Apache Commons Lang Testing

Group Name: Hello World

Group Members: Zicheng Shan, Chenxu Wang

Date: Feb 2, 2022

Introduction

The tool we are going to test is Apache Commons Lang which provides some additional method tools for manipulating its core classes since the standard Java library does not provide enough methods. Apache Commons Lang provides a large number of helpers for the java.lang API, such as string manipulation, basic numeric methods, and object reflection. By using Apache Commons Lang, we would be able to manipulate data more efficiently (Team, C., 2021).

Apache Commons Lang contains over 100K lines of code and 97.7% of them are written in Java.

Set-Up

How to build

1. Clone project from GitHub by the following path: https://github.com/apache/commons-lang
2. Open the project and reload the Maven project.

How to run

Add the following dependency to the pom.xml file

```
<dependency>
  <groupId>org.apache.commons</groupId>
  <artifactId>commons-lang3</artifactId>
  <version>3.12.0</version>
</dependency>
```

Existing Test Cases

The tests of the project are based on JUnit. They are all in src/test/java/org/apache/commons/lang3. They are divided into different folders. Such as ‘builder’, which contains test cases about builder function, ‘compare’ which contains test cases about comparing function, and ComparableUtils. Other files like

‘ArrayUtilsAddTest’, ‘ArrayUtilsInsertTest’, ‘CharSetUtilsTest’ contains test cases correspond to the relative utils. The existing test cases provide a lot of good partition testing examples. For example, ArrayUtilsInsertTest was partitioned by the data types and then partitioned by the inserting position. It splits the test region efficiently and finally, it would be able to cover most situations. As of February 2, 2022, there are 7893 test cases.

Functional and Partition Testing

Functional testing is the process by which quality assurance personnel determine whether a piece of software operates according to predetermined requirements. It usually uses black-box testing techniques, where the tester does not need to know the logic of the internal system. And functional testing is only concerned with verifying that the system works as expected (Bose, S., 2021). Figure 1 shows the build process from specification to test cases.

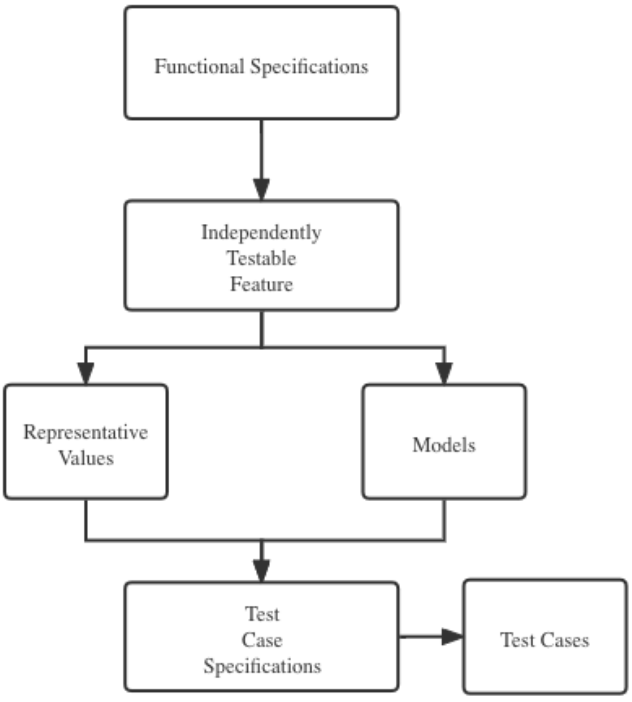

Figure1 From specification to test cases

Partition testing:

Equivalent partitioning is a software testing technique that divides the input data of a software unit into equivalent data partitions and then summarizes test cases from each of the different partitions. Failures are sparse in the space of possible inputs and can be dense in some parts of the space. When it comes to our own testing cases, it is important to split the test region efficiently.

Test Example

StringUtils.IsAlpha(final CharSequence cs)

IsAlpha(final CharSequence cs) is used to check whether the CharSequence contains only Unicode letters. It is selected to do the test and can be found by the following path: *src/main/java/org/apache/commons/lang3/StringUtils.java*

Partition

In general, the test can be partitioned into two parts: the input is null or not null
When the input is not null, it has two dimensions: the type of input and length of the input

For type, it can be partitioned into space, number, letter (uppercase & lowercase), symbols, and mix

For length, it can be partitioned into three ranges; 0, 0-maxLength, maxLength

ArrayUtils.add(final int[] array, final int index, final int element)

add(final int[] array, final int index, final int element) is used to add an integer to the array at the corresponding index and return a new array containing the existing elements and the new element. It is selected to do the test and can be found by the following path: *src/main/java/org/apache/commons/lang3/ArrayUtils.java*

Partition

The test is aimed to test add function for the integer array. We test it by partitioning the position by adding an integer to first, mid, last, or out of bounds.

The test file (SWE261_P1_Test.java) our team created can be found by the following path: https://github.com/chenxu-wang/commons-lang/blob/master/src/test/java/org/apache/commons/lang3/SWE261_P1_Test.java

These test cases can be run by clicking the green triangle run button.

Reference

Team, C., 2021. Lang – Home. [online] Commons.apache.org. Available at <https://commons.apache.org/proper/commons-lang/ > [Accessed 2 February 2022].

Bose, S., 2021. Functional Testing: A Detailed Guide. [online] BrowserStack. Available at: <https://www.browserstack.com/guide/functional-testing > [Accessed 2 February 2022].