make software more robust.

```
/**
 * Get the largest integer value from myArray.
 */
public int getLargest(int[] myArray){
    //First check to make sure the array is
    //not empty.
    assert(myArray.length > 0);
    ...
}
```

Assertions

- ▶ Assertions can and should be used throughout your code to verify that any important conditions hold.
- ▶ Assertions provide a form of **enforced documentation**.
- ▶ Assertions help you find bugs faster.
- ▶ Assertions do not (usually) noticeably slow down software.

See `TestingExample.java`

Try-catch

Try-catch statements can be used to throw more useful error messages or to make a program more robust by handling rare special cases.

Java An Introduction to Problem Solving and Programming by Savitch.

Chapter 9.

Try-catch

```
public Sandwich makePBJ(){  
    PeanutButter pb;  
    try{  
        pb = getPB();  
        if(pb.isEmpty()){  
            throw new Exception("EXCEPTION: No "+  
                                "Peanut Butter!");  
        }  
    } catch (Exception e){  
        System.out.println(e.getMessage());  
        System.out.println("Go to the store and "+  
                            "buy peanut butter.");  
    }  
}
```

Try-catch

Try-catch blocks are required around file readers because the file might not be found.

Try-catch

```
try{  
    //Create a new file reader and pass it the  
    //file to read in.  
    FileReader fr = new FileReader("english.0");  
    5    BufferedReader bReader = new BufferedReader(fr);  
    String line = bReader.readLine();  
    while(line != null){  
        text += line+",";   
        line = bReader.readLine();  
    10    }  
    bReader.close();  
}  
catch(Exception e){  
    15    System.out.println("ERROR: There was a problem "+  
        "reading the file.\n"+e.getMessage());  
    System.exit(1);  
}
```

Try-catch

Use a try-catch block whenever there is a special case that needs to be handled differently.

Use a try-catch to trigger additional data-gathering when an exception occurs to help debug your code. For example:

Try-catch

```
try{  
    myStack.pop();  
}  
catch(Exception e){  
    System.out.println(e.getMessage());  
    myStack.printContents();  
    System.exit(1);  
}
```

What are exceptions?

- ▶ An exception signals an unusual event during execution.
- ▶ Creating an exception is called “throwing an exception”.
- ▶ Code that deals with an exception is said to “handle an exception”.

Java An Introduction to Problem Solving and Programming by Savitch.

Page 659.

What are exceptions?

Exceptions are objects that can be created and thrown

```
throw new Exception("EXCEPTION: No Peanut Butter!");
```

You can see what methods are available to be called on exceptions with Eclipse's autocomplete feature.

Testing advice

- ▶ Automated your testing process
- ▶ Make test output self-identifying
- ▶ Make tests reproducible
- ▶ Add a test for each bug
- ▶ Add tests for new features and changes
- ▶ Never throw away a test

Further information about the above advice can be found here:

<http://www.cs.princeton.edu/~bwk/testing.html>

Further resources

- ▶ Software Testing (free online course)

<https://www.udacity.com/course/cs258>


- ▶ Java Exceptions tutorial

<http://docs.oracle.com/javase/tutorial/essential/exceptions/definition.html>

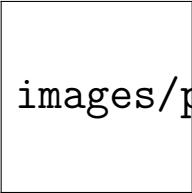
Software Testing Outline

- ▶ Terminology
- ▶ Assertions and when to use them
- ▶ Try-catch and when to use it
- ▶ What are Exceptions
- ▶ Further resources
- ▶ Practice

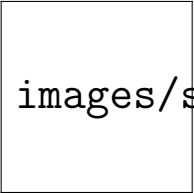
Before we practice, Stacks



images/stack.jpg



images/pez.jpg



images/stack1.jpg

Let's ...

- ▶ look at the stack code
 - ▶ where might we insert assertions?
 - ▶ what can go wrong?
- ▶ test the constructor
- ▶ discuss if negative tests are reasonable
 - ▶ use an assertion
 - ▶ use a try-catch
- ▶ divide up and write tests for the buggy stack implementations

Let's ...

To Eclipse.

`code/StackTestingCode/`