Summer Fellowship Report

On

Minor improvements and bug fixes, components additions & versioning support in Arduino on cloud

Submitted by

Kartik Gautam

Amity University Uttar Pradesh

Under the guidance of

Prof. Kannan Moudgalya

Chemical Engineering Department

IIT Bombay

Mentors

Mr. Nagesh Karmali

Ms. Firuza Ambara

August 2021

Acknowledgement

I, the summer intern of the FOSSEE - Arduino On Cloud am so overwhelmed in all humbleness and gratefulness to acknowledge my deep gratitude to all those who have helped me put my ideas to perfection and have assigned tasks well above the level of simplicity and into something concrete and unique.
I wholeheartedly thank **Prof. Kannan M. Moudgalya** for having faith in me, selecting me to be a part of his valuable project and for constantly motivating me to do better. I thank **Mr. Nagesh Karmali** and **Ms. Firuza Aibara** for providing me the opportunity to work on this project. I am also very thankful to my mentors for their valuable suggestions. They were and are always there to show me the right track when needed help. With help of their brilliant guidance and encouragement, I was able to complete tasks properly and was up to the mark in all the tasks assigned. During the process, i got a chance to see the stronger side of our technical and nontechnical aspects and also strengthen our concepts. Last but not the least, I sincerely thank all our other colleagues working in different projects under **Prof. Kannan M. Moudgalya** for helping us evolve better with their critical advice.¶

Declaration

I declare that this written submission represents our ideas in my own words and whenever other's ideas or words have been included, I have adequately cited and referenced the original sources. I declare that i have properly and accurately acknowledged all sources used in the production of this thesis.

I also declare that We have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. I understand that any violation of the above will be a cause for disciplinary action by the Institute and can also evoke penal action from the sources which have not been properly cited or from whom proper permission has not been taken when needed.

Kartik Gautam¶

Index

1. Introduction
    1.1. Project Overview

2. Feature Additions
    2.1. Import export
    2.2. Exit confirmation dialog
    2.3. Undo-Redo
    2.4. PWM support for LED
    2.5. Ability to change voltage using Potentiometer
    2.6. Versioning Support

3. Component Additions
    3.1. L293D Motor Driver
    3.2. Light Detecting Resistor
    3.3. Thermistor

4. Examples
    4.1. LED with Potentiometer
    4.2. L293D Motor Driver
    4.3. LED with LDR
    4.4. Serial output with thermistor
    4.5. LED's brightness using Thermistor

5. Bug Fixes
    5.1. Delete/Backspace button bug
    5.2. Save bug
    5.3. Breadboard issue
    5.4. Delete component issue
    5.5. Breadboard delete issue
    5.6. LED issue
    5.7. Breadboard pins issue

6. Conclusion

7. Future Work

8. References

List Of Figures

Figure 2.1: Import and Export buttons in simulator
Figure 2.2: Dialog to rename download file while exporting a circuit
Figure 2.3: Dialog to confirm whether user really wants to exit or not
Figure 2.4: Internal working of undo & redo stack (data switches between both stacks)
Figure 2.5: Placement of undo & redo buttons
Figure 2.6: PWM actual working
Figure 2.7: LED brightness when duty cycle is set at 50
Figure 2.8: LED brightness when duty cycle is set at 200
Figure 2.9: LED's brightness when potentiometer's knob is set at low
Figure 2.10: LED's brightness when potentiometer's knob is set at high
Figure 2.11: Multiple versions of project in left panel
Figure 2.12: Dialog to set name of new branch while creating it

Figure 3.1: Circuit with L293D Motor Driver
Figure 3.2: Circuit with LDR
Figure 3.3: Formula of internal resistance of thermistor
Figure 3.4: Circuit with thermistor

Figure 4.1: LED with potentiometer circuit
Figure 4.2: L293D Motor Driver circuit
Figure 4.3: LED with LDR circuit
Figure 4.4: Serial output with thermistor circuit
Figure 4.5: LED's brightness using Thermistor circuit

Figure 5.1: Circuit on which testing was caried
Figure 5.2: Error in console (Production)
Figure 5.3: Error in console (Dev)
Figure 5.4: Array (same values of y as key)
Figure 5.5: Array (different values of y as key)
Figure 5.6: Basic LED circuit
Figure 5.7: Wire to delete
Figure 5.8: Connection between LED and breadboard gets detached
Figure 5.9: Reason for detachment, same IDs of wires
Figure 5.10: Error in console
Figure 5.11: LED glows even without wire connected
Figure 5.12: LED not glowing
Figure 5.13: Set of points in breadboard

Chapter 1

Introduction

Arduino on cloud is a cloud-based simulator, which can be used by students and researchers to test and run the simulation on the web, before actual execution. This system allows the users to drag and drop Arduino components from the left pane onto the working space on the right. The pins of the Arduino board can be connected to various input/ output devices like LED, motor, push-button, etc using wires. There is also a facility to change the color of wires, LEDs, and such components, to differentiate easily. The users can then proceed to write their code in the code window which is then simulated. There is an option for the users to print or save it in pdf format for documentation purposes. The basic ERC check enables the users to find out errors if any.[1]

1.1. Project Overview

During this fellowship, I worked on the existing project which was already completed to some extent, for the most part, I did bug fixes, new component additions, and circuit additions in the gallery and features added. All of these tasks focused on the overall improvement of the simulator.

Chapter 2

Feature Additions

During the internship, I added a few features to the project, most of these features aimed at improving the accessibility of the website and facilitate some stuff in the website.

2.1 Import Export Circuit Functionality

The first feature addition that I did was "Import and export functionality". This feature aimed to provide the user with a function so that he/she can save the circuit locally on his system memory, if he/she wants to then can even share the circuit with others and upon need, he/she could import the saved circuit on website.

The easiest and most convenient approach to do this was to add two functionalities one export which would allow a user to save the current progress into a JSON file, and an import function that would allow a user to replicate the circuit on his/her website again. The format used for saving the circuit was JSON, the reason being versatility and flexibility of the format.

This feature was implemented without any issue, to carry out the required processes two simple buttons on the header/navbar were added one for exporting that allowed the user to save the JSON file into the user's system and another for importing, which would open a dialog for choosing JSON file to import.
Upon exporting, a JSON file gets saved containing all meta-information about the current circuit, whereas upon importing, a file choose window opens that allows the user to select an existing JSON file to import from. The file content then gets read, which restores all information present in JSON to the current schematic.¶

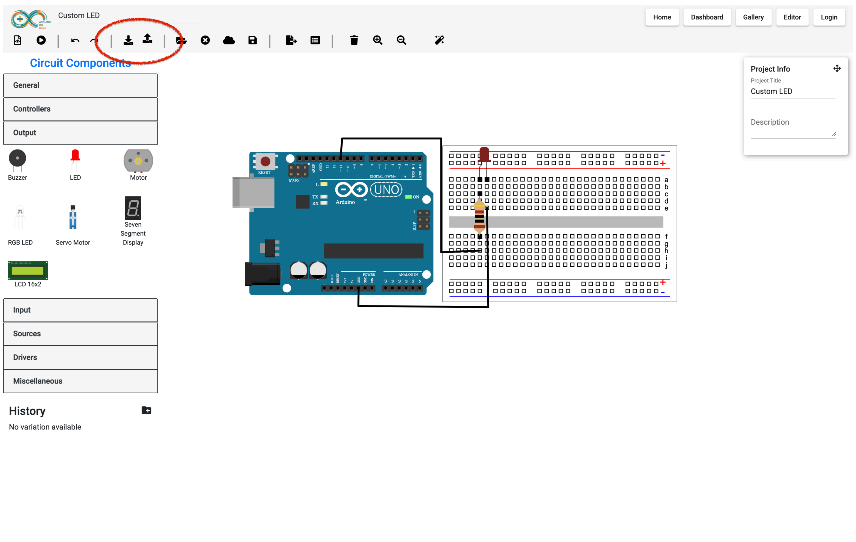
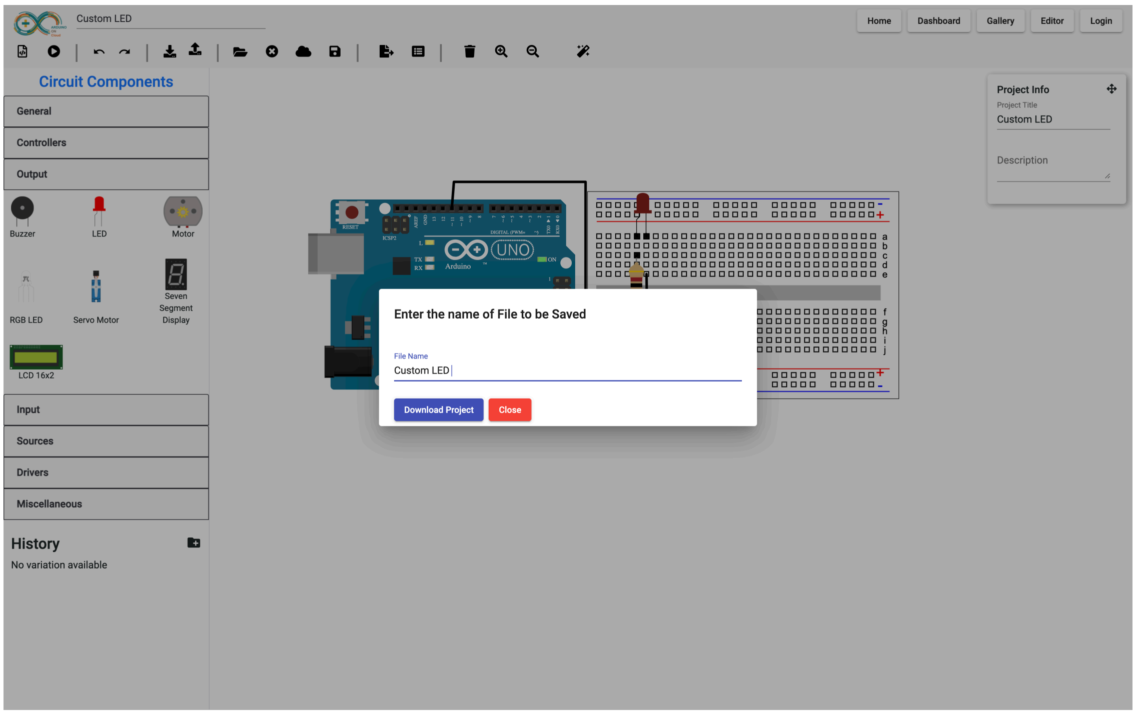


Figure 2.1: Import and Export buttons in simulator



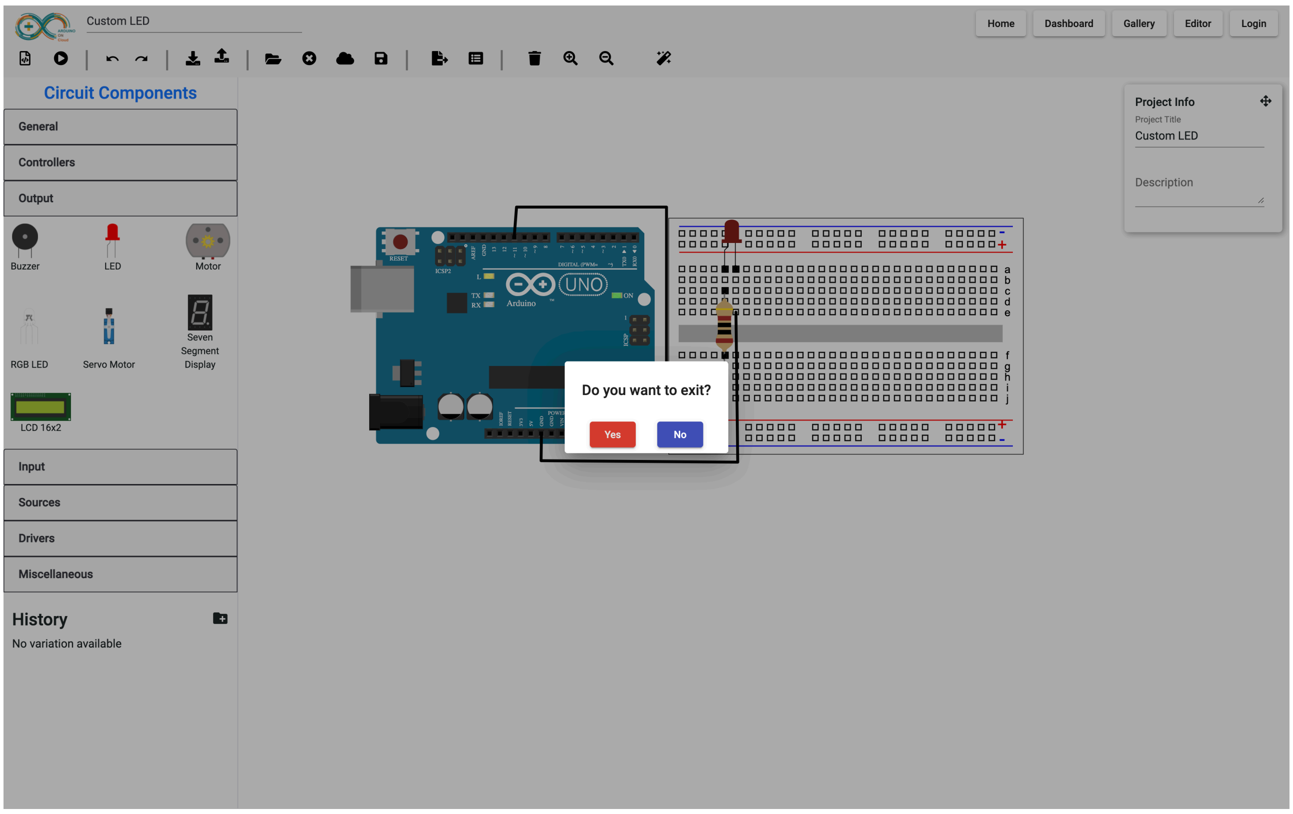
Enter the name of File to be Saved

File Name: Custom LED

Figure 2.2: Dialog to rename download file while exporting a circuit

2.2 Exit Confirmation Dialog

Exit confirmation dialog was a feature, to improve user experience inside the simulator. Basically, before this feature whenever someone clicks on the Arduino icon on the top left corner, he/she directly exit from the schematic editor, this was okay but had a fair chance that the user might easily click on the button accidentally and might lose all his/her work or progress.
Now to overcome this, it was obvious to prompt the user before exiting the schematic editor. Hence to solve it, this feature was added to prompt the user every time he/she tries to exit the editor be it intentionally or unintentionally.



Do you want to exit?

Figure 2.3: Dialog to confirm whether user really wants to exit or not

2.3 Undo Redo

This feature is self-explanatory, it is one of the most common features in any modern editor. Undo redo functionality is a way in which a user can go back to immediate previous changes, or can revert them to normal. This feature comes in handy often while designing new circuits and some change is required due to some mistake or change in an idea. This is far better than delete as it is faster, allows reverting to the previous state as it is easier to go back and forth.

This feature was quite big, and hence took a bit of time in both implementation and testing of it properly. Also, it introduced some major changes in different files, to implement them in every action that the user does on an editor. For this feature, I added two buttons in the header, one for undo, which removes the latest change done & one for redo that is to revert change done through the undoing button.

If I talk about inner implementation then, I used a stack data structure which is being used to track all changes done in the editor, basically whenever a change is done, then it gets pushed into the stack, and when we click on undo that change gets pop and change is reverted in an editor, and for redo also there is a redo stack the does similar to undo but instead of logging each change in it, instead it only pushes a change in it if undo button is pressed. Below I have added a small flow chart that explains its bare minimum working.

¶

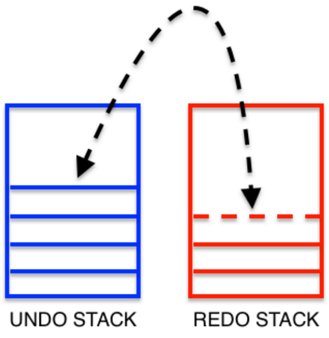


Figure 2.4: Internal working of undo & redo stack (data switches between both stacks)

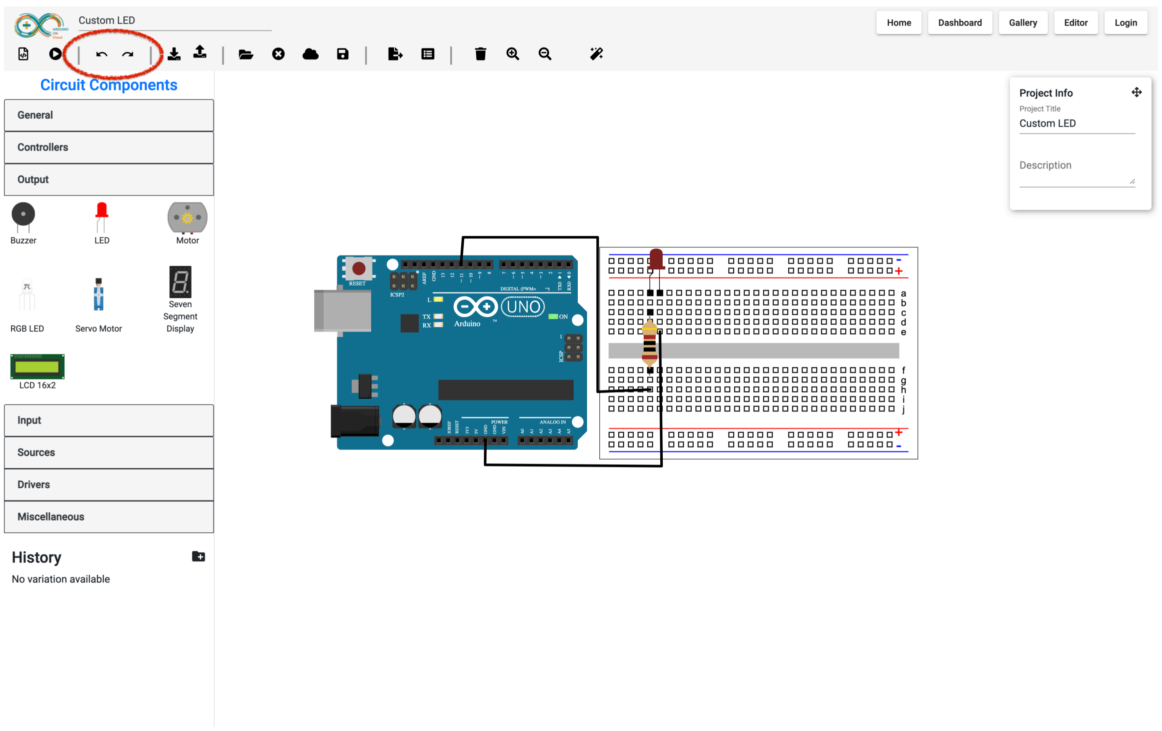


Figure 2.5: Placement of undo & redo buttons

2.4 Pwm Support

This is one of the features of Arduino that was missing in this simulator, AVR8js had this functionality but components were not able to react accordingly to the signals received. Firstly what is a PWM? PWM is a way to control analog devices with a digital output. Another way to put it is that you can output a modulating signal from a digital device such as an MCU to drive an analog device. It's one of the primary means by which MCUs drive analog devices like variable-speed motors, dimmable lights, actuators, and speakers. PWM is not true analog output, however. PWM "fakes" an analog-like result by applying power in pulses, or short bursts of regulated voltage [2]. The image given below describes the working of PWM in an Arduino.To add this functionality, it was required to be added separately in different components, below I have discussed about it in detail component wise.

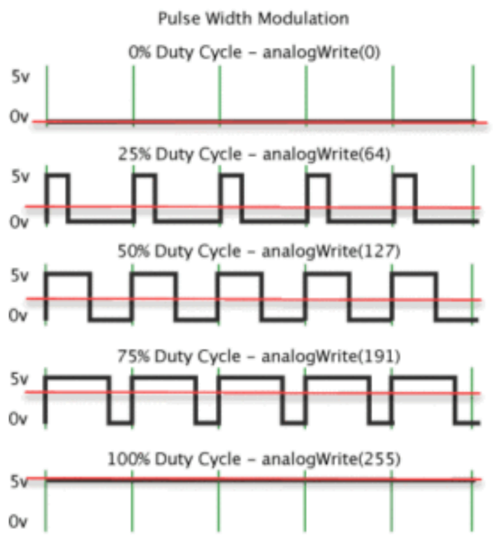
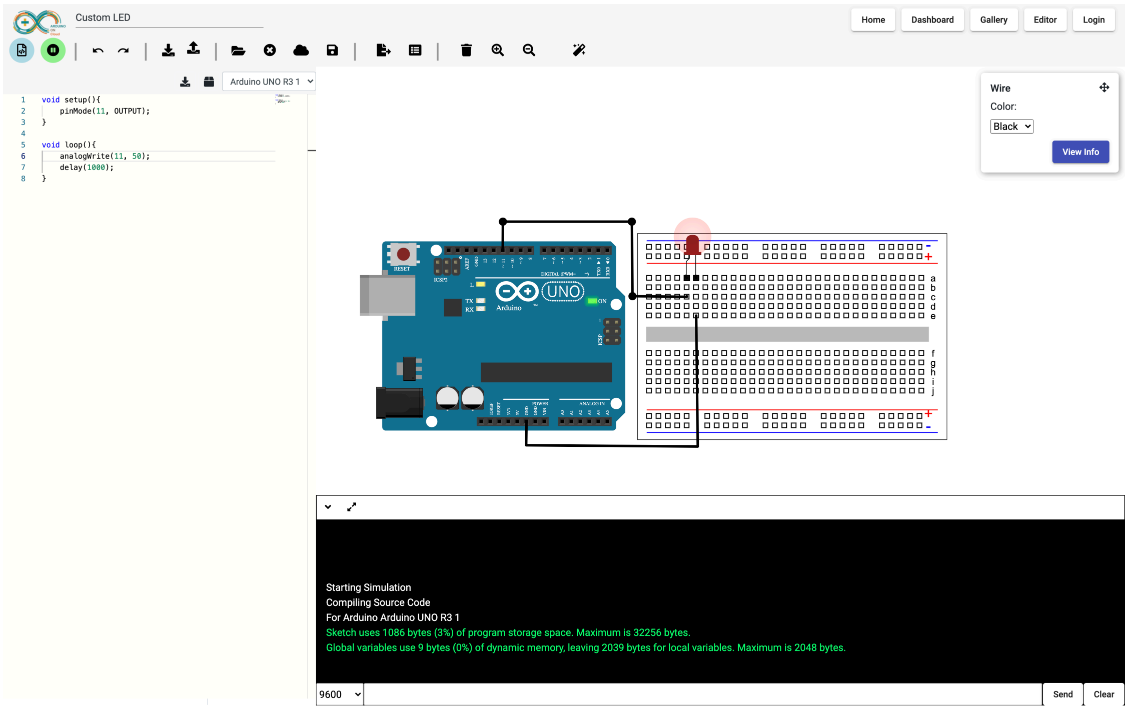

Pulse Width Modulation
0% Duty Cycle - analogWrite(0)
25% Duty Cycle - analogWrite(64)
50% Duty Cycle - analogWrite(127)
75% Duty Cycle - analogWrite(191)
100% Duty Cycle - analogWrite(255)
5v
0v

Figure 2.6: PWM actual working

2.5 FOR LED

LED was working fine for 5 voltages, and was able to turn on and off depending on voltage received from Arduino. But in the case of PWM, when Arduino was providing voltage in a range of 0 to 5, it was not aware of how to react. Hence the first task was to add a check-in LED to determine whether there is PWM functionality enabled or not, then afterward LED should react to the variable voltage received depending on PWM.
I added a check that was able to determine which pin is connected to Arduino, and afterward, it was able to change the voltage in LED after checking the PWM duty cycle in code. And lastly, I configured LED to work based on variable voltage, by changing the LED's alpha value depending on the voltage received. Below I have attached an image showing its working.¶



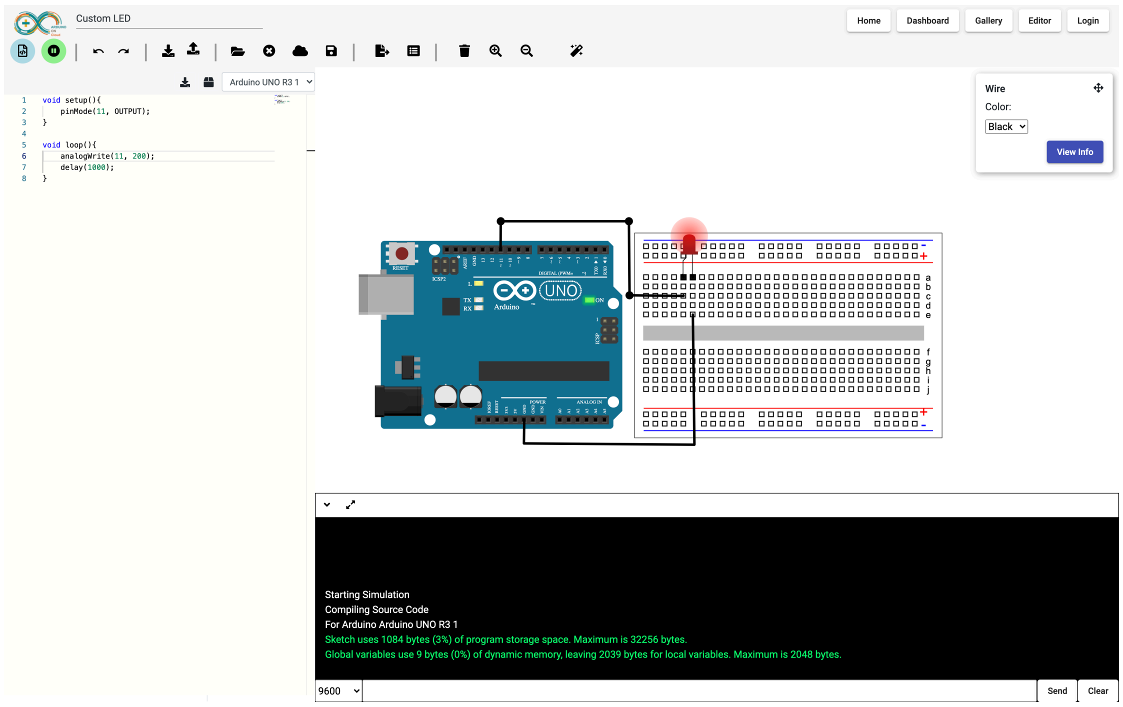
```
void setup(){
    pinMode(11, OUTPUT);
}

void loop(){
    analogWrite(11, 50);
    delay(1000);
}
```

Wire
Color: Black

```
Starting Simulation
Compiling Source Code
For Arduino Arduino UNO R3 1
Sketch uses 1086 bytes (3%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.
```

9600

Figure 2.7: LED brightness when duty cycle is set at 50



```
void setup(){
    pinMode(11, OUTPUT);
}

void loop(){
    analogWrite(11, 200);
    delay(1000);
}
```

Wire
Color: Black

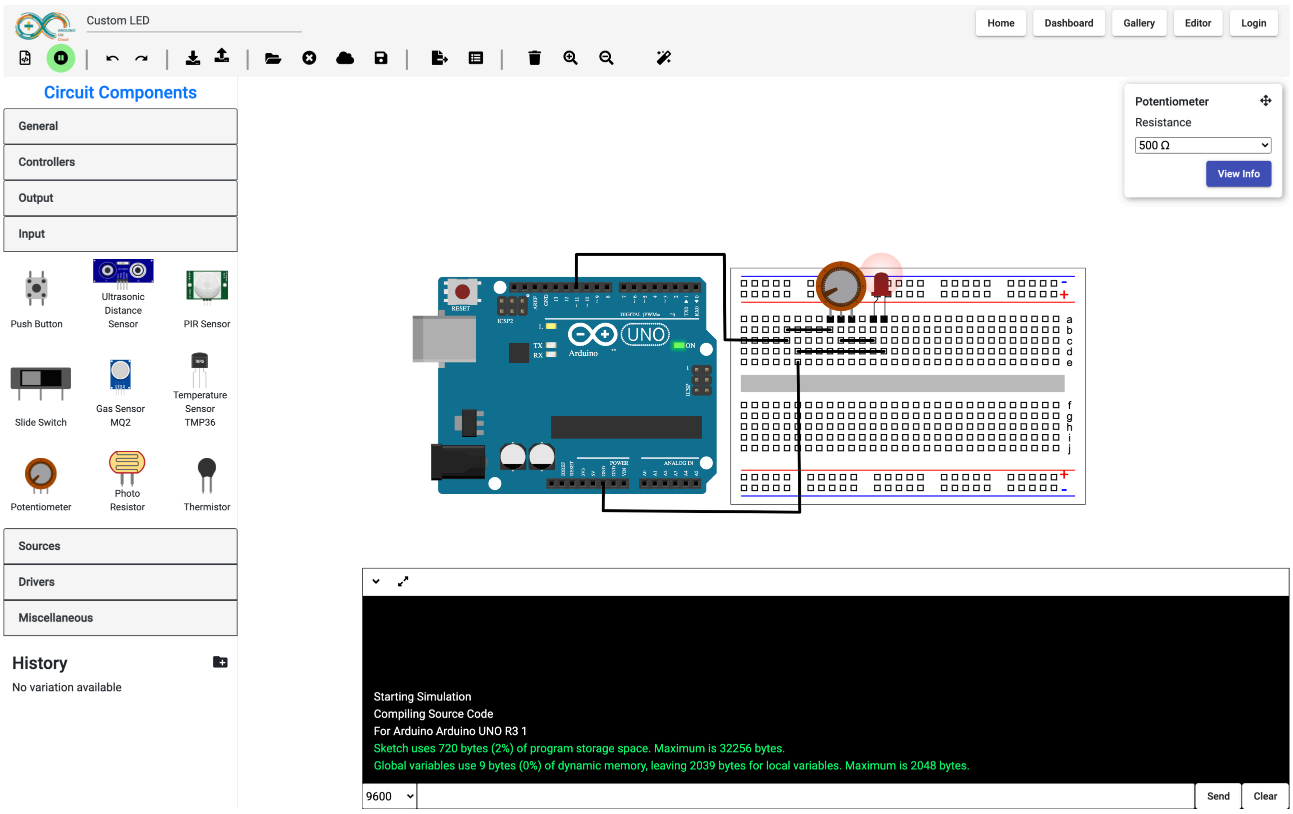
```
Starting Simulation
Compiling Source Code
For Arduino Arduino UNO R3 1
Sketch uses 1084 bytes (3%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.
```

9600

Figure 2.8: LED brightness when duty cycle is set at 200

2.6 Ability To Change Voltage Using Potentiometer

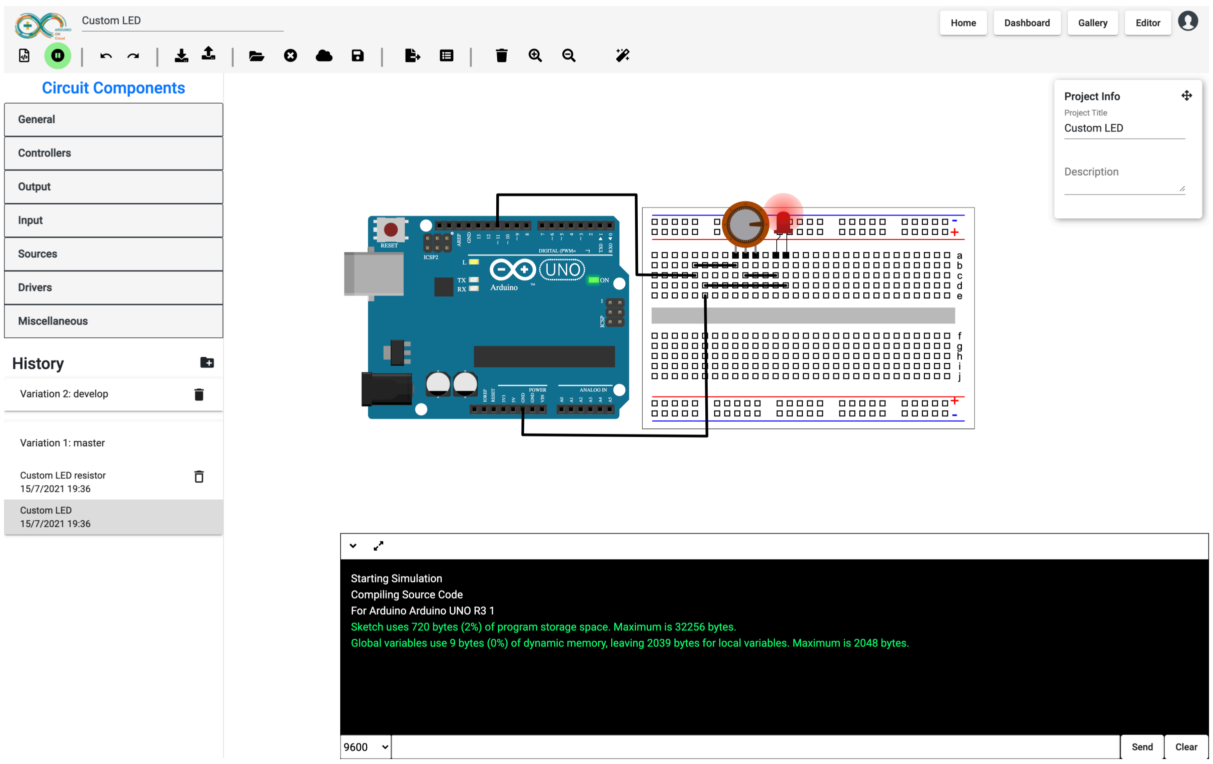
According to Wikipedia, A potentiometer is a three-terminal resistor with a sliding or rotating contact that forms an adjustable voltage divider. Potentiometers are commonly used to control electrical devices such as volume controls on audio equipment. Potentiometers operated by a mechanism can be used as position transducers, for example, in a joystick [3]. The potentiometer consists of 3 terminals, If only two terminals are used, one end and the wiper, it acts as a variable resistor or rheostat. This component was available already in the simulator, but it was not functional in some components like LED and motor.
Because it was already there hence I was only supposed to make changes in the respective code of the component, leaving aside the task to add it from scratch.
Below I have attached a few pictures which show working of it. ¶



Potentiometer
Resistance
500 Ω

```
Starting Simulation
Compiling Source Code
For Arduino Arduino UNO R3 1
Sketch uses 720 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.
```

Figure 2.9: LED's brightness when potentiometer's knob is set at low



```
Starting Simulation
Compiling Source Code
For Arduino Arduino UNO R3 1
Sketch uses 720 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.
```

Figure 2.10: LED's brightness when potentiometer's knob is set at high

2.7 Versioning Support

Versioning support was one of the most necessary features needed in this project as a few other features were also depending on this feature like LMS support etc. hence it was very necessary to take this up and work upon it.
First, let's try to understand what this is and how it works, every time we save something on our simulator then earlier the existing project gets updated, whereas if he/she makes some mistake in the future then he/she will never be able to go back to an earlier state. Hence to solve this issue it was necessary to implement some kind of version controlling flow in the simulator, hence to overcome it versioning support was implemented in the project.
To implement this APIs were already present hence only frontend API called was supposed to be implemented, which I did. After its implementation user was able to create new versions after every update of the project, able to create branches in the project to separate different versions.
Below I have attached a few pictures depicting actual implementation.

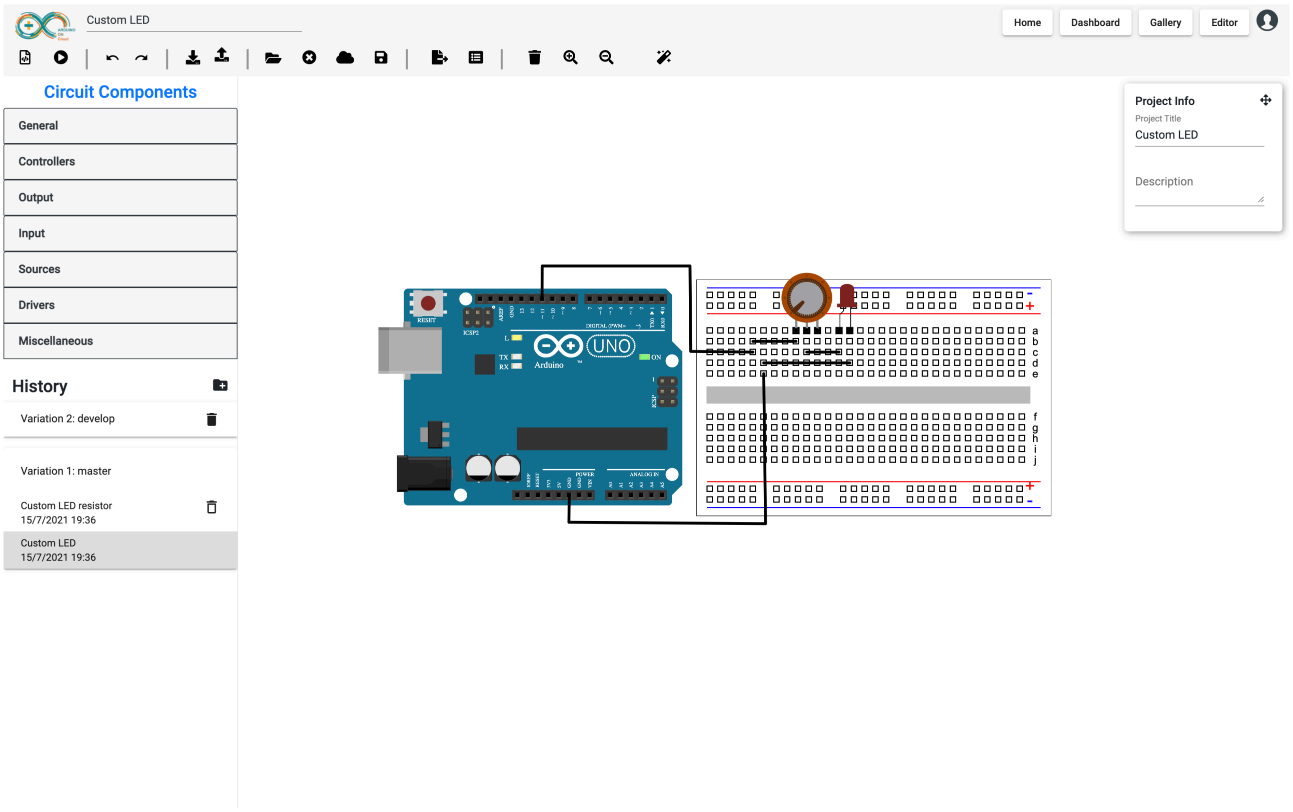


Figure 2.11: Multiple versions of project in left panel

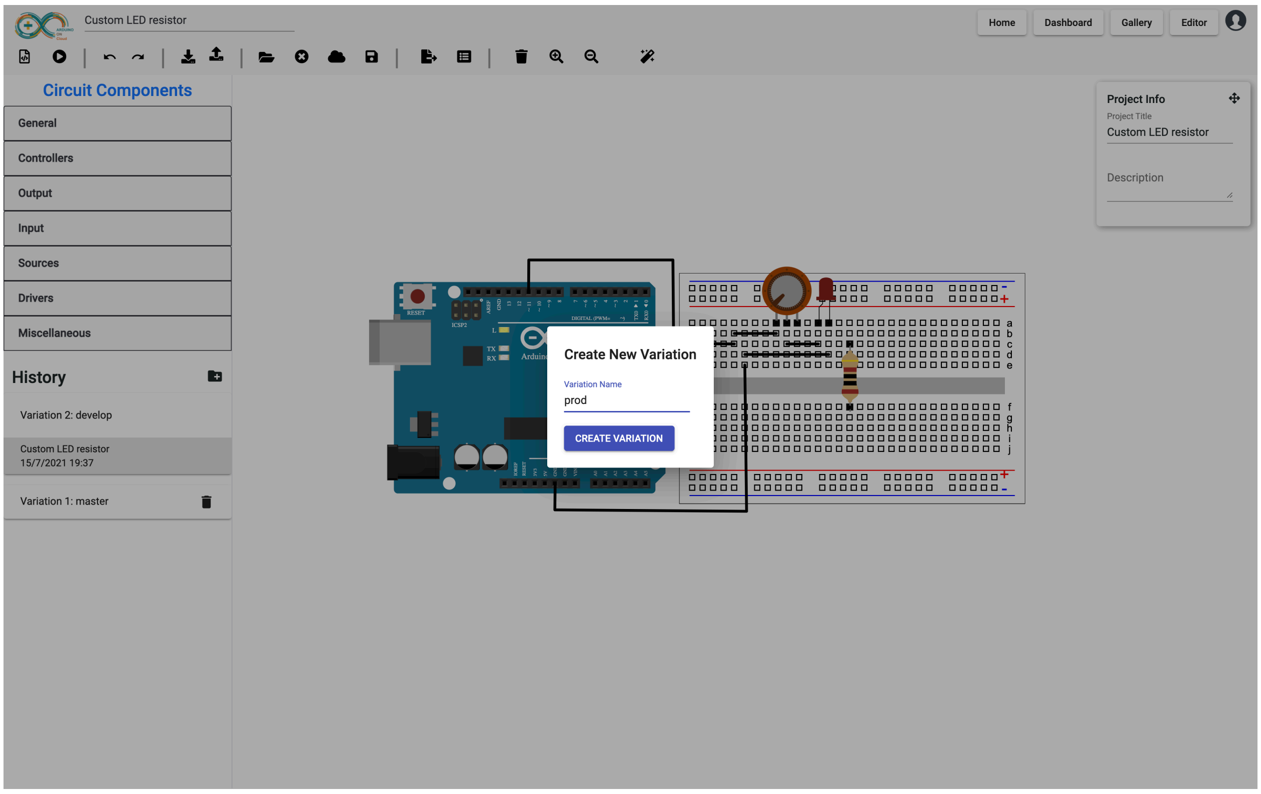


Figure 2.12: Dialog to set name of new branch while creating it

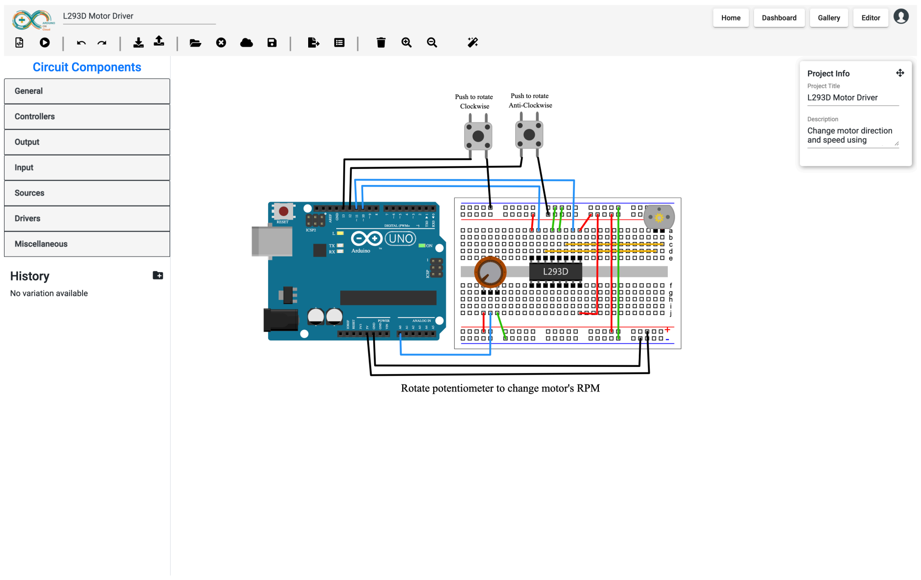
Chapter 3

Component Additions

A few components were to be added in this simulator as they were missing, and were needed in order to allow users to experiment with new circuits and be more flexible in his/ her imagination and creativity. The addition of each component consisted of a few steps which included adding the image of component, adding component in left panelist, adding component code in respective TS file. Some components needed little extra modifications in the code of the Arduino.ts file. Below I've discussed all these components I added in a bit detail.

3.1 L293D Motor Driver

The L293D is a popular 16-Pin Motor Driver IC. As the name suggests it is mainly used to drive motors. A single L293D IC is capable of running two DC motors at the same time; also the direction of these two motors can be controlled independently. All the Ground pins should be grounded. There are two power pins for this IC, one is the Vss(Vcc1) which provides the voltage for the IC to work, this must be connected to +5V. The other is Vs(Vcc2) which provides voltage for the motors to run, based on the specification of your motor you can connect this pin to anywhere between 4.5V to 36V, here I have connected to +12V.
The Enable pins (Enable 1,2 and Enable 3,4) are used to Enable Input pins for Motor 1 and Motor 2 respectively. Since in most cases we will be using both the motors both the pins are held high by default by connecting to +5V supply. The input pins Input 1,2 are used to control motor 1 and Input pins 3,4 are used to control Motor 2. The input pins are connected to any Digital circuit or microcontroller to control the speed and direction of the motor. You can toggle the input pins based on the following table to control your motor [4].



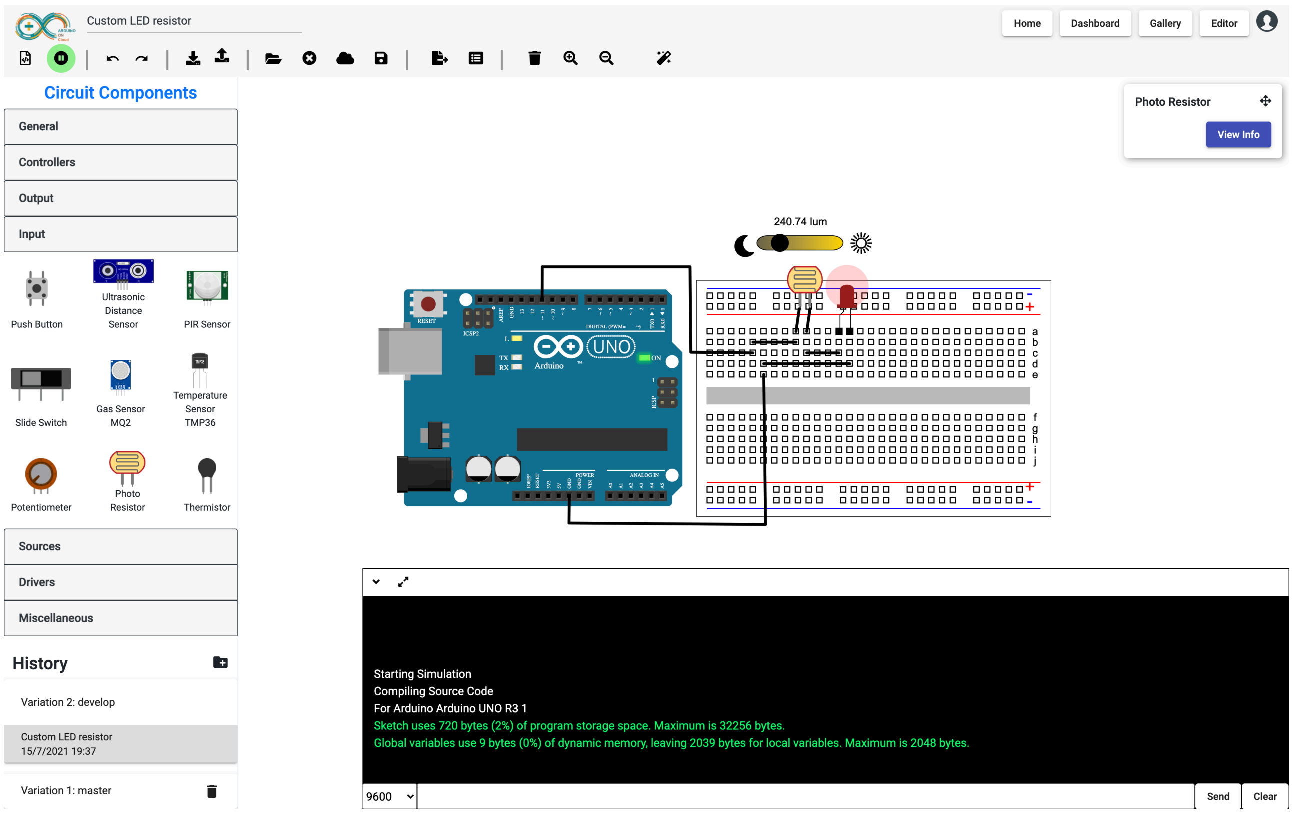
Project Info
Project Title
L293D Motor Driver
Description
Change motor direction and speed using

Figure 3.1: Circuit with L293D Motor Driver

The addition of this component was really important as it was absolutely necessary to control the motor in some of the circuits.

3.2. Light Detecting Resistor

A Light Dependent Resistor (LDR) is also called a photoresistor or a cadmium sulfide (CdS) cell. It is also called a photoconductor. It is basically a photocell that works on the principle of photoconductivity. The passive component is basically a resistor whose resistance value decreases when the intensity of light decreases. This optoelectronic device is mostly used in light varying sensor circuits, and light and dark activated switching circuits. Some of its applications include camera light meters, street lights, clock radios, light beam alarms, reflective smoke alarms, and outdoor clocks.
LDR was also an important component to be added. To add this component a few changes were required in existing components like support for variable voltage. This was already available in motor and LED. Hence only the logic part of LDR was required in its ts file.
I added a slider with the component in order to change the resistance of the LDR, with the help of it user is able to change the resistance of the LDR component.



```
Starting Simulation
Compiling Source Code
For Arduino Arduino UNO R3 1
Sketch uses 720 bytes (2%) of program storage space. Maximum is 32256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes for local variables. Maximum is 2048 bytes.
```

Figure 3.2: Circuit with LDR

3.3. Thermistor

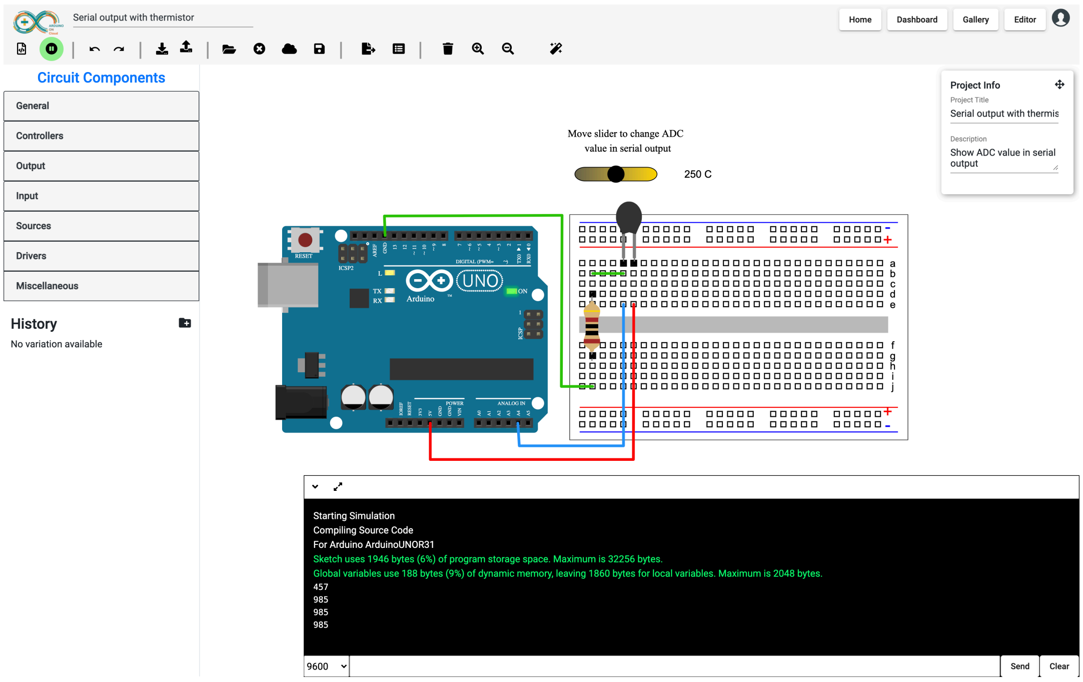
A thermistor is a resistance thermometer or a resistor whose resistance is dependent on temperature. The term is a combination of “thermal” and “resistor”. It is made of metallic oxides, pressed into a bead, disk, or cylindrical shape, and then encapsulated with an impermeable material such as epoxy or glass.
In general, there are two thermistors one of them is NTC which stands for Negative Temperature Coefficient, and another is PTC which is Positive Temperature Coefficient. NTC tends to decrease resistance when temperature increase. On the other hand PTC resistance increase with an increase in temperature. One term “rating“ of a thermistor stands for the internal resistance of the thermistor at room temperature.

Here we have configured NTC in the simulator, with a rating of 10K at room temperature. It consists of a slider that ranges from 0 to 500 degrees Celsius. When we change the temperature using a slider then internal resistance also changes and hence the voltage through the thermistor also changes.
Following is the formula to calculate the internal resistance of a resistor:[5]

$$R_2 = R_1 e^{\beta\left(\frac{1}{T_1} - \frac{1}{T_2}\right)}$$

Figure 3.3: Formula of internal resistance of thermistor

Where ß(beta) is calculated using existing values of temperature. T is in kelvin, and e is 2.71.
Below are the pictures showing working of thermistor in simulator.



```
Starting Simulation
Compiling Source Code
For Arduino ArduinoUNOR31
Sketch uses 1946 bytes (6%) of program storage space. Maximum is 32256 bytes.
Global variables use 188 bytes (9%) of dynamic memory, leaving 1860 bytes for local variables. Maximum is 2048 bytes.
457
985
985
985
```

Figure 3.4: Circuit with thermistor

Chapter 4

Examples

Apart from addition of new components, their circuits were also added along with their Pull Requests.

4.1. Led With Potentiometer

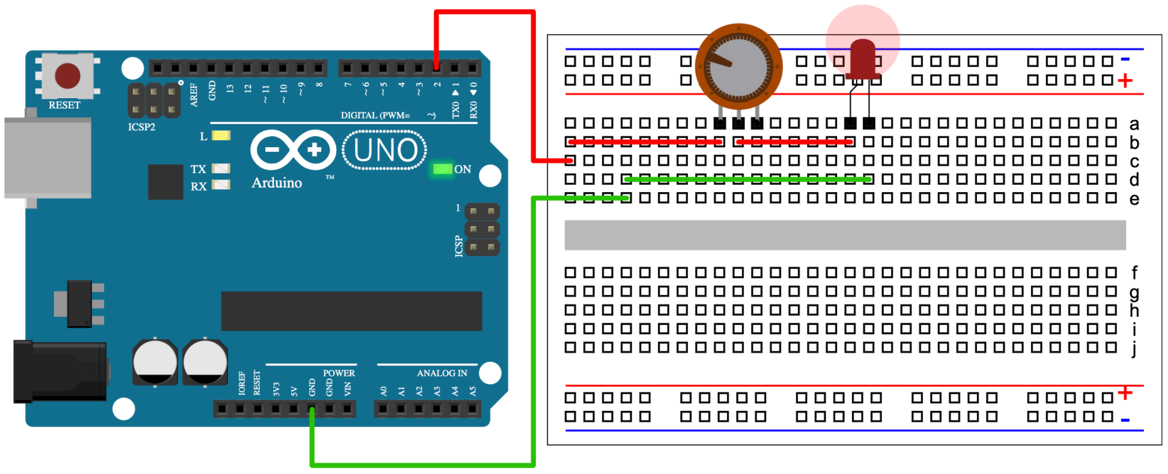
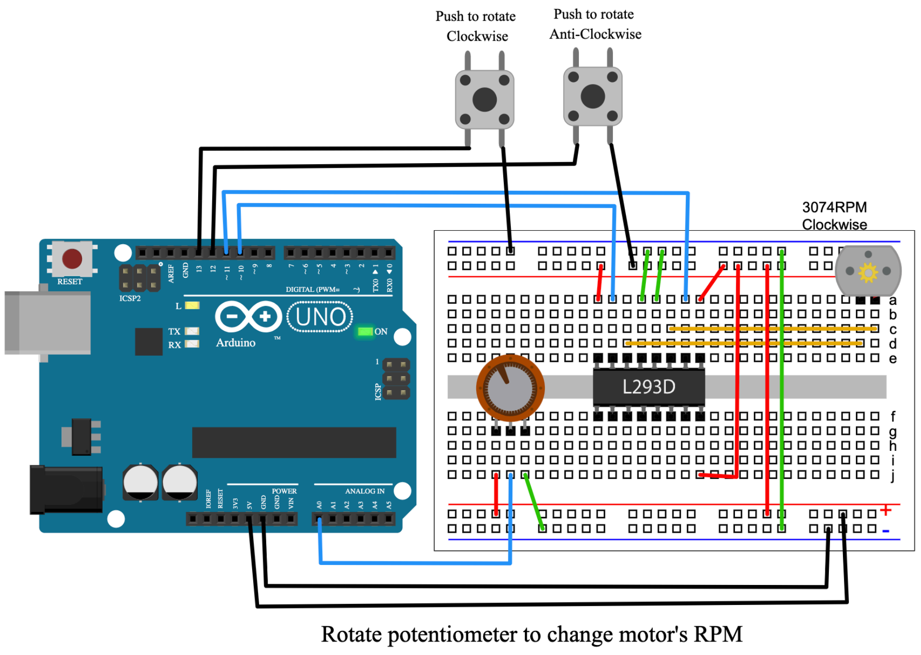


Figure 4.1: LED with potentiometer circuit



Push to rotate Clockwise
Push to rotate Anti-Clockwise
3074RPM Clockwise
L293D
Rotate potentiometer to change motor's RPM

Figure 4.2: L293D Motor Driver circuit

4.2. L293D Motor Driver

4.3. Led With Ldr

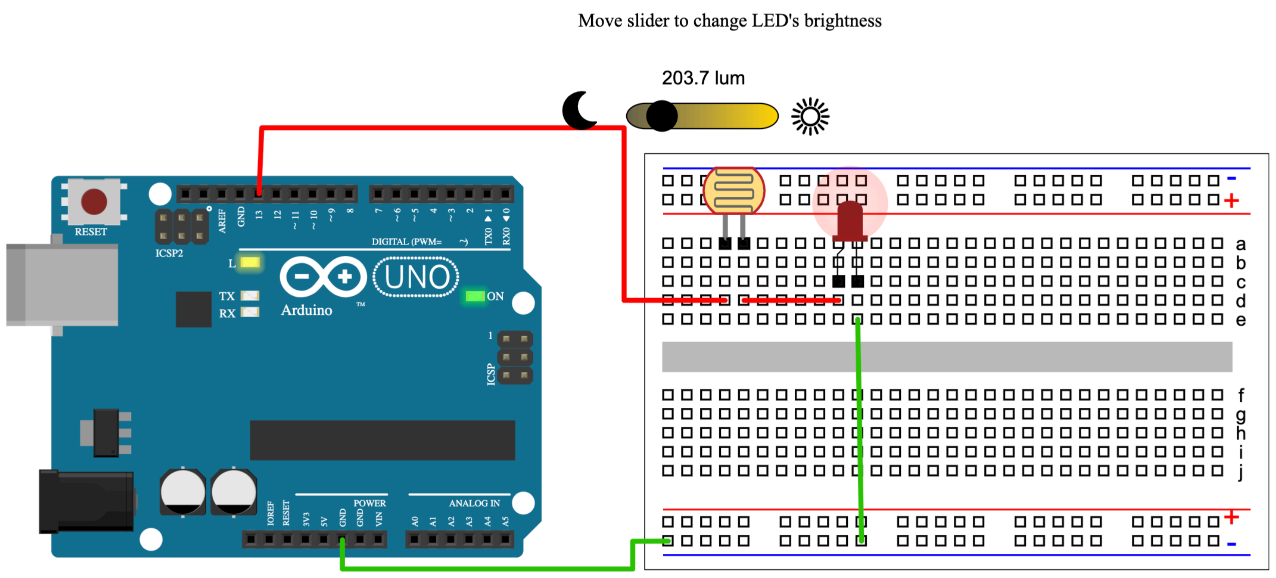


Figure 4.3: LED with LDR circuit

4.4. Serial Output With Thermistor

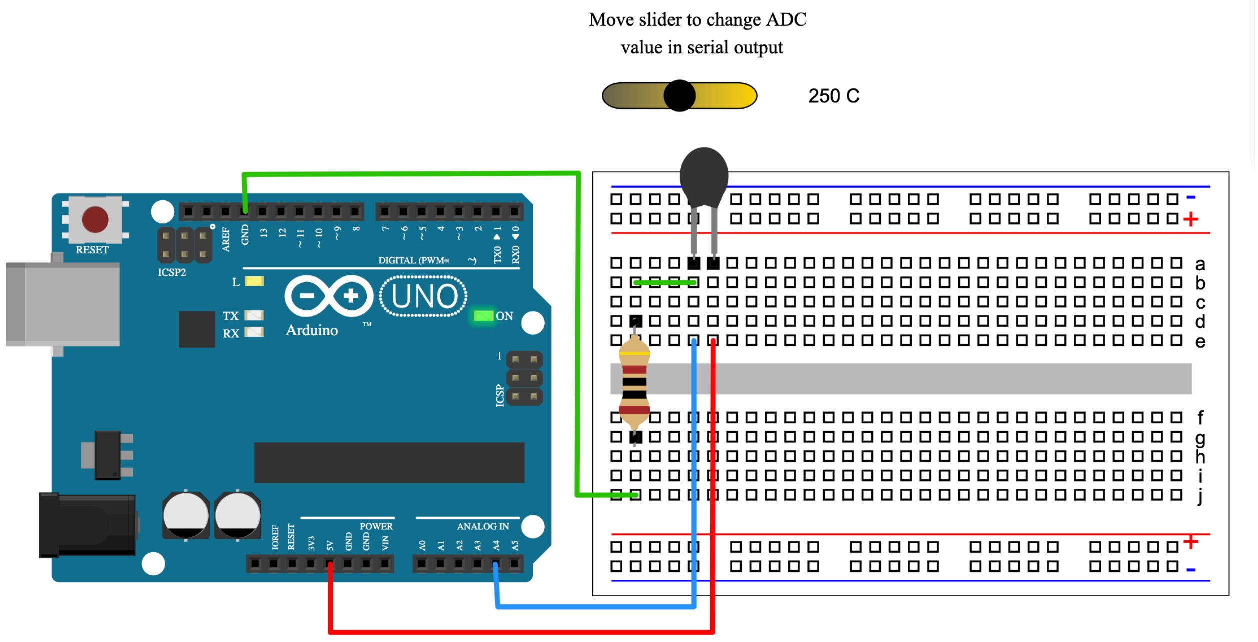


Figure 4.4: Serial output with thermistor circuit

4.5. Led’S Brightness Using Thermistor

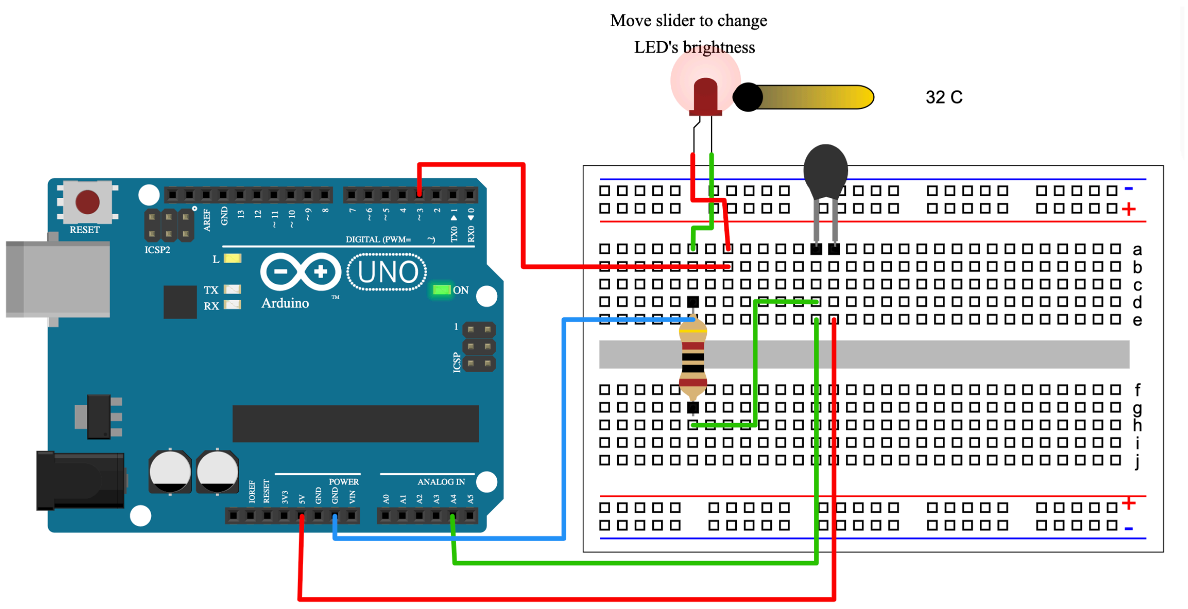


Figure 4.5: LED's brightness using Thermistor circuit

Chapter 5

Bug Fixes

Apart from feature additions, component and examples additions provided above, I even fixed some bugs in it that were already there, some of which were critical enough to stop the normal functioning of simulator. Below I have provided description of all of them in a bit detail.

5.1. Delete/Backspace Button Bug

This was an existing bug, in which whenever someone tries to type something in text box then in editor delete functionality gets triggered automatically, as backspace was connected to delete function in editor which should not happen instead it should be disabled by default on textures and input fields.

Steps to replicate the issue:
1. Drag a component from the left side into schematic
2. Click on any text field on the website
3. Hit backspace in textfield
4. The component dragged gets deleted if it's selected, if it's not selected then a banner shows up indicating "No element is selected".

Expected Behaviour
When a user is typing in a textfield, Instead of deleting the component, it should just delete the text in the textfield.

OS tested on: Ubuntu(Linux)

Solution Proposed: Added a condition to check if someone is pressing backspace on input or textarea.

BUG REPORT: **https://github.com/frg-fossee/eSim-Cloud/issues/237**
PR: **https://github.com/frg-fossee/eSim-Cloud/pull/241**

5.2. Save Bug

Due to this bug user was unable to save Arduino circuit on cloud, it was an induced bug due to some changes in backend API of project.

Steps to reproduce:
1. Create any circuit
2. Click on Save Project (cloud)
3. API will throw an error & dialog will show a lot of numbers.

Current Behaviour:
Error in API - status 500

Expected Behaviour:
API should return 200

Tested on: macOS (prod build)

Solution Proposed:
To fix this error following conditions were added in save API :-
1. checks if **'esim_libraries'** is present in request.
2. checks if request **'is_arduino'**.

BUG REPORT: https://github.com/frg-fossee/eSim-Cloud/issues/279
PR: https://github.com/frg-fossee/eSim-Cloud/pull/280

5.3. Breadboard Issue

Description: BreadBoard has some issue in it, and does give errors sometimes.
Tested on: macOS develop branch & live website
How to reproduce the error:
1. Replicate the circuit with the code below (you can try with other circuits also)
2. Click on the "start simulation" button
3. If an error does not persist then clear the circuit and replicate the circuit again(as this error occurs randomly).
Take out one BreadBoard from left panel
Move it somewhere by dragging it.
Click on 'Start Sinulation'.
TESTING
Circuit:

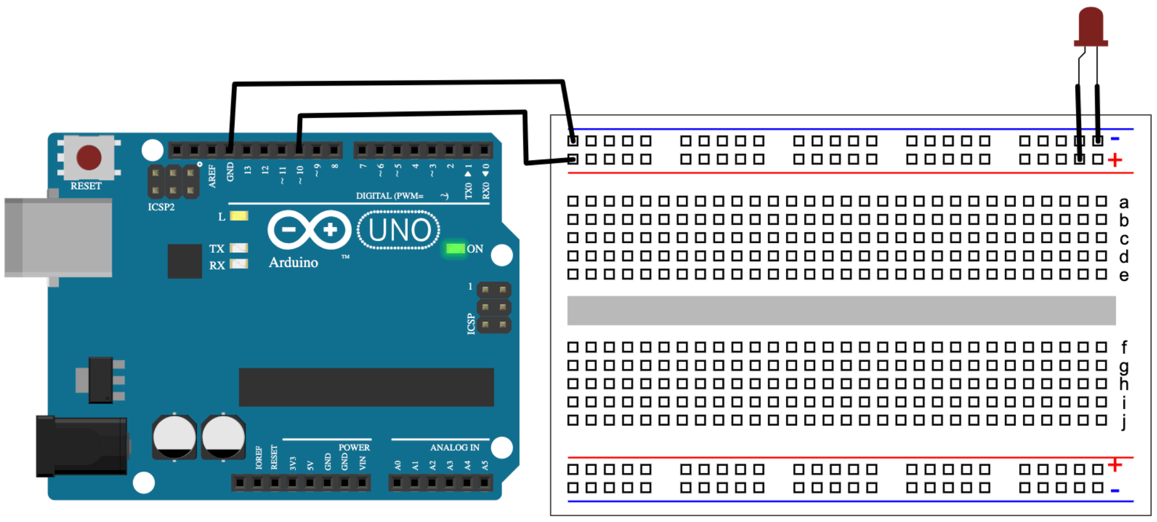
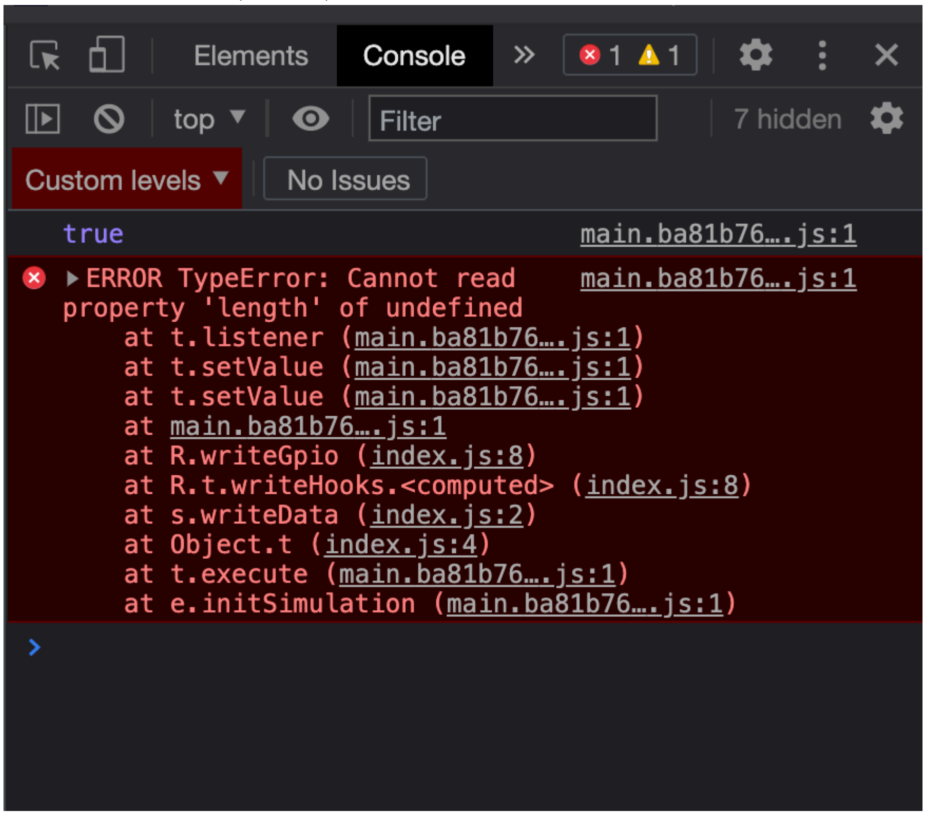


Figure 5.1: Circuit on which testing was caried

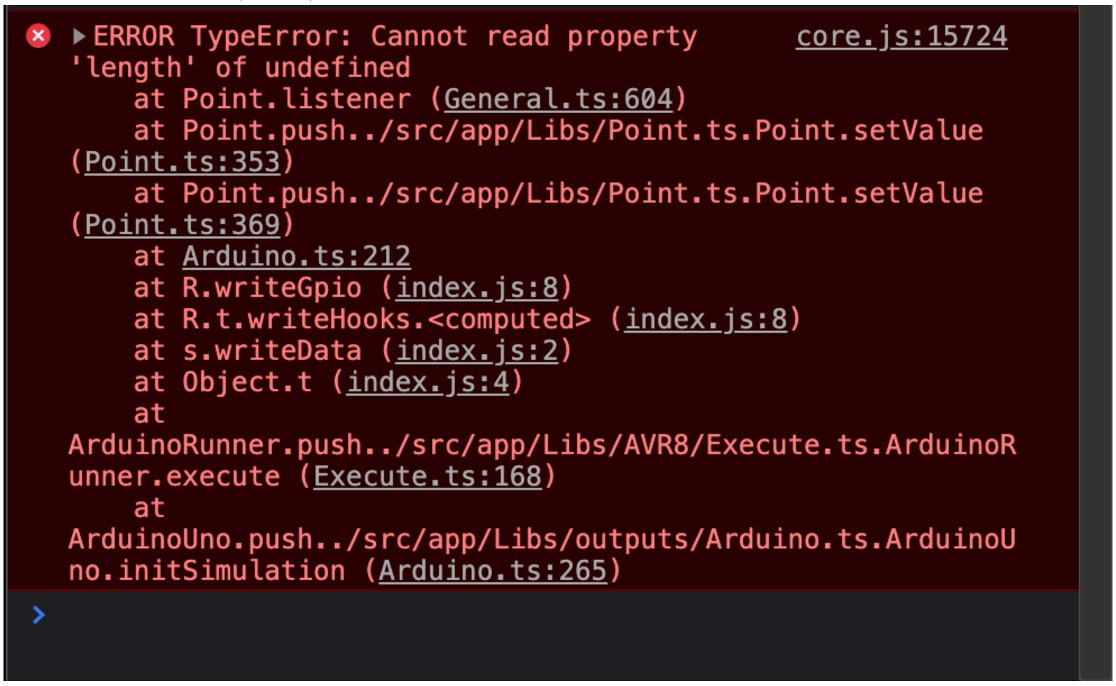
Console Error(Prod):



```
true                                         main.ba81b76….js:1
ERROR TypeError: Cannot read                 main.ba81b76….js:1
property 'length' of undefined
    at t.listener (main.ba81b76….js:1)
    at t.setValue (main.ba81b76….js:1)
    at t.setValue (main.ba81b76….js:1)
    at main.ba81b76….js:1
    at R.writeGpio (index.js:8)
    at R.t.writeHooks.<computed> (index.js:8)
    at s.writeData (index.js:2)
    at Object.t (index.js:4)
    at t.execute (main.ba81b76….js:1)
    at e.initSimulation (main.ba81b76….js:1)
```

Figure 5.2: Error in console (Production)

Console Error(Dev):

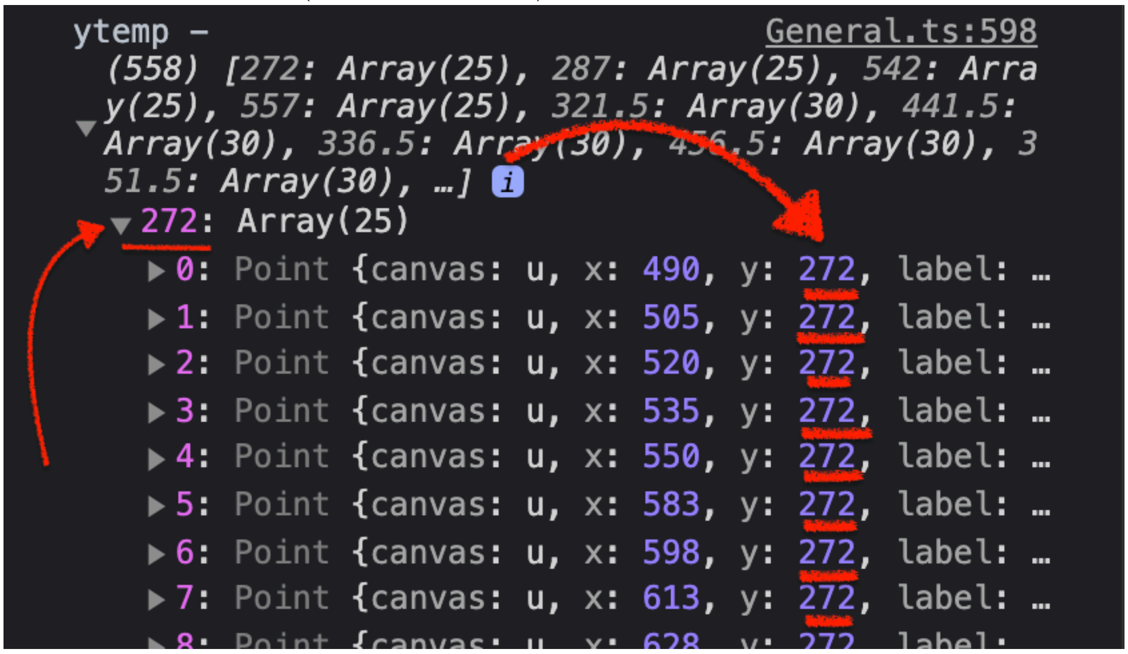


```
ERROR TypeError: Cannot read property          core.js:15724
'length' of undefined
    at Point.listener (General.ts:604)
    at Point.push../src/app/Libs/Point.ts.Point.setValue
(Point.ts:353)
    at Point.push../src/app/Libs/Point.ts.Point.setValue
(Point.ts:369)
    at Arduino.ts:212
    at R.writeGpio (index.js:8)
    at R.t.writeHooks.<computed> (index.js:8)
    at s.writeData (index.js:2)
    at Object.t (index.js:4)
    at
ArduinoRunner.push../src/app/Libs/AVR8/Execute.ts.ArduinoR
unner.execute (Execute.ts:168)
    at
ArduinoUno.push../src/app/Libs/outputs/Arduino.ts.ArduinoU
no.initSimulation (Arduino.ts:265)
```

Figure 5.3: Error in console (Dev)

REASON

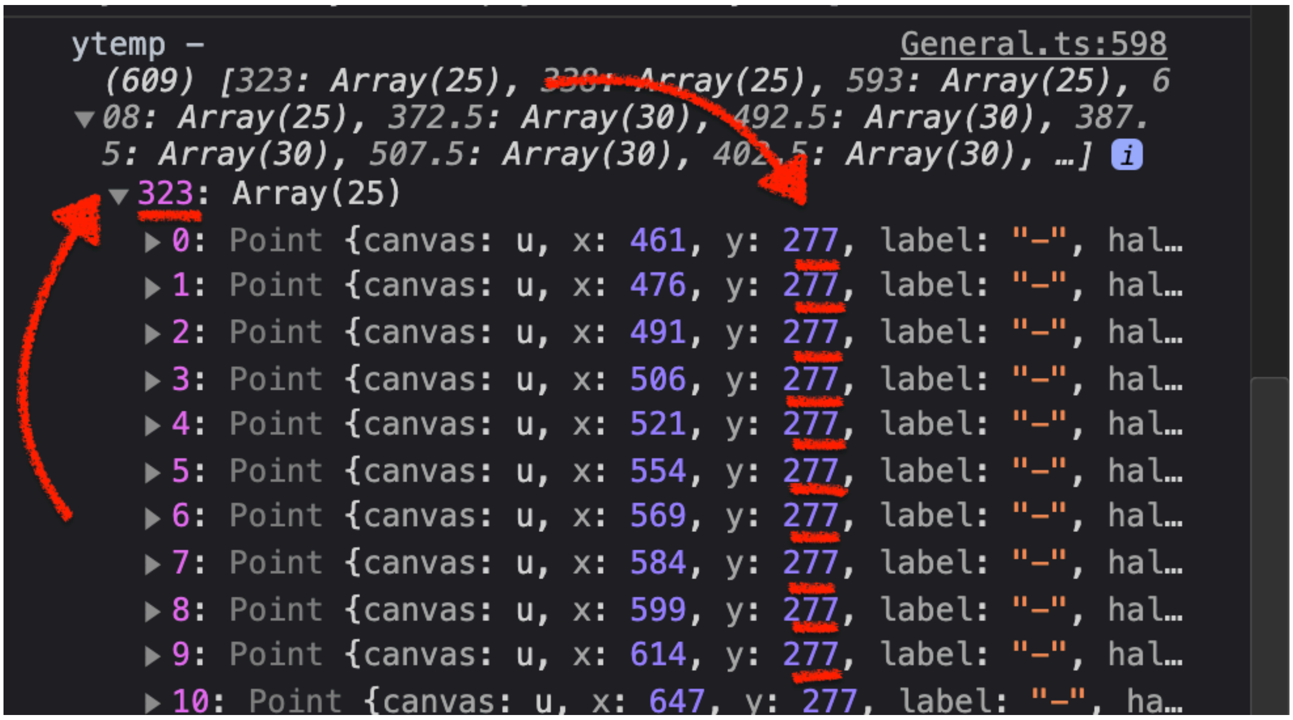
I did a bit of debugging and found that it is caused by Breadboard in General.ts. the issue is caused because the sameXNodes & sameYNodes aren't getting created as expected every time. Technically these arrays should consist of nodes with similar x & y coordinates in their respective keys (see below image).



```
ytemp -                                              General.ts:598
  (558) [272: Array(25), 287: Array(25), 542: Arra
▼ y(25), 557: Array(25), 321.5: Array(30), 441.5:
  Array(30), 336.5: Array(30), 456.5: Array(30), 3
  51.5: Array(30), …] i
 ▼272: Array(25)
   ▶0: Point {canvas: u, x: 490, y: 272, label: …
   ▶1: Point {canvas: u, x: 505, y: 272, label: …
   ▶2: Point {canvas: u, x: 520, y: 272, label: …
   ▶3: Point {canvas: u, x: 535, y: 272, label: …
   ▶4: Point {canvas: u, x: 550, y: 272, label: …
   ▶5: Point {canvas: u, x: 583, y: 272, label: …
   ▶6: Point {canvas: u, x: 598, y: 272, label: …
   ▶7: Point {canvas: u, x: 613, y: 272, label: …
   ▶8: Point {canvas: u, x: 628, y: 272, label:
```

Figure 5.4: Array (same values of y as key)

In Above image, you can observe that the key is matching with y coordinates in its items too. But if I consider error one then,



```
ytemp -                                                  General.ts:598
  (609) [323: Array(25), 338: Array(25), 593: Array(25), 6
▼ 08: Array(25), 372.5: Array(30), 492.5: Array(30), 387.
  5: Array(30), 507.5: Array(30), 402.5: Array(30), …] i
 ▼323: Array(25)
   ▶0: Point {canvas: u, x: 461, y: 277, label: "-", hal…
   ▶1: Point {canvas: u, x: 476, y: 277, label: "-", hal…
   ▶2: Point {canvas: u, x: 491, y: 277, label: "-", hal…
   ▶3: Point {canvas: u, x: 506, y: 277, label: "-", hal…
   ▶4: Point {canvas: u, x: 521, y: 277, label: "-", hal…
   ▶5: Point {canvas: u, x: 554, y: 277, label: "-", hal…
   ▶6: Point {canvas: u, x: 569, y: 277, label: "-", hal…
   ▶7: Point {canvas: u, x: 584, y: 277, label: "-", hal…
   ▶8: Point {canvas: u, x: 599, y: 277, label: "-", hal…
   ▶9: Point {canvas: u, x: 614, y: 277, label: "-", hal…
   ▶10: Point {canvas: u, x: 647, y: 277, label: "-", ha…
```

Figure 5.5: Array (different values of y as key)

the key is not matching the y coordinates in the array.
Due to this error shows up, ultimately stopping the application.

The error is throwing up in the General.ts in the init() function.
I did try a bit to fix this but can't find really why this change is happening in keys, at the moment i think its completely random, as i tried every bit to find the exact cause of the problem,

UPDATE
The issue is being caused due to dragging on breadboard. The closest guess is that sameXNode and sameYNode variable are not able to retain the change in node's coordinates

TEMPORARY FIX:
Temporary fix proposed to this issue was to either export the circuit or save it, and then import the circuit again after refreshing page. A new imported circuit never gives this error.

Permanent Solution Proposed:
Solution to this problem was to, each time the breadboard will be **dragged** a **function** will be called to rebuild SameXNodes & SameYNodes objects.

BUG REPORT: https://github.com/frg-fossee/eSim-Cloud/issues/291
PR: https://github.com/frg-fossee/eSim-Cloud/pull/297

5.4. Delete Component Issue

Description: In some circuits when we delete any wire, then other component's wire too get deleted

TESTED ON CIRCUIT:

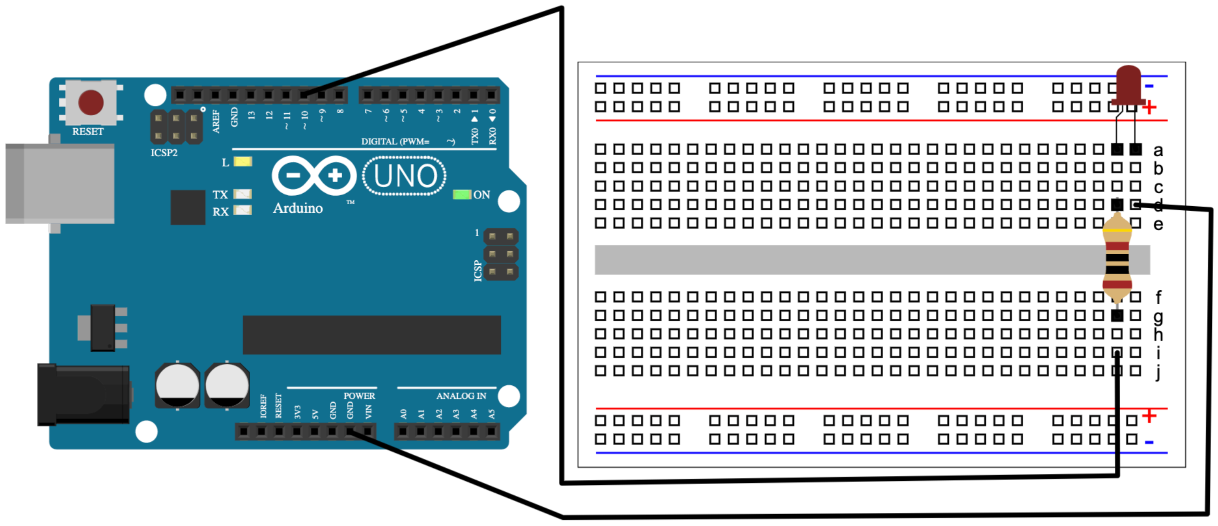


Figure 5.6: Basic LED circuit

WIRE DELETED:

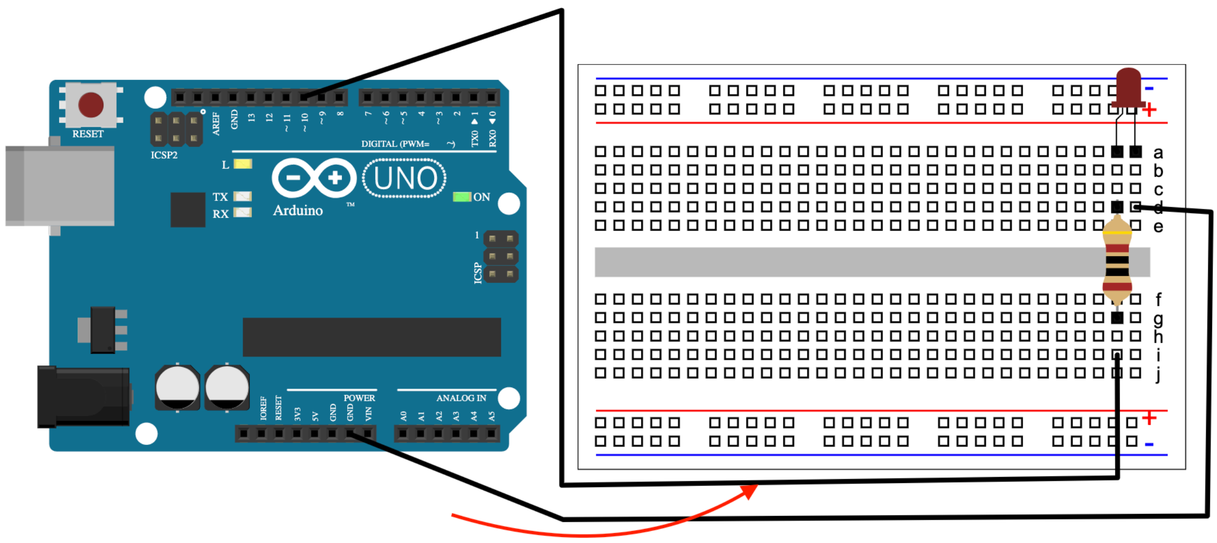


Figure 5.7: Wire to delete

EFFECT ON LED's WIRE:

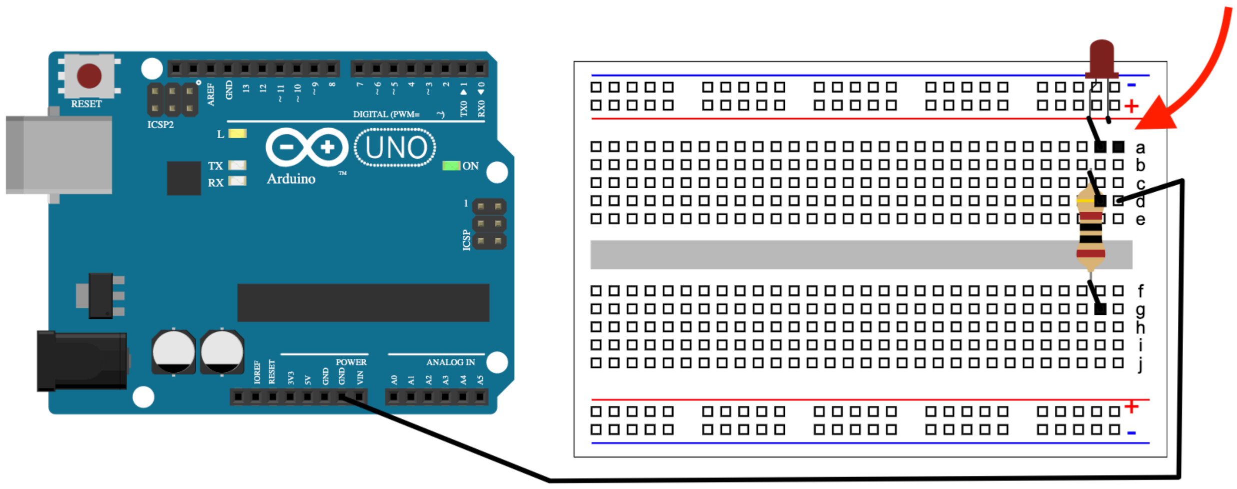
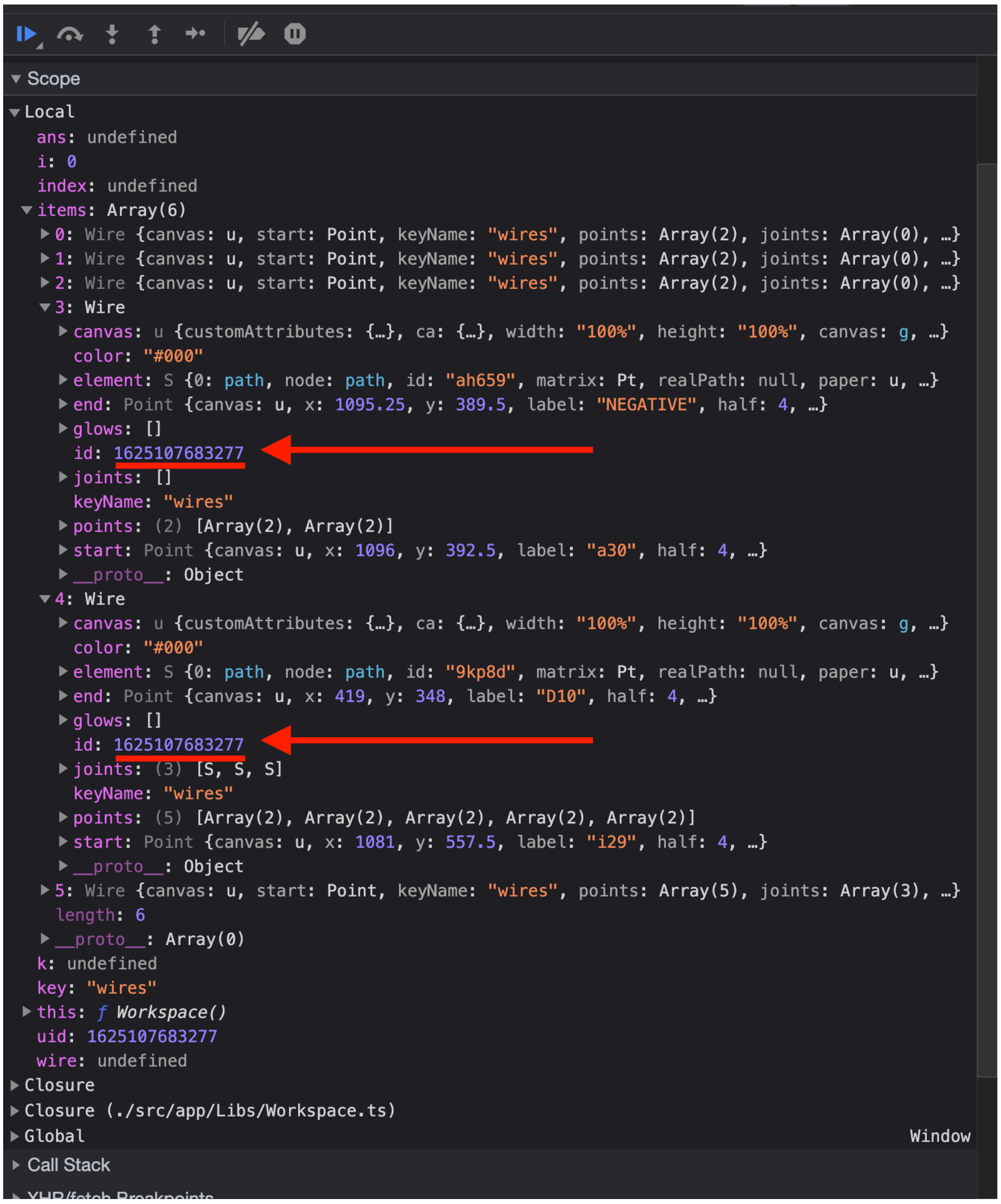


Figure 5.8: Connection between LED and breadboard gets detached

REASON:
I did a bit of debugging and encountered an issue, which I think is the reason for it. Actually when the circuit is being created then two wires get the same id. Due to this reason when one wire with same id is deleted then another wire too gets deleted partially. (See pictures below)



```
▼ Scope
▼ Local
    ans: undefined
    i: 0
    index: undefined
  ▼ items: Array(6)
    ▶ 0: Wire {canvas: u, start: Point, keyName: "wires", points: Array(2), joints: Array(0), …}
    ▶ 1: Wire {canvas: u, start: Point, keyName: "wires", points: Array(2), joints: Array(0), …}
    ▶ 2: Wire {canvas: u, start: Point, keyName: "wires", points: Array(2), joints: Array(0), …}
    ▼ 3: Wire
      ▶ canvas: u {customAttributes: {…}, ca: {…}, width: "100%", height: "100%", canvas: g, …}
        color: "#000"
      ▶ element: S {0: path, node: path, id: "ah659", matrix: Pt, realPath: null, paper: u, …}
      ▶ end: Point {canvas: u, x: 1095.25, y: 389.5, label: "NEGATIVE", half: 4, …}
      ▶ glows: []
        id: 1625107683277
      ▶ joints: []
        keyName: "wires"
      ▶ points: (2) [Array(2), Array(2)]
      ▶ start: Point {canvas: u, x: 1096, y: 392.5, label: "a30", half: 4, …}
      ▶ __proto__: Object
    ▼ 4: Wire
      ▶ canvas: u {customAttributes: {…}, ca: {…}, width: "100%", height: "100%", canvas: g, …}
        color: "#000"
      ▶ element: S {0: path, node: path, id: "9kp8d", matrix: Pt, realPath: null, paper: u, …}
      ▶ end: Point {canvas: u, x: 419, y: 348, label: "D10", half: 4, …}
      ▶ glows: []
        id: 1625107683277
      ▶ joints: (3) [S, S, S]
        keyName: "wires"
      ▶ points: (5) [Array(2), Array(2), Array(2), Array(2), Array(2)]
      ▶ start: Point {canvas: u, x: 1081, y: 557.5, label: "i29", half: 4, …}
      ▶ __proto__: Object
    ▶ 5: Wire {canvas: u, start: Point, keyName: "wires", points: Array(5), joints: Array(3), …}
      length: 6
    ▶ __proto__: Array(0)
    k: undefined
    key: "wires"
  ▶ this: ƒ Workspace()
    uid: 1625107683277
    wire: undefined
▶ Closure
▶ Closure (./src/app/Libs/Workspace.ts)
▶ Global                                                                    Window
▶ Call Stack
▶ XHR/fetch Breakpoints
```

Figure 5.9: Reason for detachment, same IDs of wires

In the above image, you can observe that id of two wires is same whereas the end and start nodes are different and has nothing to do with each other.
Tested on: Live website & develop branch
OS: macOS

SOLUTION PROPOSED:
Added a function that checks for similar id, if present then returns another unique id.

BUG REPORT: https://github.com/frg-fossee/eSim-Cloud/issues/300
PR: https://github.com/frg-fossee/eSim-Cloud/pull/303

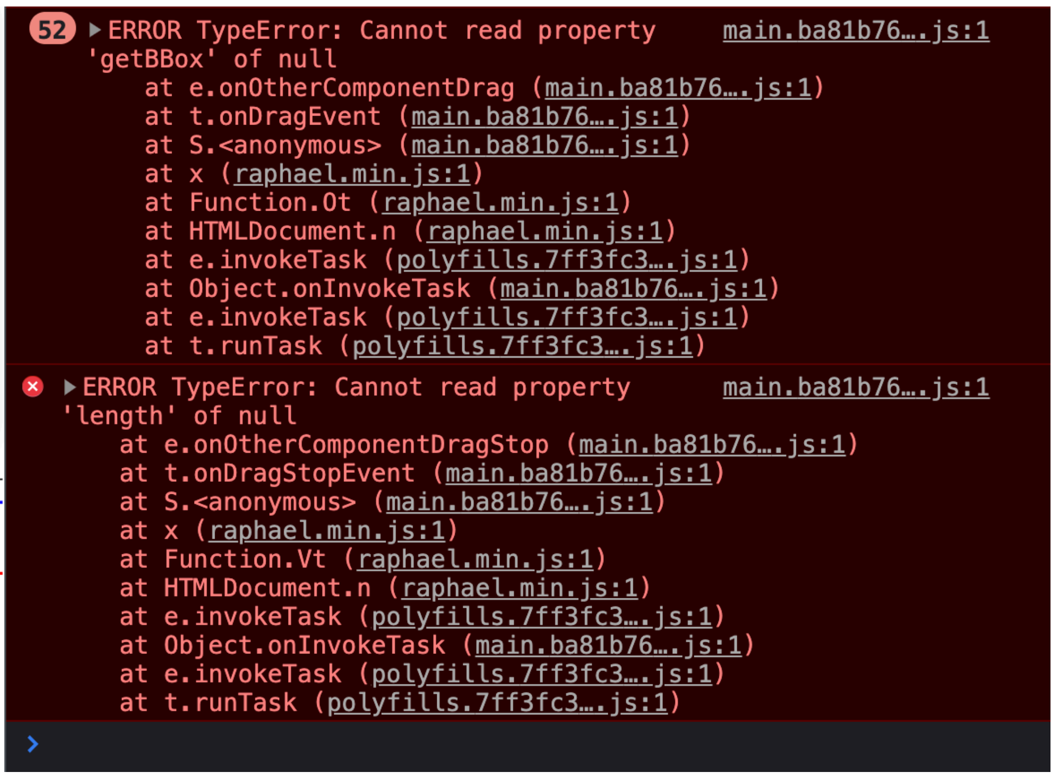
5.5. Breadboard Delete Issue

Brief: Whenever we delete a breadboard and again add a breadboard, then breadboard's drag listeners give error.

Steps to recreate:
1. Add a new breadboard
2. delete the breadboard
3. again add a new breadboard
4. now take any component from the left panel
5. start dragging the element

Expected Behaviour:
Upon dragging of other elements, console should not give error.
Tested on: develop branch & live website

OS: macOs



```
52 ▸ERROR TypeError: Cannot read property    main.ba81b76….js:1
'getBBox' of null
    at e.onOtherComponentDrag (main.ba81b76….js:1)
    at t.onDragEvent (main.ba81b76….js:1)
    at S.<anonymous> (main.ba81b76….js:1)
    at x (raphael.min.js:1)
    at Function.Ot (raphael.min.js:1)
    at HTMLDocument.n (raphael.min.js:1)
    at e.invokeTask (polyfills.7ff3fc3….js:1)
    at Object.onInvokeTask (main.ba81b76….js:1)
    at e.invokeTask (polyfills.7ff3fc3….js:1)
    at t.runTask (polyfills.7ff3fc3….js:1)
▸ERROR TypeError: Cannot read property      main.ba81b76….js:1
'length' of null
    at e.onOtherComponentDragStop (main.ba81b76….js:1)
    at t.onDragStopEvent (main.ba81b76….js:1)
    at S.<anonymous> (main.ba81b76….js:1)
    at x (raphael.min.js:1)
    at Function.Vt (raphael.min.js:1)
    at HTMLDocument.n (raphael.min.js:1)
    at e.invokeTask (polyfills.7ff3fc3….js:1)
    at Object.onInvokeTask (main.ba81b76….js:1)
    at e.invokeTask (polyfills.7ff3fc3….js:1)
    at t.runTask (polyfills.7ff3fc3….js:1)
>
```

Figure 5.10: Error in console

Error:

SOLUTION PROPOSED:
To fix this I added id attribute in dragListener which can be used to identify breadboard secondly upon deletion added a condition for breadboard to delete specific dragListener associated with the deleted element.

BUG REPORT: https://github.com/frg-fossee/eSim-Cloud/issues/309
PR: https://github.com/frg-fossee/eSim-Cloud/pull/313

5.5. Led Issue

Brief: LED component lack a check if it is connected to Arduino's Ground(GND) or not, whilst it is connected using a breadboard. This results in running of LED even when only positive terminal is connected to arduino.

Project: Arduino on Cloud

Steps to Reproduce:
1. Create any basic structure with Arduino, LED & breadboard in which LED glows.
2. Delete the wires connected to the 'Negative' terminal.
3. Run the simulation.

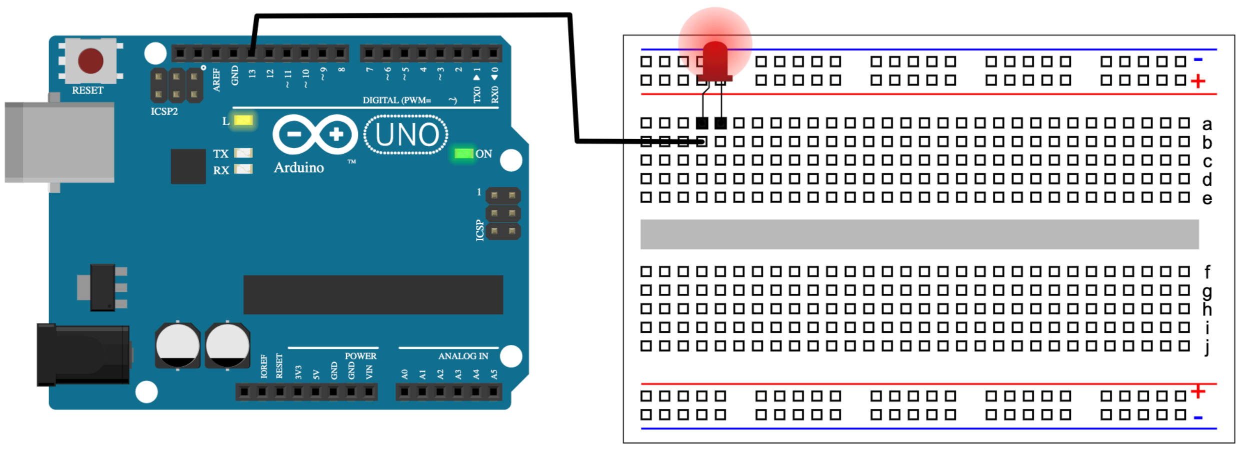


Figure 5.11: LED glows even without wire connected

4. LED will still glow. (see image below)

Expected Behaviour:
LED should not glow & prompt the user if the wire is not connected.

Tested on: Develop branch

OS Used: macOS 11

BUG REPORT: https://github.com/frg-fossee/eSim-Cloud/issues/318
PR: https://github.com/frg-fossee/eSim-Cloud/pull/319

5.6. Breadboard Pins Issue

Brief: A few circuits do not change the value of same line oriented nodes in breadboard. This happens in deeply nested circuits which need value to be transferred from one wire to another and then on another node of breadboard.

Steps to Reproduce:

1. Implement a similar circuit as given below in this Image.
2. Click on start simulation.
3. LED won't glow.

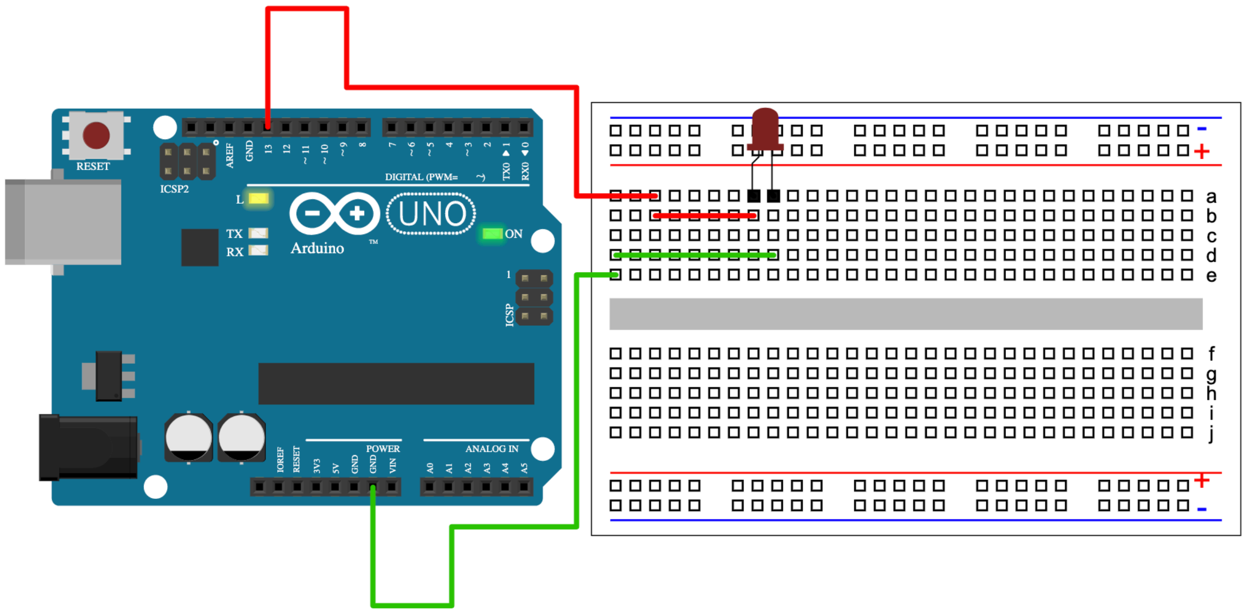


Figure 5.12: LED not glowing

Expected Behaviour: LED must glow.

REASON for problem:

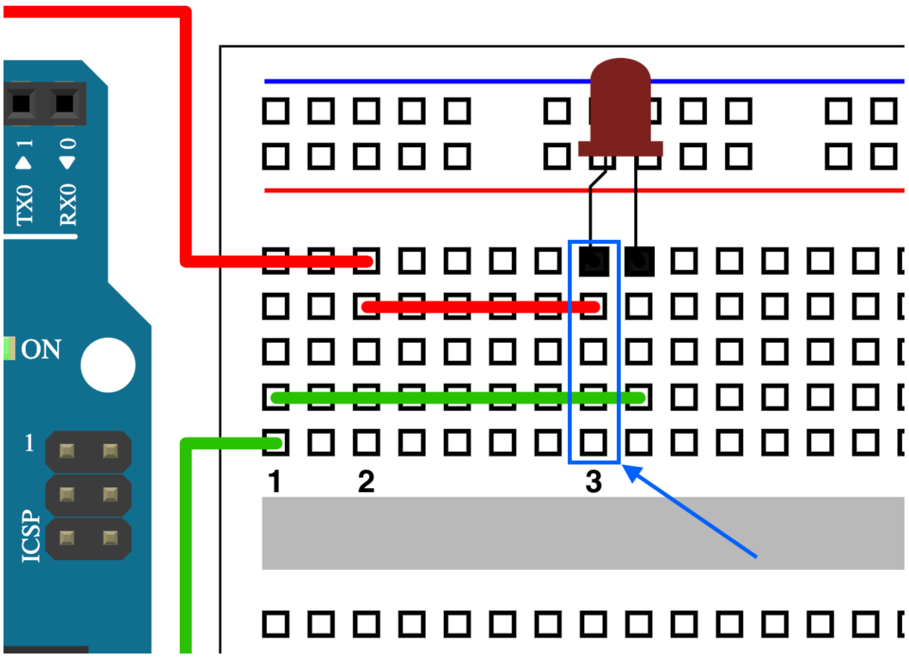


Figure 5.14: Set of points in breadboard

In above image the nodes that are marked in **blue** rectangle are not able to set value for whole set of nodes, received from nodes marked with 2 and 1.

Tested on: Live website & develop branch

Browser: Chrome 91 (macOs)

SOLUTION PROPOSED:
Modified the existing condition to return false in case of straight-line oriented nodes

BUG REPORT: https://github.com/frg-fossee/eSim-Cloud/issues/324
PR: https://github.com/frg-fossee/eSim-Cloud/pull/325

Chapter 6

Conclusion

This project was a piece of art of its own, I had a great time working on it and implementing different features. I learned many new technologies and techniques working on this project. I would like to thank my mentors for guiding me throughout the project. This project has a lot of potentials and I am sure it will be very helpful for the students and researchers working on Arduino in this era of a global pandemic.¶

Chapter 7

Future Work

This project can be improved further by adding new features and fixing already present bugs.

- Adding more sensors and components.
- Fixing bug in motor, due to which motor's rotor don't rotate in if PWM is attached.
- Adding support for more programming languages.
- Optimisation of application.
- Adding collaboration support in it, example GitHub.
- Fixing bug, user is able to delete/interact with elements event when simulation is running.

References

[1] https://esim-cloud.readthedocs.io/en/latest/overview/index.html#arduino-on-cloud

[2] https://www.analogictips.com/pulse-width-modulation-pwm/

[3] https://en.wikipedia.org/wiki/Potentiometer

[4] https://components101.com/ics/l293d-pinout-features-datasheet

[5] https://www.youtube.com/watch?v=nXmkkyw8v5A