Apache Commons Lang Testing

Group Name: Hello World

Group Members: Zicheng Shan, Chenxu Wang

Date: Feb 2, 2022

Introduction

Apache Commons Lang provides a host of helper utilities for the java.lang API, notably String manipulation methods, basic numerical methods, object reflection, concurrency, creation and serialization and System properties. Additionally, it contains basic enhancements to java.util.Date and a series of utilities dedicated to help with building methods, such as hashCode, toString and equals (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 the repository to the local
2. Use IDE (IntelliJ IDEA) to import the project as a Maven project
3. Right Click at the pom.xml file and choose Maven-> Reload Project

How to run

Add the following dependency to pom.xml

```
<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. As of February 2, 2022, there are 7893 test cases.

Functional and Partition Testing

Functional testing is the process through which QAs determine if a piece of software is acting in accordance with pre-determined requirements. It uses black-box testing techniques, in which the tester has no knowledge of the internal system logic. Functional testing is only concerned with validating if a system works as intended (Bose, S., 2021). In the simplest words, functional testing checks an application, website, or system to ensure that it is doing exactly what it is meant to. It is better than other forms of testing (particularly structural testing) for missing logic.

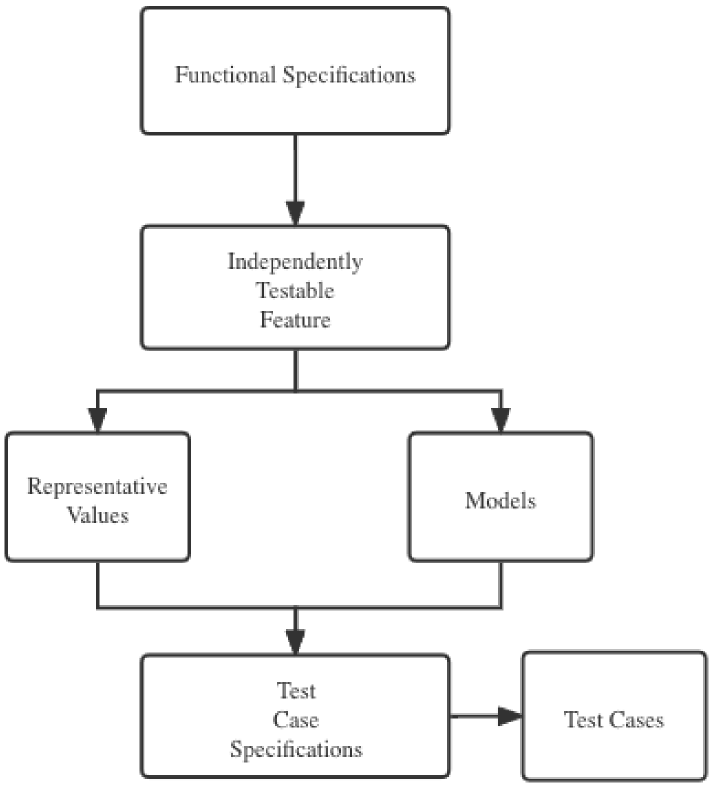

Figure1 From specification to test cases

Partition testing:

Equivalence Partition is a software testing technique that divides the input data of a software unit into partitions of equivalent data from which test cases can be derived. 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 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 integer array. We test it by partitioning the position by adding an integer to first, mid, last, or out of bounds.

The test file we created is in src/test/java/org/apache/commons/lang3/SWE261_P1_Test.java. The test cases can be run by clicking the green triangle run button to run the test cases.

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].