


Google Summer of Code

# GWT Unit Testing

Proposed by **Reshu Kumari**

# 1. About me

Name: Reshu Kumari

College: Banasthali Vidyapith

Branch: Computer Science

Email: reshukumari786@gmail.com

Country: India

Timezone: Asia/Kolkata (UTC +05:30)

Github: ReshuKumari

Linkedin: Reshu Kumari

# 2. Interest in MIT App Inventor

I am interested in contributing to MIT App Inventor because I felt the idea of the MIT App to be very unique and Impactful. People with no technical background can use it to make applications. We can use this in multiple ways for making educational apps, fun applications, productivity applications like to-do lists, etc. Someone who wants to convert their idea into execution can rely on MIT App Inventor. It is very fast and easy to build apps. Also, The community of MIT App Inventor is very supportive.

# 3. Interest In Introductory Programming

I have been part of college clubs and held sessions for juniors. I held sessions on technologies like HTML, CSS, and javascript.

# 4. Experience with development tools

I am a final year undergrad student and have worked on multiple technologies and projects. I have worked in languages and technologies like C++, Java,

Angular, Javascript, Typescript, and Spring Boot, HTML, CSS, PHP. I like solving programming challenges and working on projects.
It feels great to build things that people will use and will make their life easier. It gives me immense satisfaction after completing the planned tasks, and when people use our created product or additions we do to that product.

## 5. Experience with teams, online developer communities, and large codebases

I used to contribute to open source earlier and all my pull requests can be viewed here. I was a member of that organization and also a part of one of their teams. I have also written unit tests for the frontend part for the changes made. Therefore, I have experience working with large codebases. I have also done multiple internships and worked on different technologies like Minio, and Keyclock and built microservices and REST apis in Java from scratch. I also built a dashboard in one of my internships from scratch both frontend and the backend. From all my experiences I learned collaboration, time management, and planning.

## 6. Abstract

MIT App Inventor is built in GWT. There is no systematic code coverage metric for the AppEngine client. So the project aims to set up a code coverage metric and test the client code of AppEngine.

Unit testing helps us in strengthening the application. When we add new features then testing ensures that we don't break existing features. Increase

product quality and enhance the development process. Unit testing ensures the stability of its component modules in large applications.

Code coverage increases the effectiveness of testing. Increasing code coverage makes it easy to spot bugs and increases code quality

# 7. Implementation

MIT App Inventor has numerous features. The AppEngine is based on GWT. The project aims to set up a unit testing architecture that can tell us the coverage of the client code in AppEngine. Code coverage assures the tested lines of code. It tells us which branch and block of code is tested and which needs to be tested. To set up the code coverage in AppEngine we can use Jacoco.
Jacoco is a visual tool that tells us the code coverage through a visual report. It shows us the branched lines tested thoroughly, partially, or incompletely. It tells us the code coverage percentage also.

**Setting up Jacoco in AppEngine**
We will add the Jacoco jar file in the appinventor in lib folder and modify the files mentioned below to add jacoco coverage, we will also add the jacoco report to it. And thus this will set up the jacoco for our AppEngine and we can run the tests. Our test report will be created and shall be accessible to us.
build-common.xml file has the testing task for the AppInventor so we can modify it to access jacoco code coverage reports also.

Changes in build-common.xml

```
<project name="CommonDefinitions" xmlns:if="ant:if" xmlns:unless="ant:unless"
    xmlns:jacoco="antlib:org.jacoco.ant">
```

```
<taskdef uri="antlib:org.jacoco.ant" resource="org/jacoco/ant/antlib.xml">
  <classpath path="${appinventor.dir}/lib/jacoco/jacocoant.jar"/>
</taskdef>
```

Wrap the junit of AppEngine in jacoco coverage

```
<jacoco:coverage destfile="${local.build.dir}/jacoco.exec" inclnolocationclasses="true">
</jacoco:coverage>
```

Resulting coverage information is collected during the execution of tests and written to a file when the process terminates.

Write the jacoco report

```
<jacoco:report>
       <executiondata>
         <file file="${build.dir}/jacoco.exec"/>
       </executiondata>
       <structure name="coverage report">
         <classfiles>
              <fileset dir="${class.dir}"/>
         </classfiles>
         <sourcefiles encoding="UTF-8">
              <fileset dir="${src.dir}"/>
         </sourcefiles>
       </structure>

         <html destdir="${coverage.dir}"/>
     </jacoco:report>
```

We will use the jacoco coverage data to convert it into a coverage report. For this we will execute the data in jacoco.exec which will be generated in jacoco coverage this is the first necessary element.

The next necessary element is the structure it will have two parts classifier which will have the directory which will have Java class files.
And the sourcefiles will have the directory of the source corresponding source files.
And finally, we will store the report at location reports/coverage and can access that report.

```
<property name="coverage.dir"
location="${basedir}/reports/coverage"/>
```

After setting up jacoco we can move forward to writing unit tests for the files in AppEngine. We will be writing the tests in JUnit.

Files which we aim to write unit tests for

1. appengine/src/com/google/appinventor/client/boxes
2. appengine/src/com/google/appinventor/client/editor/simple/components

**Files in appengine/src/com/google/appinventor/client/boxes**

```
ubuntu@ubuntu-Lenovo-IdeaPad-S540-15IWL-D:~/mitapp/appinventor-sources/appinventor$ cloc appengine/src/com/google/appinventor/client/boxes
      12 text files.
      12 unique files.
       0 files ignored.

github.com/AlDanial/cloc v 1.82  T=0.01 s (1123.3 files/s, 73387.2 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
Java                            12            112            272            400
-------------------------------------------------------------------------------
SUM:                            12            112            272            400
-------------------------------------------------------------------------------
```

*1.AdminUserListBox.java*

*2.BlockSelectorBox.java*

*3.MotdBox.java*

*4.ProjectListBox.java*

*5.SourceStructureBox.java*

*6.AssetListBox.java*

*7.MessagesOutputBox.java*

*8.OdeLogBox.java*

*9.PaletteBox.java*

*10.PropertiesBox.java*

*11.ViewerBox.java*

**Files in**

**appengine/src/com/google/appinventor/client/editor/simple/components**

```
ubuntu@ubuntu-Lenovo-IdeaPad-S540-15IWL-D:~/mitapp/appinventor-sources/appinventor$ cloc appengine/src/com/google/appinventor/client/editor/simple/components
      96 text files.
      96 unique files.
       0 files ignored.

github.com/AlDanial/cloc v 1.82  T=0.10 s (962.6 files/s, 168668.9 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
Java                            96           2118           5477           9227
-------------------------------------------------------------------------------
SUM:                            96           2118           5477           9227
-------------------------------------------------------------------------------
```

*1.DataFileChangeListener.java*

*2.MockChartDataModel.java*

*3.MockFirebaseDB.java*

*4.MockLineChartBaseDataModel.java*

*5.MockPieChartDataModel.java*

*6.MockSwitch.java*

*7.DesignPreviewChangeListener.java*

*8.MockChart.java*

*9.MockFormHelper.java*

*10. MockLineChartDataModel.java*

*11.MockPieChartView.java*

*12. MockTableArrangement.java*

*13. FormChangeListener.java*

*14.MockChartLayout.java*

*15.MockForm.java*

*16.MockLineChartViewBase.java*

*17.MockPointChartDataModel.java*

*18.MockTableLayout.java*

*19. LayoutInfo.java*

*20.MockChartView.java*

*21.MockFormLayout.java*

*22.  MockLineChartView.java*

*23.MockPointChartView.java*

*24. MockTextBoxBase.java*

*25.MockAreaChartDataModel.java*

*26. MockCheckBox.java*

*27.MockFusionTablesControl.java*

*28.MockLineString.java*

*29.MockPolygonBase.java*

*30.MockTextBox.java*

*31.MockAreaChartView.java*

*32.MockCircle.java*

*33.MockHorizontalArrangement.java*

*34.MockListPicker.java*

*35.MockPolygon.java*

*36.MockTimePicker.java*

*37.MockAxisChartView.java*

*38.MockCloudDB.java*

*39.MockHVArrangementHelper.java*

*40.MockListView.java*

*41.MockRadioButton.java*

*42.MockToggleBase.java*

*43.MockBall.java*

*44.MockComponent.java*

*45.MockHVArrangement.java*

*46.MockMapFeatureBase.java*

*47.MockRectangle.java*

*48.MockTranslator.java*

*49.MockBarChartDataModel.java*

*50.MockComponentsUtil.java*

*51.MockHVLayoutBase.java*

*52.MockMapFeatureBaseWithFill.java  53.MockScatterChartDataModel.java*

*54.MockVerticalArrangement.java*

*55.MockBarChartView.java*

*56.MockContactPicker.java*

*57.MockHVLayout.java*

*58.MockMapFeature.java*

*59.MockScatterChartView.java*

*60.MockVideoPlayer.java*

*61.MockButtonBase.java*

*62.MockContainer.java*

*63.MockImageBase.java*

*64.MockMap.java*

*65.MockScrollHorizontalArrangement.java  66.MockVisibleComponent.java*

*67.MockButton.java*

*68.MockDataFile.java*

*69.MockImage.java*

*70.MockMapLayout.java*

*71.MockScrollVerticalArrangement.java*

*72.MockWebViewer.java*

*73.MockCanvas.java*

*74.MockDatePicker.java*

*75.MockImagePicker.java*

*76.MockMarker.java*

*77.MockSlider.java*

*78.MockWrapper.java*

*79.MockCanvasLayout.java*

*80.MockEmailPicker.java*

*81.MockImageSprite.java*

*82.MockNonVisibleComponent.java*

*83.MockSpinner.java*

*84.package-info.java*

*85.MockChartData2D.java*

*86.MockFeatureCollection.java*

*87.MockLabel.java*

*88.MockPasswordTextBox.java*

*89.MockSpreadsheet.java*

*90. utils*

*91.MockChartData.java*

*92.MockFeatureCollectionLayout.java*

*93.MockLayout.java*

*94.MockPhoneNumberPicker.java*

*95.MockSprite.java*

All these files are approximately 10k lines of code which we plan to unit test in the given time period. We will create a separate test file of each file in these locations in the test folder to write the JUnit tests.
I will also be documenting the jacoco setup and writing a document titled “How to write Junit test” which can be added to the developer guide.

## 8. Deliverables/Outcome

We will have Jacoco code coverage setup for AppEngine with tests written for the above-mentioned files, with updated documentation.

## 9. Timeline

First phase coding period breakdown

| **Week** | **Task** |
|---|---|
| **29 May - 3 June** | Set up the Jacoco code coverage |
| **4 June - 10 June** | Set up the Jacoco code coverage |
| **11 June - 17 June** | 1-8 files from the box folder |
| **18 June - 24 June** | 8-12 files from the box folder and |

| | |
|---|---|
| | Files in appengine/src/com/google/appinventor/client/editor/simple/components/utils |
| **25 June - 1 July** | 1-8 in editor/simple/component |
| **2 July - 8 July** | Buffer Time to complete work if left |
| **10 July** | Evaluation |

Final Phase Coding Period breakdown

| **Week** | **Task** |
|---|---|
| **11 July - 15 July** | 9-20 in editor/simple/component |
| **16 July - 22 July** | 21-35 in editor/simple/component |
| **23 July - 29 July** | 36-50 in editor/simple/component |
| **30 July - 5 Aug** | 51-65 in editor/simple/component |
| **6 Aug - 12 Aug** | 66-75 in editor/simple/component |
| **13 Aug - 19 Aug** | 76-95 in editor/simple/component |
| **20 Aug - 28 Aug** | Final Evaluation |

I will have my summer vacation starting on May 4 so I will be accessible during the summer. And will commit 30-40 hours per week to complete the tasks.

## 10. Future Work

I will write tests for files that are left and not included here. I also wish to contribute to MIT AppInventor in its other ongoing projects/tasks.