GSOC Proposal: MIT APP INVENTOR, 2024

Block Editor Projects: Improve the Blockly workspace multi-select plugin

Lakshya Shishir Khandelwal

Indian Institute of Technology, Roorkee

Uttarakhand, India

Table of Contents

About Me

Personal Info

- Name: Lakshya Shishir Khandelwal
- Email: lakshyashishir1@gmail.com
- GitHub: https://github.com/lakshyashishir
- Mentors: Evan Patton, Songlin Jiang
- Location: India
- Time zone: IST (UTC +05:30)

Educational Info

- University: Indian Institute of Technology, Roorkee
- Major: Geophysical Technology
- Current Year: Sophomore (2nd) (Expected Graduation: 2027)
- Degree: Integrated Master of Technology (Integrated MTech)

Background

Engaging in hackathons and course projects, I've cultivated my skills in software development. My journey began with MIT App Inventor in 8th grade, and I've since built numerous projects with it, even becoming a finalist in Google Code to Learn. My interests span web development, AI, and machine learning. Currently, I'm a developer at SDSLabs (Student Technical Group), focusing on React, JavaScript, and Python for machine learning projects.

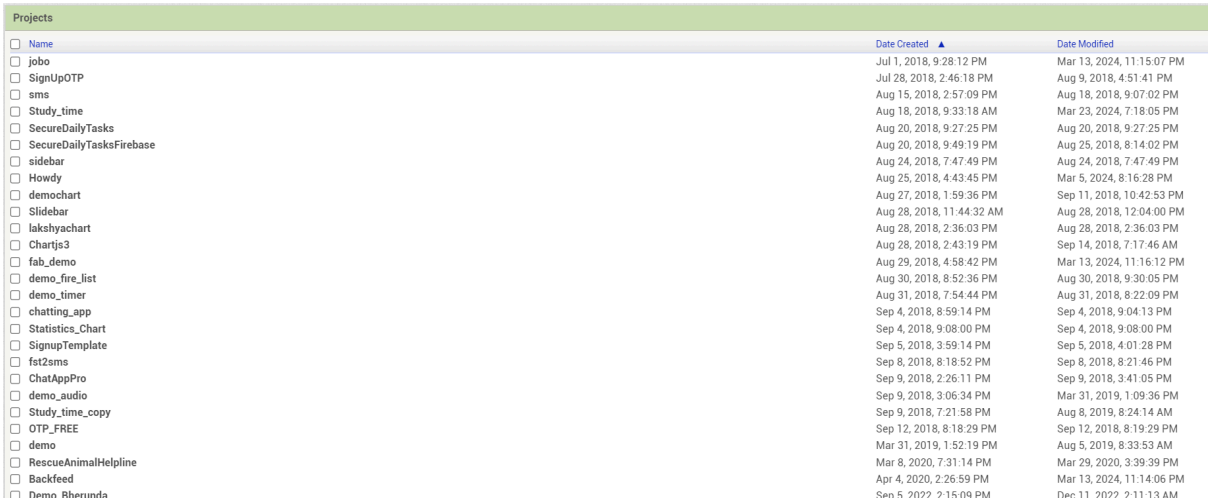
Programming Languages

React.js, JavaScript, Node.js, Python, Go

Why MIT App Inventor?

My deep bond with MIT App Inventor began at the start of my programming journey, igniting my curiosity and enabling me to bring ideas to life through app development. Throughout high school, it became pivotal in my projects, fostering creativity and fueling my passion for innovation.

As I join Google Summer of Code, I'm eager to give back to the community by reconnecting with MIT App Inventor and contributing to its success. I aim to be an active member, empowering others to explore app development.



Projects

Name	Date Created ▲	Date Modified
jobo	Jul 1, 2018, 9:28:12 PM	Mar 13, 2024, 11:15:07 PM
SignUpOTP	Jul 28, 2018, 2:46:18 PM	Aug 9, 2018, 4:51:41 PM
sms	Aug 15, 2018, 2:57:09 PM	Aug 18, 2018, 9:07:02 PM
Study_time	Aug 18, 2018, 9:33:18 AM	Mar 23, 2024, 7:18:05 PM
SecureDailyTasks	Aug 20, 2018, 9:27:25 PM	Aug 20, 2018, 9:27:25 PM
SecureDailyTasksFirebase	Aug 20, 2018, 9:49:19 PM	Aug 25, 2018, 8:14:02 PM
sidebar	Aug 24, 2018, 7:47:49 PM	Aug 24, 2018, 7:47:49 PM
Howdy	Aug 25, 2018, 4:43:45 PM	Mar 5, 2024, 8:16:28 PM
demochart	Aug 27, 2018, 1:59:36 PM	Sep 11, 2018, 10:42:53 PM
Slidebar	Aug 28, 2018, 11:44:32 AM	Aug 28, 2018, 12:04:00 PM
lakshyachart	Aug 28, 2018, 2:36:03 PM	Aug 28, 2018, 2:36:03 PM
Chartjs3	Aug 28, 2018, 2:43:19 PM	Sep 14, 2018, 7:17:46 AM
fab_demo	Aug 29, 2018, 4:58:42 PM	Mar 13, 2024, 11:16:12 PM
demo_fire_list	Aug 30, 2018, 8:52:36 PM	Aug 30, 2018, 9:30:05 PM
demo_timer	Aug 31, 2018, 7:54:44 PM	Aug 31, 2018, 8:22:09 PM
chatting_app	Sep 4, 2018, 8:59:14 PM	Sep 4, 2018, 9:04:13 PM
Statistics_Chart	Sep 4, 2018, 9:08:00 PM	Sep 4, 2018, 9:08:00 PM
SignupTemplate	Sep 5, 2018, 3:59:14 PM	Sep 5, 2018, 4:01:28 PM
fst2sms	Sep 8, 2018, 8:18:52 PM	Sep 8, 2018, 8:21:46 PM
ChatAppPro	Sep 9, 2018, 2:26:11 PM	Sep 9, 2018, 3:41:05 PM
demo_audio	Sep 9, 2018, 3:06:34 PM	Mar 31, 2019, 1:09:36 PM
Study_time_copy	Sep 9, 2018, 7:21:58 PM	Aug 8, 2019, 8:24:14 AM
OTP_FREE	Sep 12, 2018, 8:18:29 PM	Sep 12, 2018, 8:19:29 PM
demo	Mar 31, 2019, 1:52:19 PM	Aug 5, 2019, 8:33:53 AM
RescueAnimalHelpline	Mar 8, 2020, 7:31:14 PM	Mar 29, 2020, 3:39:39 PM
Backfeed	Apr 4, 2020, 2:26:59 PM	Mar 13, 2024, 11:14:06 PM
Demo_Bherunda	Sep 5, 2022, 2:15:09 PM	Dec 11, 2022, 2:11:13 AM

(Some projects I built on MIT App Inventor in my school time)

Why me?

I possess extensive experience in contributing to open-source projects and have a solid background in front-end development. My proficiency spans React, JavaScript, Node.js, and AI&ML with years of coding experience in these languages. Moreover, I have experience working in SDSLabs where we build production-level applications from scratch and thus I know the technical difficulties that might arise and how to deal with them.

Achievements

Google Code to Learn 2019, 2020, 2021: Finalist and Winner (x2)

Syntax Error 2023: Finalist

Winter of Code 2023: Accepted

Past Projects /Experience

- Quizio - I contributed to developing the front end for quiz-taking and checking processes, as well as debugging the backend. This project provided hands-on experience with a large-scale CRUD application built using **JavaScript, ReactJS, and MongoDB**
  Frontend : https://github.com/sdslabs/Helios
  Backend : https://github.com/sdslabs/Athena

- LeSo - An online legal service provider. I single-handedly developed the entire product and assisted in its launch. My collaboration with LeSo spanned from July 2023 to February 2022.
  Tech Stack: JavaScript, ReactJS, MongoDB

- xPay - xPay is a payment orchestration product. I am an early member of the team, interning for a month. **Tech Stack: Javascript, Java**

- Study Time Management - One of my **App Inventor** projects I built in school. Became a finalist in Google Code to Learn with the same project. Continued to build complex apps on the platform even after this.
  AIA :
  https://drive.google.com/file/d/1lkrgvlF6FXrdRElAY1rXY2CGH9s_fkYf/view?usp=sharing

Project Details

Overview

This GSOC proposal aims to enhance the Blockly workspace multi-select plugin for MIT App Inventor by implementing two crucial improvements. Firstly, I intend to integrate the IDragger introduced in Blockly 11; currently, the multiselect plugin uses a custom block dragger for dragging multiple blocks. Once it's implemented it will make the plugin easier to maintain in the future.

I will also focus on resolving the transparent SVG issue associated with DragSelect and Blockly, ensuring a visually cohesive interface and improving user interaction within the workspace.

Issues (175 Hrs)

I am picking up two enhancements for the project Improve the Blockly workspace multi-select plugin. Hence this is a medium 175 Hrs project.

1. **Use IDragger introduced in Blockly 11 for multi-select dragging #39**

2. **Fix the known transparent SVG issue related to DragSelect and Blockly**

Implementation Details and Deliverables

In this section, I will be going over the main implementation steps of this project.

1. Custom Draggable

To enhance the multiselect plugin, we will implement a **custom IDraggable** that encapsulates all selected elements. It can delegate drag operations to the individual elements.

```
class MultiselectDraggable implements IDraggable {
  private subDraggables: IDraggable[];

  // Method to add sub-draggables representing individual selected elements
  addSubDraggable() {
    // Implementation
  }

  // Initiates drag operation for all selected elements
  startDrag(e: PointerEvent) {
    for (const draggable of this.subDraggables) {
      draggable.startDrag(e);
    }
  }

  // Drags all selected elements to new location
  drag(newLoc: Coordinate, target: IDragTarget, e?: PointerEvent) {
    for (const draggable of this.subDraggables) {
      draggable.drag(
        Blockly.Coordinate.sum(newLoc, draggable.dragOffset),
        target,
        e
      );
    }
  }

  // Ends drag operation for all selected elements
  endDrag(e: PointerEvent) {
    for (const draggable of this.subDraggables) {
      draggable.endDrag(e);
    }
  }
}
```

This implementation ensures compatibility with other plugins like the scroll options plugin, as the custom IDraggable provides a seamless integration without requiring additional modifications to the multiselect plugin.

2. Selection

Currently, the multiselect plugin's selection mechanism is limited, randomly selecting one of the blocks and passing it to **Blockly.common.setSelected**. To improve this, we will modify the selection process to ensure that the **MultiselectDraggable** is correctly passed to **setSelected**, facilitating proper handling by the Gesture and subsequent interaction with the IDragger.

```
// Function to handle selection of multiple blocks
function handleMultiselectSelection(blocks: Block[]) {
  // Create instance of MultiselectDraggable
  const multiselectDraggable = new MultiselectDraggable(blocks);

  // Pass MultiselectDraggable to setSelected
  Blockly.common.setSelected(multiselectDraggable);
}
```

In this implementation, when multiple blocks are selected, the handleMultiselectSelection function is called. This function creates an instance of MultiselectDraggable, encapsulating all selected blocks. Then, it passes this instance to setSelected, ensuring that the Gesture can properly interact with the IDragger for seamless multi-selection dragging.

3. Fixing transparent SVG issues related to DragSelect and Blockly

The multiselect plugin currently relies on DragSelect to determine block selection. However, DragSelect listens to SVG path elements, treating them as rectangles with transparent areas. This can cause issues with irregularly shaped blocks. To address this, a patch was introduced (v0.1.4) that includes a filter function. This function, called filterParent, removes parent blocks when their child blocks are selected. It iterates through selected blocks, identifying cases where a parent block fully covers its child's selection area. Such parent blocks are then removed from the selection. This filtering logic is applied within the filterSelected function of dragSelect_, ensuring that only child blocks remain selected. This approach resolves issues with invisible rectangles, ensuring correct selection behavior within the Blockly workspace.

A proper fix to this issue can be adding functionality to the `**BlockSVG**` class in Blockly to determine whether a given point lies inside the SVG representation of a block. This functionality would be facilitated by the addition of an `**isPointInside**` method within the `BlockSVG` class.

When called, the `**isPointInside**` method would take the coordinates of the point in question and transform them into local coordinates relative to the SVG representation of the block. This transformation would involve applying any necessary adjustments, such as accounting for the workspace offset or considering the current zoom level, through the `**transformToSvgCoordinates**` method.

Once the coordinates are transformed, the method would calculate the bounding box of the SVG representation using the `**getBoundingBox**` method. By determining the minimum and maximum coordinates along the x and y axes, the bounding box would define the boundaries of the SVG.

Finally, the `**isPointInside**` method would compare the transformed coordinates against the bounding box. If the coordinates fall within the boundaries of the SVG, the method would return `**true**`, indicating that the point is inside the block. Otherwise, it would return `**false**`. This approach would provide a more accurate means of determining point containment within block SVG compared to relying on external libraries like DragSelect.

Project Timeline

Following is the project timeline for my GsoC. I would be taking up the 175 hour format.

Pre GSoC Period	
April 2nd - May 3rd	● Contribute to solving more issues related to Blocky Editor. Complete any incomplete PR. I will try picking up issues related to Multiselect Plugin (if any).

Community Bonding Period

May 4th - May 28th	● Get to know more about the App Inventor community. ● Discuss methods of implementation with my mentor. ● Identify more relevant parts in the codebase, and study the multi-select plugin code.

Coding Period	
June 9th - June 22nd	● Begin implementing the custom IDraggable, as outlined here in issue #39. ● Develop methods for startDrag, drag, and endDrag to handle dragging operations for multiple selected elements. ● Test the initial implementation of the custom IDraggable class to ensure it properly wraps all selected elements and delegates drag operations.
June 23rd - July 7th	● Address any issues or bugs encountered during testing and debugging. ● Work on passing the MultiselectDraggable to setSelected instead of a single selected block to ensure proper handling by the IDragger.
July 8th - July 12th	● Evaluation 1 Phase

July 13th - July 27th	• Perform comprehensive testing of the plugin and performance improvements. • Understand the transparent SVG issue, and fix that is introduced in v0.1.4. • Discuss with mentors for possible solutions and implementation.
July 28th - Aug 10th	• Work on the transparent SVG issue. • Address any issues or bugs encountered during testing and debugging.
Aug 10th - Aug 28th	• Document the implementation process, including any modifications made to the plugin. • Prepare a final report summarizing the work completed, challenges faced, and outcomes achieved during the implementation of Tasks 1 and 3. • Review the timeline with mentors and seek feedback for further improvements or adjustments if needed.

Post GSoc Goals

I have learned a lot and was introduced to a new ecosystem by contributing to the MIT App Inventor and even after the GSoC period ends, I plan on contributing to this organization, by adding to my past projects and working on open issues.

Development Environment

To build and code MIT App Inventor, I will use Windows with a manual app inventor setup.

Time Commitment

4th May-10th May: College end Semester examinations (10-15 hours/week)

10th May-15th July: Summer Vacation (35 hours/week)

16th July-28th August: College Resumes (30 hours/week)

As for other commitments or vacation plans during the summer, I have none, so my focus will remain solely on this project. I will regularly update all community members on my status and ensure transparency in the project.

I am 100% dedicated to MIT App Inventor and have absolutely no plans to submit proposals to any other organizations.