Google Summer of Code

# Project Proposal 2022

**MIT App Inventor**

# Make Blockly Capable of Dragging a set of blocks

Hollow Man

**Last Modified Date:** Friday, 15 April 2021

# Contents

# 1. My Interest

## A. Interest in App Inventor

I heard about MIT App Inventor when I was learning Android Application Development using Kotlin. The App Inventor does make it pretty much easier with Google Blockly and VB-like UI designing pages for starters. I've also applied to work for App Inventor in GSoC 2021 but got selected by openSUSE.

## B. Interest in Introductory Programming

I used to take Blockly Introduction Course in my University, and I have also used a software based on BlocklyDuino called Freduino (https://hollowman.ml/FreDuino/blockly/apps/blocklyduino_new/index.html) which uses Blockly to generate Arduino code to build robot cars. I think software like these can attract children a lot and help develop their computational thinking.

# 2. My Experience

## A. Experience with the development tools

## Javascript

I have extensive experience with Javascript including ES5, ES6, Node.JS. I have also done a lot of projects using Javascript, here's a list.

- **ChaoXing ErYa & Wisdom Tree Lessons Helper**

Developed by myself alone, it's a chrome plugin for automating course playing processes.

**Link:** https://github.com/HollowMan6/ChaoXing-ErYa-Wisdom-Tree-Lessons-Helper

- **My Home Page https://hollowman.ml**

The index and funding pages are mainly coded by me along without using framework, and of course with a lot of hidden functionalities. (see https://github.com/HollowMan6/HollowMan6.github.io#function-surprise for more)

I have also created several JS repositories for the background canvas source code used in my Home Page.

**Links:** https://github.com/HollowMan6/HollowMan6.github.io

https://github.com/HollowMan6/Staggered-Mouse-Balls

https://github.com/HollowMan6/canvas-explosion-click.js

https://github.com/hollowman6/canvas-ribbon.js

https://github.com/hollowman6/canvas-ball.js

https://github.com/hollowman6/canvas-nest.js

https://github.com/hollowman6/sun.js

- **Emacs Application Framework**

I participated in this project as a student of Open Source Promotion Plan – Summer 2020 (an activity pretty much like GSoC held by Intelligent Software Research Centre, Institute of Software, Chinese Academy of Sciences in China). During the project I completed 15 compulsories + 3 voluntarily different level tasks using Emacs Lisp, Python, JavaScript, SQLite to add new features for Emacs Application Framework (EAF). One of my greatest contribution was that I added carrot browsing function using JavaScript for the browser in EAF that got the Owner and Project Founder's high appraisal (In Chinese): https://manateelazycat.github.io/emacs/eaf/2020/07/22/eaf-support-caret-browse.html

Here's the detailed tasks that I have done (In Chinese): https://github.com/HollowMan6/My-OSPP-Summer/blob/master/2020-Final-Report.md

**Link to the project**: https://github.com/emacs-eaf/emacs-application-framework

## Java

I also have extensive experience with Java not only because we learn it in class, but also because I have used it in real big projects.

- **Coursework in class**

**Link**: https://github.com/HollowMan6/Answers-for-My-LZU-UG-Courses/tree/master/Java

- **Arthas**

I participated in this project as a student of Alibaba Summer of Code 2020 (an internationalized activity pretty much like GSoC but held by Alibaba). Arthas is an Alibaba Java diagnostic tool, which is the most popular project in star number of Alibaba Group. During the project I wrote Chinese and English online tutorial for Arthas using Katacoda and Vue.js. I also used Java to add command parameters of searching Class Loader by name instead of Hash Code.

Here's the full task list that I have done: https://github.com/HollowMan6/My-Alibaba-Summer-of-Code/blob/master/2020-Arthas/Alibaba%20Summer%20of%20Code%202020%20-%20Arthas%20Final%20Report.md

**Link to the project**: https://github.com/alibaba/arthas

## Git/GitHub

I have made lots of contributions with many PRs (247) and issues (18), as well as many personal projects on GitHub: https://github.com/HollowMan6

## Blockly

I have extensive experience with Blockly as I've taken two courses at my university related to Blockly.

The first one is the robot car course I mentioned above in the “Interest in Introductory Programming” part taken in 2018 Fall (in Chinese): https://github.com/HollowMan6/Answers-for-My-LZU-UG-Courses/tree/master/Making%20a%20Robot%20Car

in that course I learnt how to use blockly and coded several programmes using Blockly(exported in xml format):

Judge whether the multiplication of two numbers is greater than the addition: https://github.com/HollowMan6/Answers-for-My-LZU-UG-Courses/blob/master/Making%20a%20Robot%20Car/%E5%88%A4%E6%96%AD%E4%B8%A4%E4%B8%AA%E6%95%B0%E7%9B%B8%E4%B9%98%E6%98%AF%E5%90%A6%E5%A4%A7%E4%BA%8E%E7%9B%B8%E5%8A%A0.xml

Solving quadratic equations of one variable:

https://github.com/HollowMan6/Answers-for-My-LZU-UG-Courses/blob/master/Making%20a%20Robot%20Car/%E6%B1%82%E8%A7%A3%E4%B8%80%E5%85%83%E4%BA%8C%E6%AC%A1%E6%96%B9%E7%A8%8B%E7%BB%84%E8%A7%A3.xml

Calculate triangle area:

https://github.com/HollowMan6/Answers-for-My-LZU-UG-Courses/blob/master/Making%20a%20Robot%20Car/%E8%AE%A1%E7%AE%97%E4%B8%89%E8%A7%92%E5%BD%A2%E9%9D%A2%E7%A7%AF.xml

Then later in last year, I took the Blockly Creative Programming Course which taught us about the development of Blockly (Some coursework answers In Chinese):

https://github.com/HollowMan6/Answers-for-My-LZU-UG-Courses/tree/master/Blockly%E5%88%9B%E6%84%8F%E8%B6%A3%E5%91%B3%E7%BC%96%E7%A8%8B%EF%BC%88%E7%BD%91%E7%BB%9C%E5%85%B1%E4%BA%AB%E8%AF%BE%EF%BC%89

## B. Experience with teams, online developer communities and large code bases

I have sent a PR #2434 and got merged for MIT App Inventor https://github.com/mit-cml/appinventor-sources/pull/2434 which fixes multiple locales showing in login failed page and solves help-wanted issue #1518.

I’ve also sent several typo-fixing PRs for App Inventor:

https://github.com/mit-cml/appinventor-sources/pull/2450

https://github.com/mit-cml/appinventor-sources/pull/2449

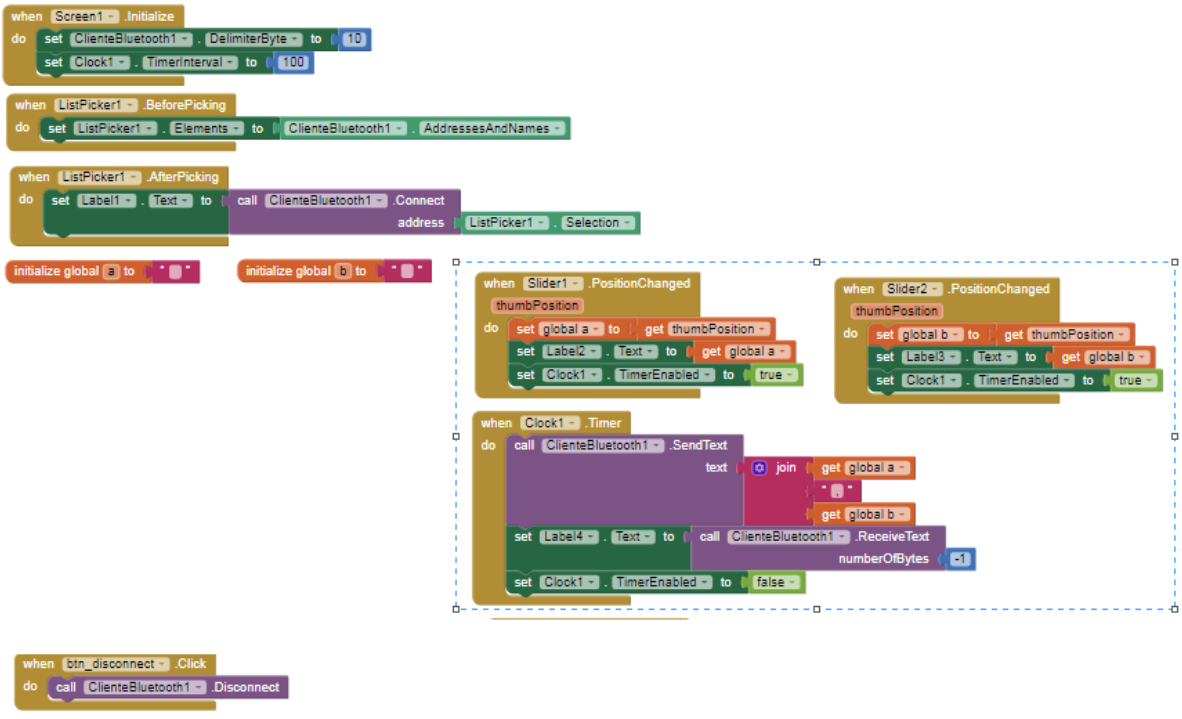
I have also took global open-source internships four times in total, separately funded by Alibaba Summer of Code 2020 https://github.com/HollowMan6/My-Alibaba-Summer-of-Code#my-alibaba-summer-of-code , Google Summer of Code 2021 https://github.com/HollowMan6/My-Google-Summer-of-Code#my-google-summer-of-code , and the Institute of Software, Chinese Academy of Sciences (twice) for Open Source

Promotion Plan - Summer 2020 & 2021 https://github.com/HollowMan6/My-OSPP-Summer#my-open-source-promotion-plan---summer . Some details can also be found on my CV. The most memorable one to me is the Google Summer of Code. The project I did there was to improve the UI usability of IBus. IBus is a framework popular in Asia that integrates with Linux to enable users who speak non-Latin languages to input. Google Summer of Code is a competitive program where tons of students with various project ideas are actively applying. To stand out among the applicants for application, I must be very promising to show that I could finish the project excellently. Although I had no earlier experience developing the GNOME Shell extension, I still want to do the project to benefit Linux users in Asia. Then during the application month, I devoted most of my free time to studying the IBus and GNOME Shell codebase, referring to their documentation, and successfully made a demo. It was a challenging time indeed, but it is worthwhile. Google eventually decided to sponsor my project during the program.

# 3. Proposed summer project

## A. Aim

The aim of this project is to add features allowing a user to drag-select multiple blocks and move them as a group as shown in the example image. Since this functionality will greatly improve users' coding experience, I think it's a priority. I will also work closely with my mentors to define and scope the projects I will tackle during the programme.




## B. Coding

Currently drag on canvas will result in scrolling. I have contacted the mentor and discussed with him last year, we reached a consensus then that it would be best to remain that functionality. So, the drag-select will be activated while the Shift key is pressed.

## 1. One possible and simple solution

I have checked the documentation and tested some functionalities of the following JS library, it's licensed under MIT which is compatible with MIT App Inventor project license:

https://github.com/ThibaultJanBeyer/DragSelect

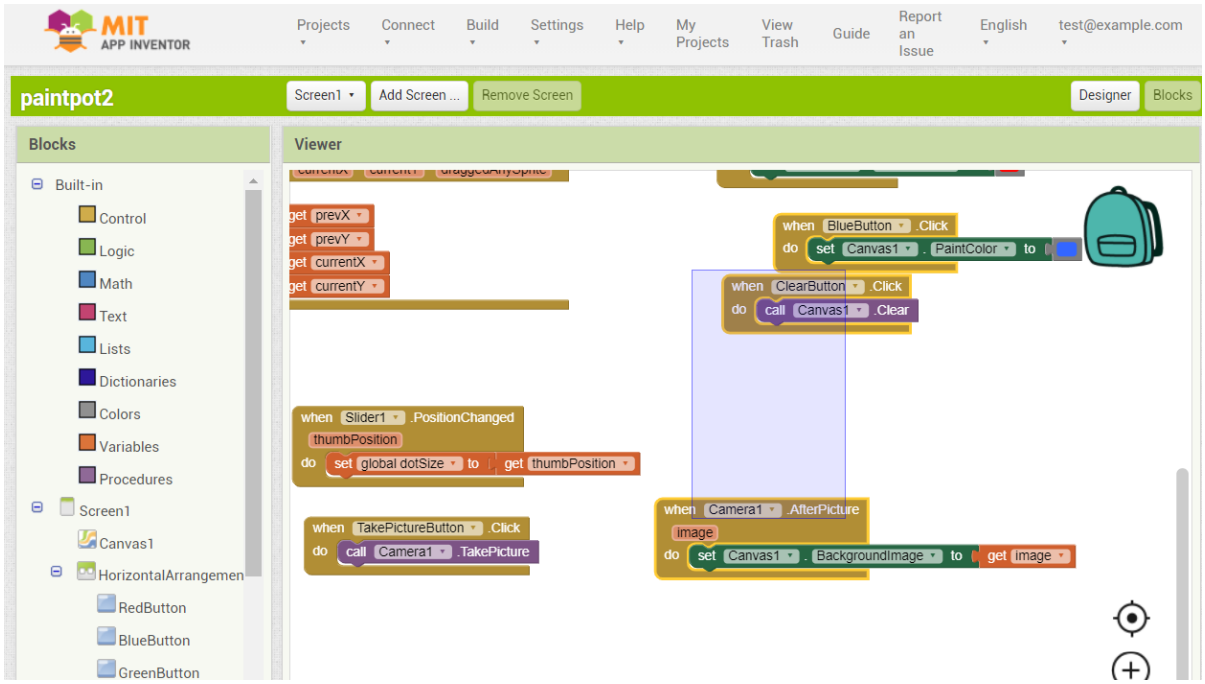
The library can be imported at appinventor/appengine/war/index.html:

```
47    <script src="static/leaflet/Leaflet.Editable.js"></script>
48    <script src="static/leaflet/leaflet.geometryutil.js"></script>
49    <script src="static/leaflet/leaflet.snap.js"></script>
50    <script src="https://dragselect.com/v2/ds.min.js"></script>
51    <script>
52      if (window.navigator.userAgent.indexOf("MSIE") != -1){
53        document.getElementById("unsupported").style.display = 'block';
54      }
55    </script>
56  </body>
57 </html>
```

After adding an Event Listener for Shift key, when Shift key is pressed, the following JS code should be executed to get DragSelect initialized:

```
> s = new DragSelect({
    selectedClass: "blocklySelected",
    selectables:
  document.querySelectorAll("g.blocklyBlockCanvas >
  g.blocklyDraggable"),
    area: document.querySelector('.blocklyWorkspace'),
    multiSelectMode: true,
    draggability: false
  });
<-  t {start: ƒ, getSelection: ƒ, getSelectables: ƒ, getIn
  ▶ itialCursorPosition: ƒ, getCurrentCursorPosition:
    ƒ, …}
```

Then the user begins to make a selection. Users can select multiple nodes by either drag or click on the blocks.




When the user releases the Shift key, executing s.getSelection() to get selected blocks. Then execute s.stop():


The work we need to do in the project period is to modify the blockly library https://github.com/mit-cml/blockly/tree/mitmaster so that we can drag multiple blocks in the block list at the same time since currently only one block is allowed to be dragged.

## 2. Another possible and best solution

I've also submitted an issue to Google Blockly last year https://github.com/google/blockly/issues/4739 and got a response from the Blockly team telling me about their progress in the functionality https://github.com/google/blockly-samples/issues/267.

So another solution can be a Blockly workspace plugin, so its name would be @blockly/workspace-multi-select

The details can be (ideas taken from https://github.com/google/blockly-samples/issues/267#issuecomment-464184902 ):

1. If all of the selected blocks are in one block stack (e.g. there is only one top block in the selection), treat the connections behavior the same way as when dragging the top block.
2. If all of the selected blocks are not in one block stack (e.g. there are multiple top blocks in the selection), then unplug the group from wherever it is and connect previous and next blocks to close the generated space.

3. The "Duplicate" menu will duplicate all selected blocks. The selection will be changed to all newly created duplicate blocks.
4. The "Add Comment" option will not appear or will be disabled if multiple blocks are selected.
5. "Collapse Block" will instead be read as "Collapse [x] Blocks" and will collapse all selected blocks. Children of selected blocks which are themselves not selected will not be in the count and will not be collapsed.
6. "Disable Block" will instead read as "Disable [x] Blocks" and will disable all selected blocks. Children of selected blocks which are themselves not selected will not be in the count and will not be set as disabled.
7. As with deleting a block with children, "Delete [x] Blocks" will appear in the menu and will represent the count of all selected blocks *and* their children. Deletion will delete all blocks and their children. (This logic may require some finagling to prevent attempting to delete blocks twice if a child block is one of the selected blocks.)
8. The "Help" option would not appear or would be disabled when multiple blocks are selected.
9. Copy would copy all selected blocks, and paste would paste all blocks on the clipboard.
10. As with the current behavior, all blocks could be deselected by clicking on the workspace background. Additional blocks would be able to be selected by holding shift while clicking the new block. Clicking a new block without pressing shift would change the selection to that block. Holding shift and dragging an area to select would deselect any previously selected blocks and select all blocks touched by the new selection rectangle. Holding shift *and* ctrl and dragging an area to select would *add* all blocks touched by the new selection rectangle to the currently selected blocks.
11. Touchscreen problems would be addressed by drawing a rectangle and automatically selecting all blocks within it and also adding some type of "selection mode" toggle button which would change selection behavior on touchscreens between single and multiple. This is pretty much like the selecting and dragging files on OS like Windows, Mac, etc. and the use case of the library I mentioned above https://github.com/ThibaultJanBeyer/DragSelect

Then when the project finishes, I will help out what I can with the community and follow the instructions of mentors.

# 4. Timeline and Goals

I can work full time (unlimited time, at least 30 hours per week) because I have no other commitments during the project period.

I know a mentor isn't a replacement for Google or stack overflow. I will only discuss the difficulties that I can't overcome by myself with mentors and send a "summary of the week" every week by email to mentors to inform my progress of the work so far and what I have achieved this week.

Here is the timeline:

## A. Community Bonding period (May 20, 2022 - June 12, 2022)

During the community bonding period, I aim to go through the MIT App Inventor codebase thoroughly. Currently, I already have some knowledge about the Blockly and codebase. However, going through the entire codebase will help me work faster during my coding period and make it possible to contribute to other features of the project in the future.

I would also like to interact with the mentors as well as get to know the community as a whole and help out whenever I can.

I expect to try my best right after the start of the community bonding period instead of waiting for the coding period to start.

### B. First Part of Coding Time (June 13, 2022 – July 25, 2022)

● Finished adding feature allowing a user to drag-select multiple blocks and move them as a group.

● Help out in the community.

● Other tasks instructed by the mentors.

### C. First Evaluation Phase (July 25 - 29, 2022)

**Deliverables**

● Finished adding feature allowing a user to drag-select multiple blocks and move them as a group.

### D. Final Part of Coding Time (July 25, 2022 - September 4, 2022)

● Time for delays in adding features allowing a user to drag-select multiple blocks and move them as a group.

● Help out in the community.

● Other tasks instructed by the mentors.

### E. Final Evaluation Phase (September 5 - 12, 2022)

**Deliverables**

● Finished adding feature allowing a user to drag-select multiple blocks and move them as a group.

● Finished other tasks instructed by the mentors.

### F. Future (September 20, 2022 -)

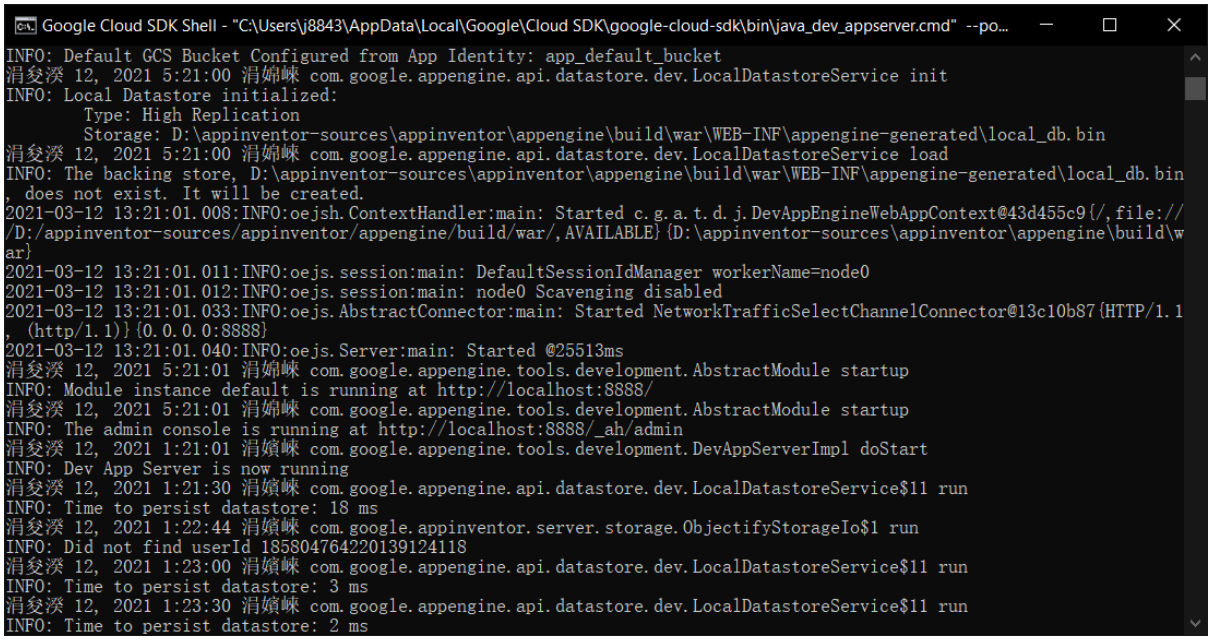
I will continue contributing and do my best to maintain the MIT App Inventor project after I successfully complete the GSoC project and if possible, I am happy to be a mentor of MIT App Inventor project in the future GSoC.

## 5. Application Challenges

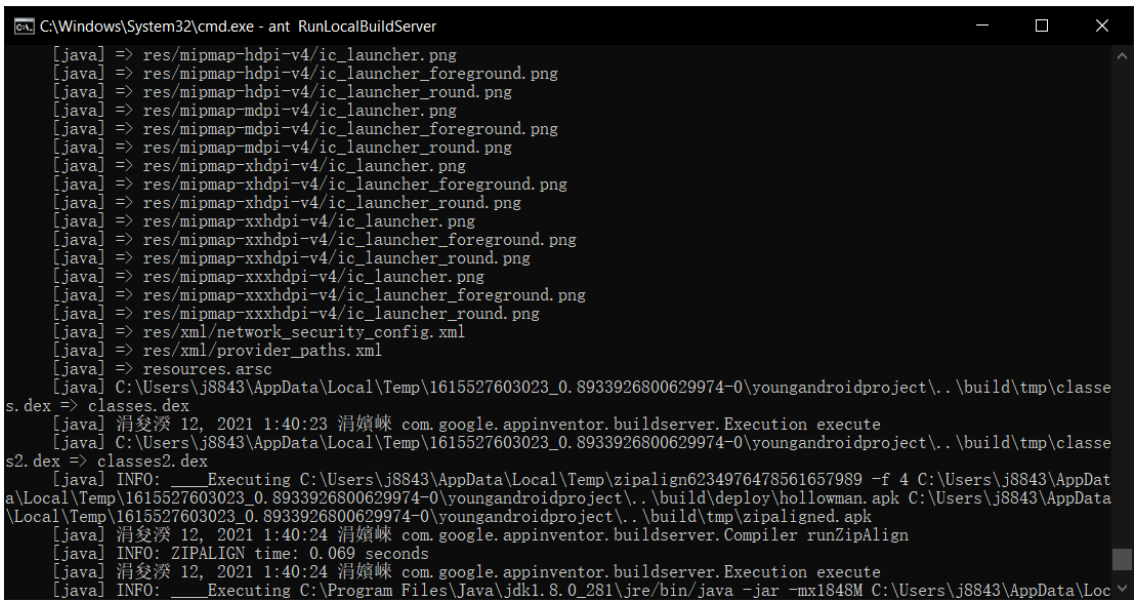
### A. Challenge 1: Creating a non-trivial app

I chose to manually set up the running environment on Windows. Here’s my result

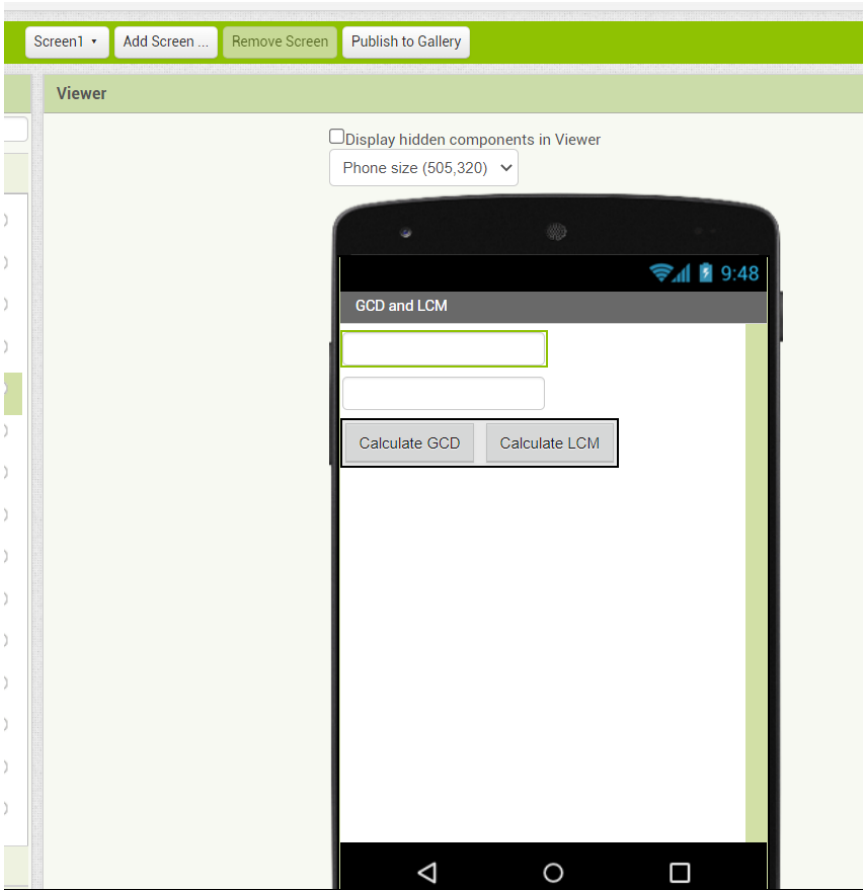
Main server:



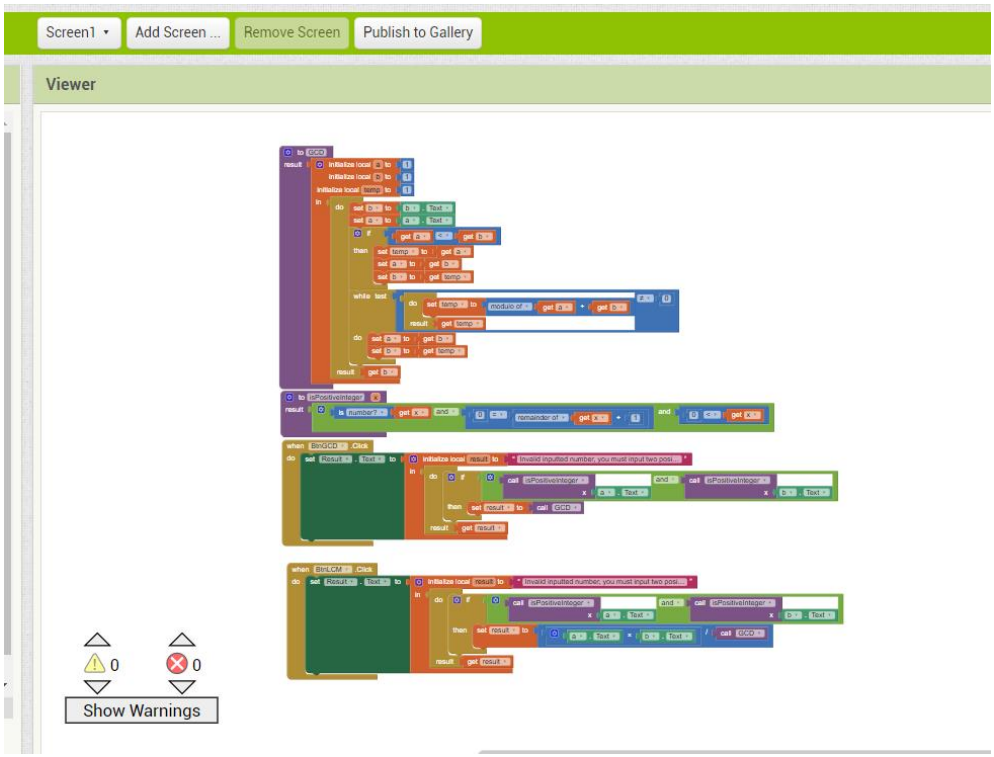

Build server:




User interface:


Screen1
Add Screen ...
Remove Screen
Publish to Gallery
Viewer
Display hidden components in Viewer
Phone size (505,320)
9:48
GCD and LCM
Calculate GCD
Calculate LCM

Logic:




I created a simple app called GCDandLCM that can calculate the GCD and LCM with the two inputted positive integer.

Here's the link for APK and the App Inventor Source file (AIA): https://drive.google.com/drive/folders/1QbryeslYYCEtz3XQX9Sd9Jybp9Kzykip?usp=sharing

## B. Challenge 2: Design challenge -- Enhancing the Camera operation

**useFront:** Checking the source code https://github.com/mit-cml/appinventor-extensions/blob/master/appinventor/components/src/com/google/appinventor/components/runtime/Camera.java#L173 and commit history, useFront is realized by adding intent extra CAMERA_FACING. Then tracing back to issue #180 https://github.com/mit-cml/appinventor-sources/issues/180 of MIT App Inventor in Dec 16, 2014, the problem seems to occur earlier than that, and @marksherman said it appears that the feature had been removed from nearly all androids, including older ones. He guesses that a google play services update was pushed out that removed it, even to old 2.3 devices, since the function is used for test and not officially documented, check the Android source files, there do exist that "test" method that are in the Util class file but not officially documented:

```
private static final String EXTRAS_CAMERA_FACING =
        "android.intent.extras.CAMERA_FACING";

// This is for test only. Allow the camera to launch the specific camera.
public static int getCameraFacingIntentExtras(Activity currentActivity) {
    int cameraId = -1;

    int intentCameraId =
            currentActivity.getIntent().getIntExtra(Util.EXTRAS_CAMERA_FACING, -1);

    if (isFrontCameraIntent(intentCameraId)) {
        // Check if the front camera exist
        int frontCameraId = CameraHolder.instance().getFrontCameraId();
        if (frontCameraId != -1) {
            cameraId = frontCameraId;
        }
    } else if (isBackCameraIntent(intentCameraId)) {
        // Check if the back camera exist
        int backCameraId = CameraHolder.instance().getBackCameraId();
        if (backCameraId != -1) {
            cameraId = backCameraId;
        }
    }
    return cameraId;
}
```

Since the function never occurs in documentations, I think it's because there were some bugs and vulnerabilities that finally resulted in depreciation.

Then I searched in Google and Stackoverflow, and found this answer on https://stackoverflow.com/a/41056291/14343335, so a possible solution can be using the following code by adding intent extra LENS_FACING_FRONT and USE_FRONT_CAMERA so that nearly all the phones may use front camera by default. A PR has been sent in project https://github.com/miguelbcr/RxPaparazzo/pull/97#discussion_r227495088 but still open after so many years. Since I don't have many Android devices, I can't make sure this will work, otherwise I will send a PR to fix this issue:

intent.putExtra("android.intent.extras.CAMERA_FACING", android.hardware.Camera.CameraInfo.CAMERA_FACING_FRONT);

intent.putExtra("android.intent.extras.LENS_FACING_FRONT", 1);

intent.putExtra("android.intent.extra.USE_FRONT_CAMERA", true);

A further investigation in Android API 21 https://developer.android.com/reference/android/hardware/camera2/CameraMetadata#LENS_FACING_FRONT proves that intent extra LENS_FACING_FRONT may work in some sense.

**Let the camera take pictures automatically:** Referring to Google, it seems like that due to security reasons, most Android devices are not allowed to do so to avoid spying. Most of the methods on the Internet won't work for all devices.

But for Nexus devices, add intent extra quickCapture can do this referring to the project https://github.com/react-native-image-picker/react-native-image-picker/issues/647#issuecomment-319668463. To do so, just as useFront but replace https://github.com/mit-cml/appinventor-extensions/blob/master/appinventor/components/src/com/google/appinventor/components/runtime/Camera.java#L173 android.intent.extras.CAMERA_FACING with android.intent.extras.quickCapture