Government Polytechnic Jalgaon

Academic Year 2020-21

Course

Embedded system

Code

22532

Of EJ 5 I

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION

GOVERNMENT POLYTECHNIC, JALGAON

(0018)

Program Name And Code : ELECTRONICS &TELECOMMUNICATION

Course Name And Code : Embedded System (22532)
Academic Year : 2020-21

Semester : Fifth.

A MICRO PROJECT

On

To study the tinkercad simulation web application. To show the demonstration of it to class via presentation.

Submitted On _____2020 by the group of 4 students.

Sr. No.	Roll No.	Name Of Student	Enrollment No.	Seat No.
1	11	Prathamesh Saraf	1800180265	
2	23	Mohit Bhangale	1800180288	
3	24	Mandar Patil	1800180290	
4	25	Mohish Khadse	1800180291	

MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION

Certificate

This Is To Certify That Master Mr/Ms. **Prathamesh , Mohit , Mohish , Mandar** Roll No.**11, 23, 24, 25** Of **5th** Semester Of Diploma In **E&TC.** Of Institute, **Government Polytechnic, Jalgaon (Code:0018)** Has Completed The **Micro Project** Satisfactorily In The Subject Embedded System (22532) For The Academic Year 2020- 2021 As Prescribed In The Curriculum.

Place**: Jalgaon** Enrollment No:-

1800180265 ,1800180288,1800180290,1800180291

Date:-

Exam. Seat No:-

Course Teacher Head Of The Department Principal

(Electronics of telecommunication department)

Seal of Institution

TITLE

To study the tinkercad simulation web application. To show the demonstration of it to class via presentation.

Submitted By-:

1. Prathamesh Saraf (11)
2. Mohit Bhangale (23)
3. Mandar Patil(24)
4. Mohish Khadase(25)

Under The Guidance Of:

Mr. K. P. Akole

INDEX

The project to introduce the students about tinkercad has been completed in 2 lectures

Lecture 1 (8-12-2020)

Content

- **Introduction of tinkercad**
- **Tinkercad community**
- **Layout**
- **Circuit design**
- **Circuit programing**

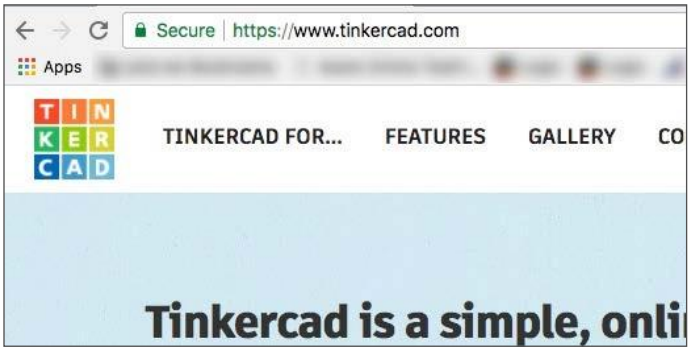
INTRODUCTION

TinkerCAD is a free online service for creating basic 3D shapes and developing digital prototypes of electronic components. These prototypes include basic circuits with LED lights, buzzers, switches, and even light sensors.

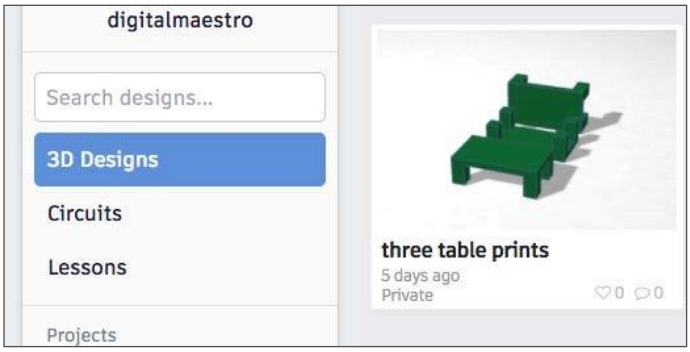
These prototypes can include a microprocessor as part of the design. Microprocessors are the simplest form of computer that can be programmed. They can be programmed to manipulate electronic components like LED lights and buzzers. Microprocessors can be programmed to gather information from sensors and interpret that information. They are used in a variety of devices all around us. They are in microwaves, refrigerators, cars, computers and many other electronic devices.

The process used in TinkerCAD is often used for rapid prototyping. Prototyping is a process where we can develop components in a flexible manner than can be quickly updated and modified to test a variety of options when developing a project or product. We will use this process of prototyping to learn how to create basic electronic circuits.

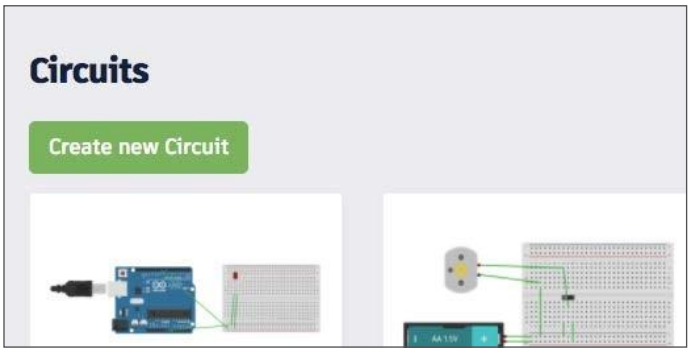
Go over to https://tinkercad.com and create a free account. Tinkercad integrates with Social Media services like Facebook. It also Integrates with services like Microsoft and Google. Students can use their district accounts to log into Tinkercad if your district uses active directory accounts with Google or Microsoft.



Go to the left side of the page after logging in and click the circuits menu option.

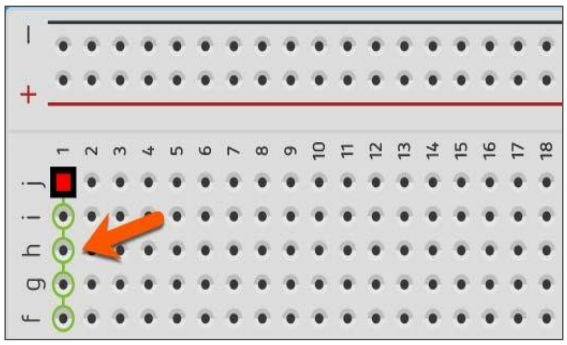


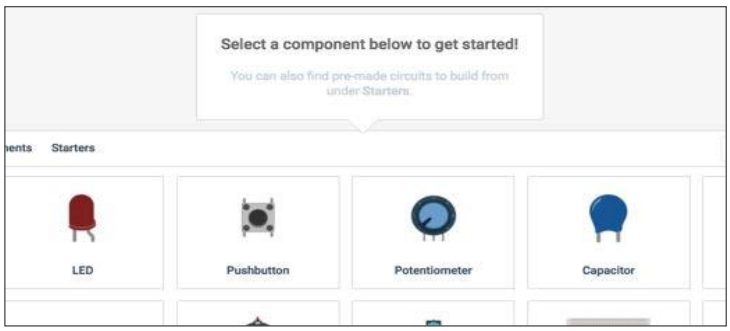
Click the create button to build a new circuit.

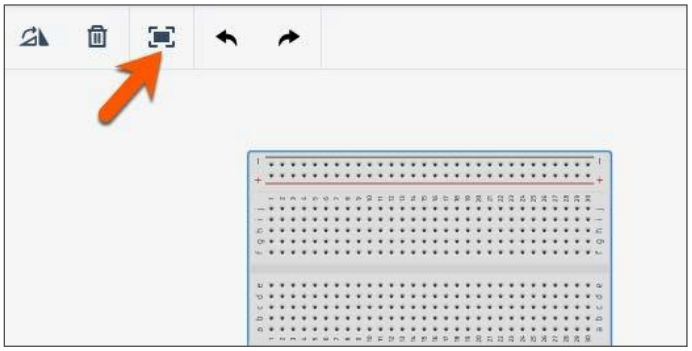


A circuit project is created and opened. A circuit project includes a variety of electronic components. Electronic components include LEDs, buttons, resistors, and a power source. The components we can use are displayed in a panel at the bottom of the page.

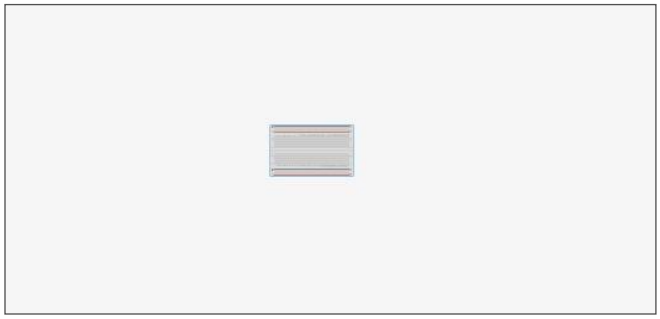
Components are commonly assembled using a Breadboard. A Breadboard is a piece of plastic that has several holes. These holes are used to hold different components. Find the Breadboard component and click on it once.





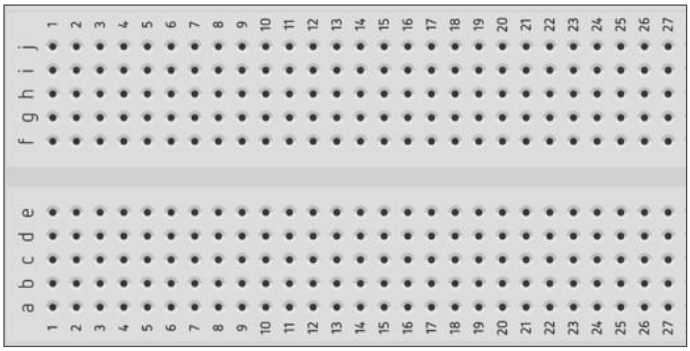


The Breadboard will be selected and temporarily attached to the mouse pointer. Move the mouse pointer onto the workspace and click the mouse button to place the Breadboard onto the workspace.



The Breadboard might appear too small or too large in the work area. Click on the zoom to fit button so the Breadboard is centered and magnified.

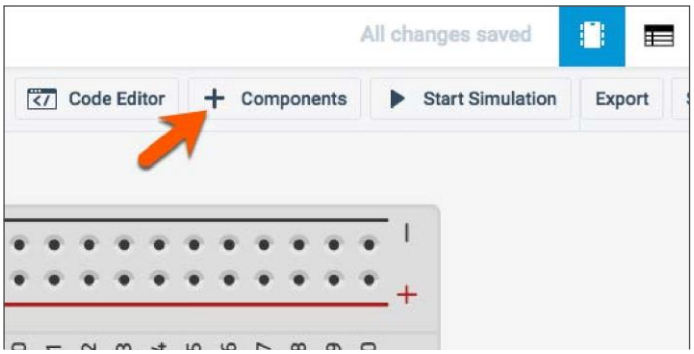
The Breadboard has a grid of thirty by ten holes in the main area. The rows are numbered 1 to 30 and the columns are labeled with the letters A through J. The columns A through E are separated from columns F through J by a piece of plastic.



The edges of the board contain two columns with the same number of rows. These columns have negative and positive symbols. These columns and holes are used for the power supply. The components in the center part of the board will tap into these columns to draw electric current.

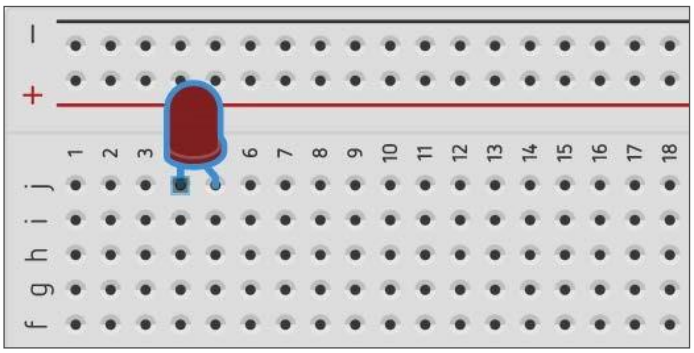
Closed Circuit with an LED

Our first project will create a basic closed circuit with a light. Our light will be supplied by a Light Emitting Diode, LED. Click the Components button to open the components drawer if it isn't already open.

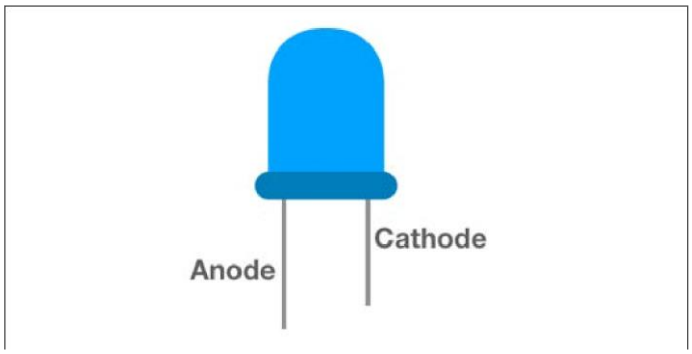


Click the LED component. It will attach itself to the mouse pointer.

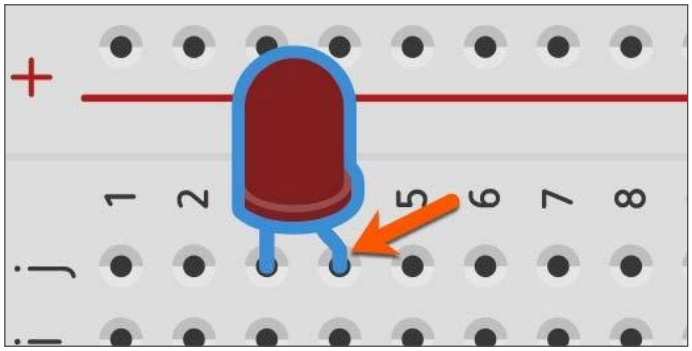
Place the LED onto the board so that each wire of the LED is in a hole. The wire coming from a component is often called a lead. It is pronounced like the word lead in **leader** and not the soft metal lead.



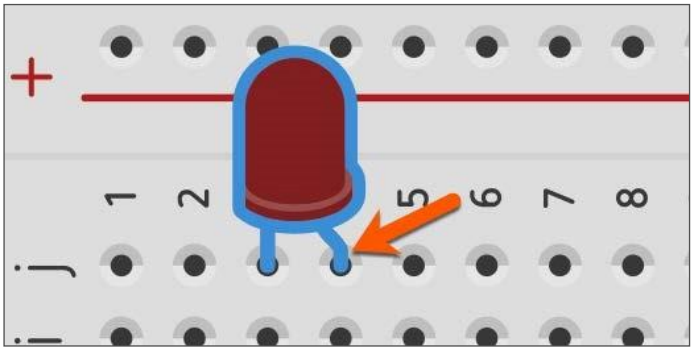
An LED has connections that make it different from a regular light bulb. An LED has one connection called a Cathode and another called an Anode. The Anode lead is usually longer than the cathode. This distinction is important because the anode must be connected to the positive end of an electric circuit. Current flows in only one direction through an LED.



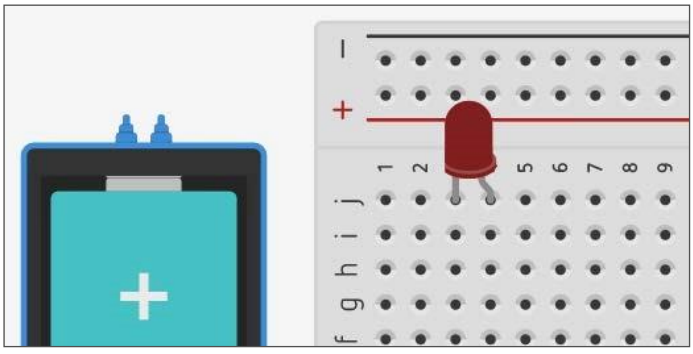
The LED anode in our circuit is identified by a bent lead. This is where the positive current must connect.



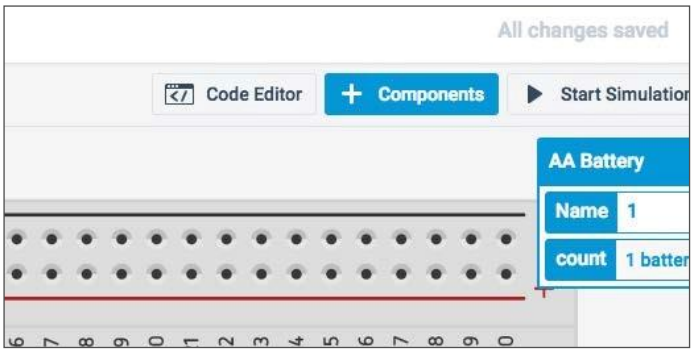
We need a power source to power the LED. Open the components panel and find a battery. We will use the AA battery with 1.5 volts for our first circuit.



Place the battery along the left side of the Breadboard.

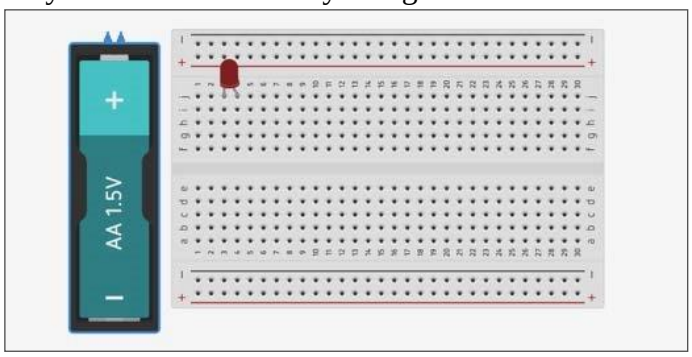


Click the active components button to close the components panel.

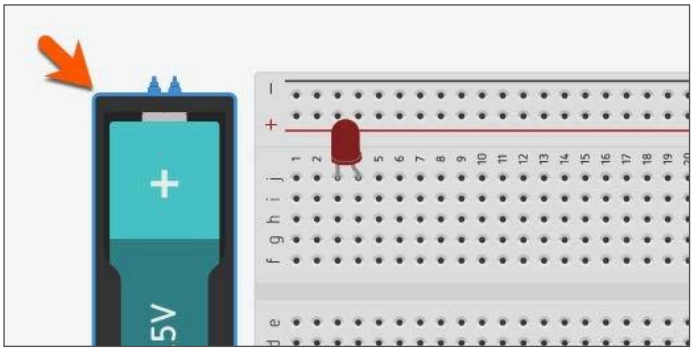


We can zoom in and out of our project using the scroll wheel on our mouse or touch gestures on a track pad, but that can be difficult. It might be easier to use these short cut key combinations. If you are on a Windows or Chromebook computer, hold the Control key and press the Plus key to zoom in or the minus key to zoom out. On a Mac, hold the Command key and use the Plus to zoom in or the minus to zoom out. *Tap the plus or minus key a couple of times.*

Zoom out of the project so you can see the battery alongside the Breadboard.

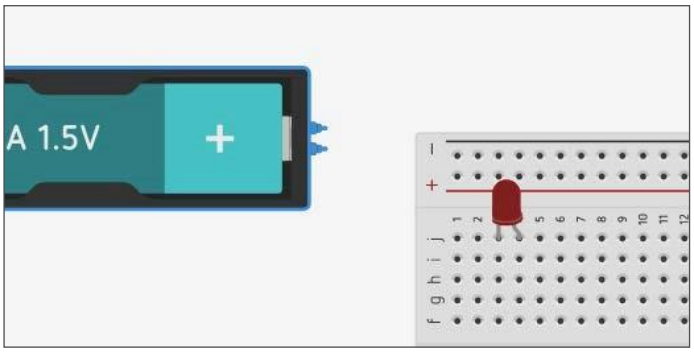


The terminals for the battery are at right angles to the Breadboard connectors. Battery terminals are the negative and positive connections. The terminals in this example, are pointing up.
I find it useful for the terminals to be directly across from the connectors. We can rotate the battery so the connections are easier. Click the battery once to make sure it is selected. A blue border appears around selected components.



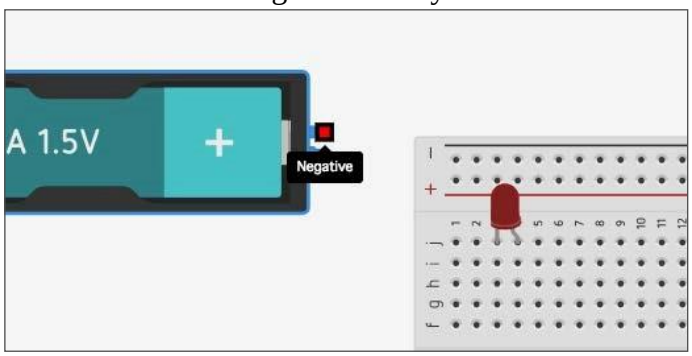
Click the rotate button in the button bar. This will rotate the component clockwise in small increments of about 30 degrees each. Click the button three times to rotate the battery 90-degrees clockwise.

The terminals should be facing the Breadboard.

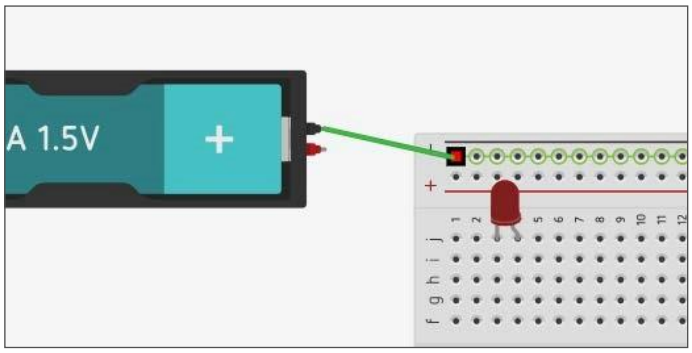


We need to connect the battery to the Breadboard so it supplies voltage to our LED. We connect components to one another when they are not on the same row with Lead wires or Jumper wires. These are wires coated with a plastic and exposed end that can be used to jump from one component to another. This is one reason they are called jumper wires.

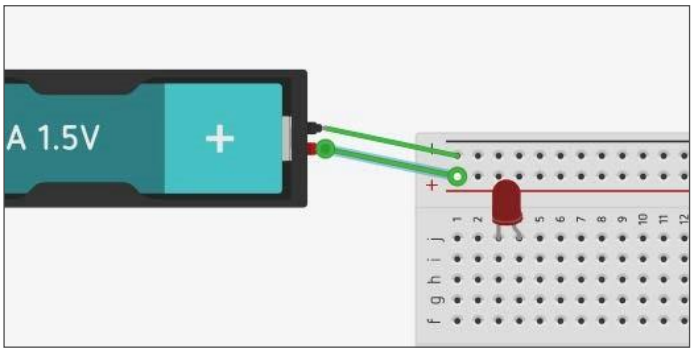
Move your mouse pointer over the top terminal. A square will appear over the terminal and a label will appear identifying the terminal as the negative battery terminal.



Click once on the terminal and move the mouse pointer to the first hole in the negative column. Click once on the hole to complete the connection. This is how we create jumper wire connections. All the other holes in the negative column are identified with a green circle. This means that all these holes are joined and can be used to connect the negative end of wires or components.



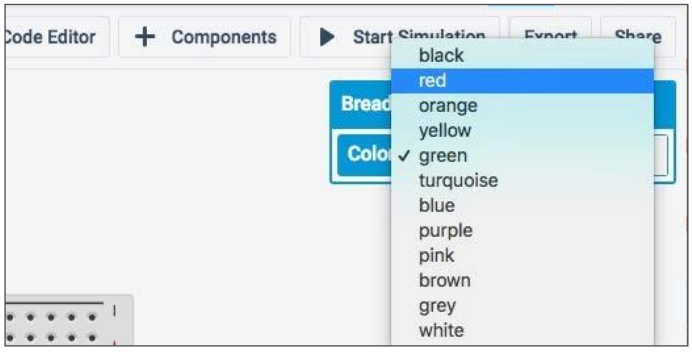
Repeat the process with the positive terminal and place the jumper wire onto the positive hole on the breadboard. When a connection is completed the ends of the wire will be identified with circles.



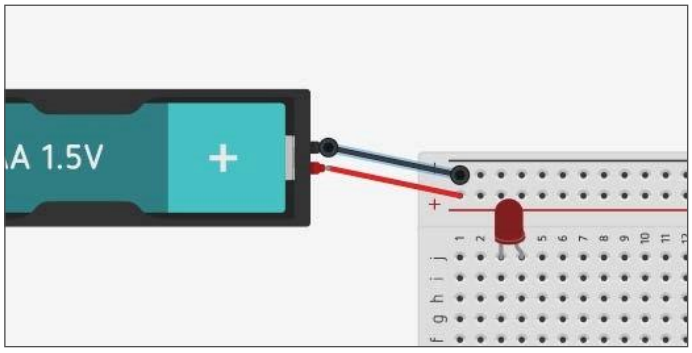
This is a simple circuit with only a few wires but circuits can get complicated very quickly with many wires jumping from one location to another. It is usually a good idea to identify these wires with colors. There are some standard colors that are used in electronics. Red is usually used for positive connections and black for negative connections in DC circuits. Sometimes green is used.

Make sure the positive terminal wire is still selected. A Breadboard wire configuration panel is located on the right side of the page. Click the color selector.

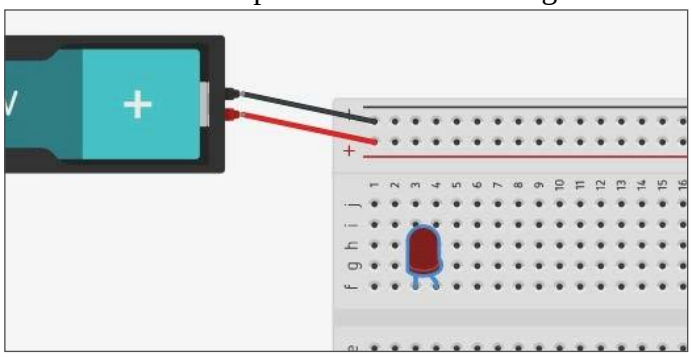
Select the red color.



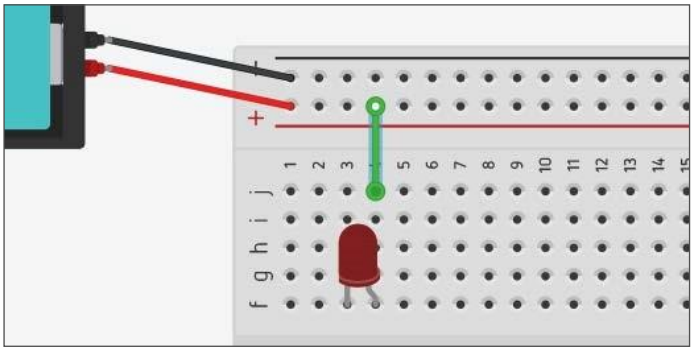
Repeat the process for the negative terminal wire and change the wire color to black.



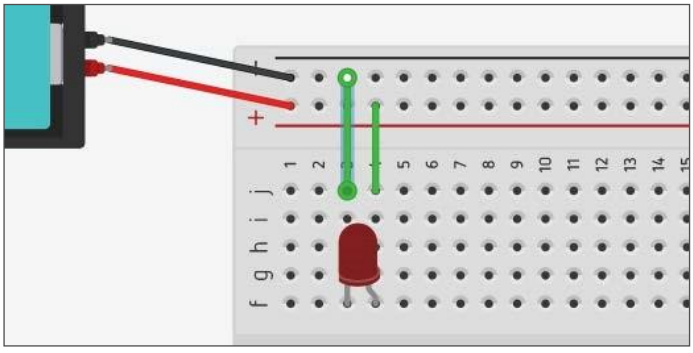
One of the benefits of using a Breadboard is that components can be easily moved around to form new connections or to make room for other components. Click and drag the LED to column f.



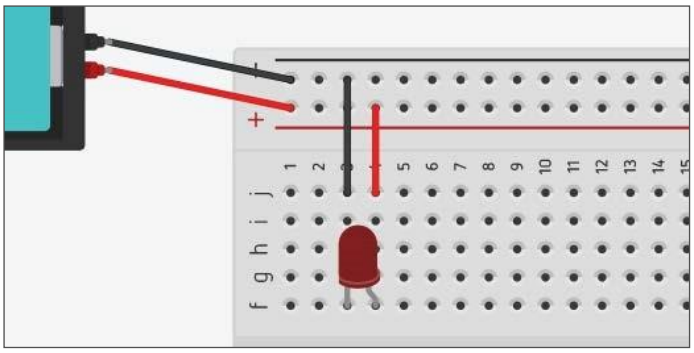
The LED will light up when we have a closed circuit. This circuit isn't closed yet. We need two wires to close the circuit. One wire for the positive and one for negative. The positive current must flow through the anode in our LED. Click on Row 4 Column J to begin a jumper wire. Connect the other end of the wire to the positive column.



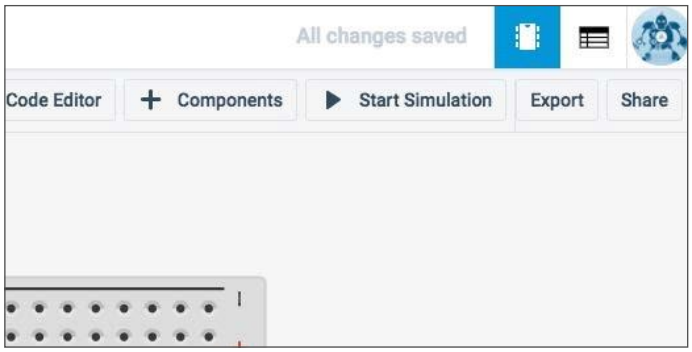
Connect another wire from Row 3 Column J to the negative terminal column.



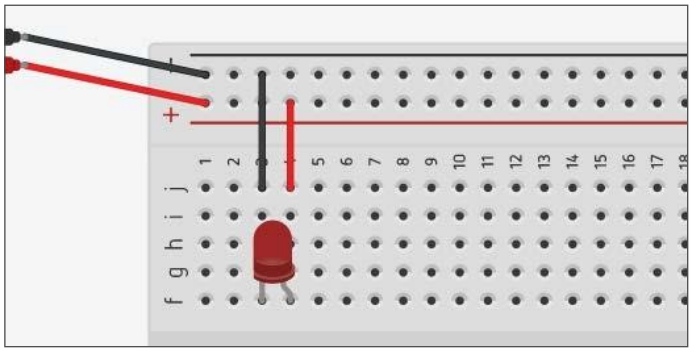
Our circuit is now complete. We should change the color or our wires so they match the polarities.



In the simulation, nothing happens until we run the simulation. Click the Start Simulation button.

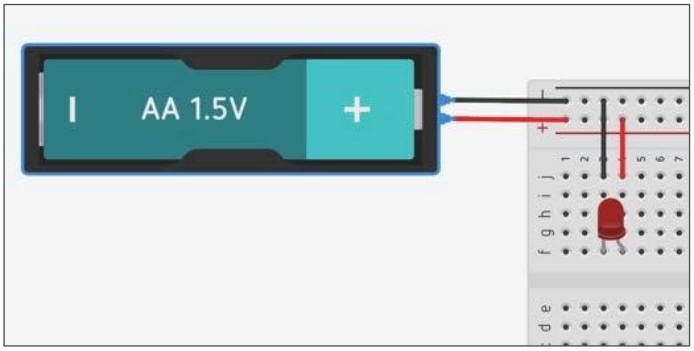


The LED on the Breadboard will change to a lighter color to simulate that the LED has turned on.

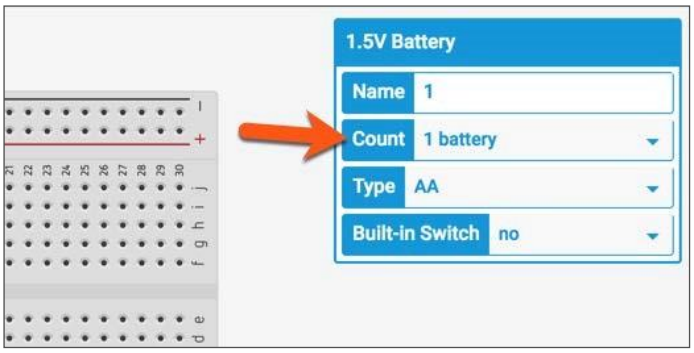


The LED doesn't appear all that bright. That isn't because of the program but because of the voltage being supplied to the LED. The single battery we used in the circuit is supplying a low voltage. In this example, we want a low voltage because LEDs have a limit. Too much voltage and the LED will blow out and stop working. This is where a simulator is very useful. In a real-world application, we wouldn't want to burn out our LED.

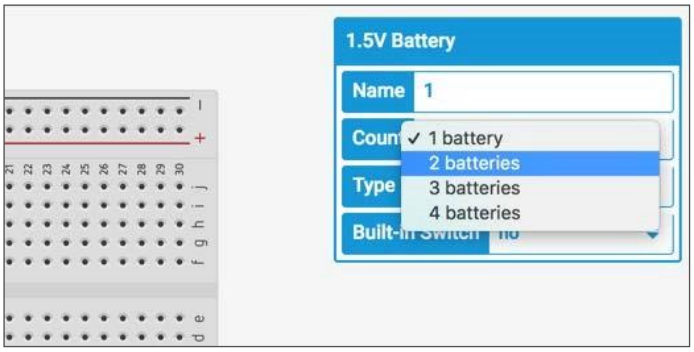
Click the battery once.



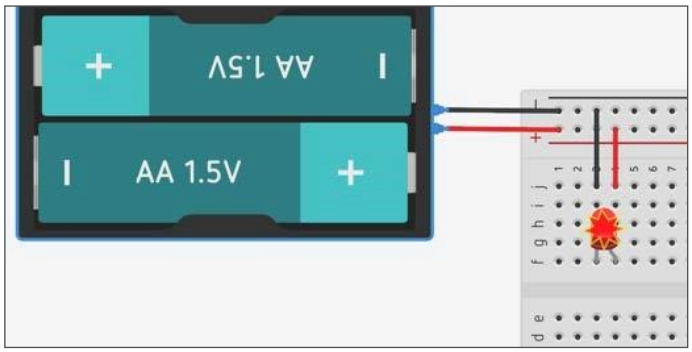
Go to the battery configuration panel and click the battery count selector.



Select two batteries.



Start the simulation and observe what happens to the LED. The star over the LED indicates that the LED blew out. In the physical world, this would mean that our LED is no longer good and needs to be replaced. LEDs are not very expensive but if you start blowing out several at a time it can get expensive. In a future lesson, we will learn how to use resistors so the LED doesn't get damaged.



Click the battery component and return the number of batteries to one.\

Circuits and Resistors

In the previous lessons, we created a circuit that lit an LED. In one of the lessons we added two batteries to the circuit and the LED was damaged. The LED was damaged because it received too much voltage. LEDs are low voltage or low wattage components and don't need much electricity to get them to emit light. This is one of the reasons they are so useful for lighting homes, businesses, and even cities. Their low power consumption makes them ideal when conserving energy. Power sources like batteries are designed to produce large voltages and generate current to power energy hungry components. In the past, one of these energy hungry components was the Incandescent Light Bulb, which requires more voltage than an LED.

When constructing circuits, we have a variety of components that require different voltages and current. We will learn how resistors are used to restrict the flow of current to electrical components. Electrical current is measured in Amps. In this lesson, we will learn about Ohm's Law and how we use it to understand what is going on when we use resistors in a circuit.

Ohm's Law states that we can calculate the amount of voltage in a circuit by multiplying the electrical current by the resistance in the circuit. In the formula, we use the letter "I" for Current. The letter "I" is used in honor of the man who formulated Ampère's force law by which we measure current and made electric motors possible from electro magnets. André-Marie Ampère, who was French, used it in a French phrase, "intensité de courant, (current intensity)". So, the letter "I" actually stands for Intensity.

$$\text{Voltage} = \text{Current} \times \text{Resistance}$$

$$V = I \times R$$

The formula can be manipulated to find the current in a circuit. To find the current in a circuit we divide the Voltage by the Resistance.

$$\text{Current} = \frac{\text{Voltage}}{\text{Resistance}}$$

$$I = V / R$$

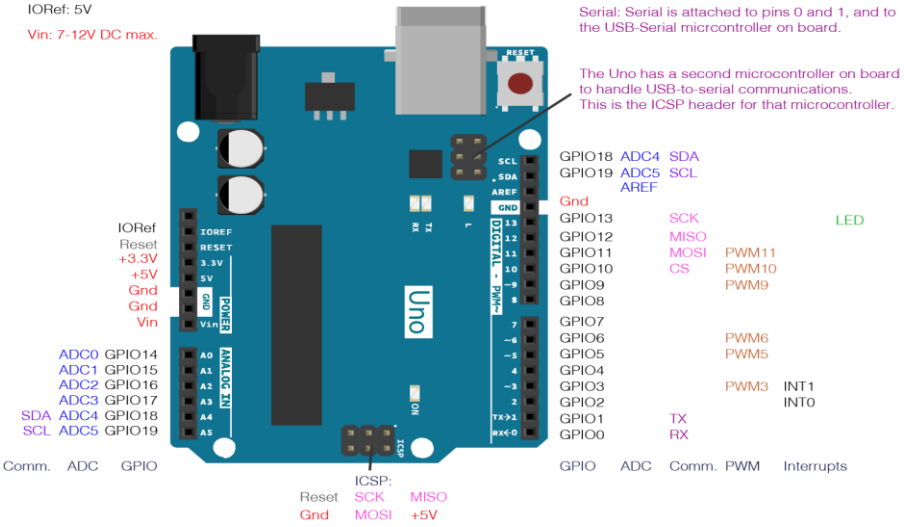
We can use the formula to find the resistance in a circuit. To solve for resistance, we divide voltage by the current.

Teaching: Student should work out how we arrived as these formulas. It is a good idea to introduce or revisit formulas and how we can manipulate them to solve for different solutions.

$$\text{Resistance} = \frac{\text{Voltage}}{\text{Current}}$$

$$R = V / I$$

Introduction to Microcontroller (Arduino uno)



Arduino is an open-source hardware and software company, project and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices. Its hardware products are licensed under a CC-BY-SA license, while software is licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially from the official website or through authorized distributors.

Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards ('shields') or breadboards (for prototyping) and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs from personal computers. The microcontrollers can be programmed using the C and C++ programming languages, using a standard API which is also known as the "Arduino language". In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) and a command line tool (arduino-cli) developed in Go. The Arduino project started in 2005 as a tool for students at the Interaction Design Institute Ivrea in Ivrea, Italy, aiming to provide a low-cost and easy way for novices and professionals to create devices that interact with their environment using sensors and actuators. Common examples of such devices intended for beginner hobbyists include simple robots, thermostats and motion detectors.

Concepts coding for Arduino

`digitalWrite()`

Write a `HIGH` or a `LOW` value to a digital pin.If the pin has been configured as an `OUTPUT` with `pinMode()`, its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for `HIGH`, 0V (ground) for `LOW`.

If the pin is configured as an `INPUT`, `digitalWrite()` will enable (`HIGH`) or disable (`LOW`) the internal pullup on the input pin. It is recommended to set the `pinMode()` to `INPUT_PULLUP` to enable the internal pull-up resistor. See the Digital Pins tutorial for more information.If you do not set the `pinMode()` to `OUTPUT`, and connect an LED to a pin, when

calling digitalWrite(HIGH), the LED may appear dim. Without explicitly setting pinMode(), digitalWrite() will have enabled the internal pull-up resistor, which acts like a large current-limiting resistor.

Syntax

digitalWrite(pin, value)

d i g i t a l R e a d ( ) Reads the value from a specified digital pin, either HIGH or LOW.

Syntax

digitalRead(pin)

i f

[Control Structure]

The if statement checks for a condition and executes the proceeding statement or set of statements if the condition is 'true'.

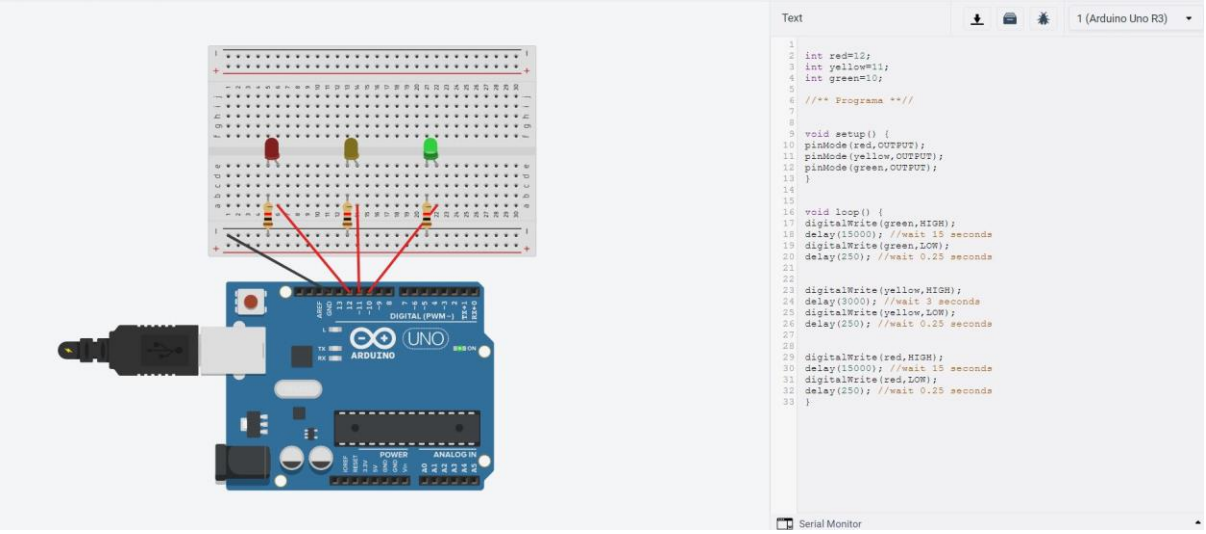
Syntax
if (condition) {

Serial communication

Serial communication on pins TX/RX uses TTL logic levels (5V or 3.3V depending on the board). Don't connect these pins directly to an RS232 serial port; they operate at +/- 12V and can damage your Arduino board.
Serial is used for communication between the Arduino board and a computer or other devices. All Arduino boards have at least one serial port (also known as a UART or USART): **Serial**. It communicates on digital pins 0 (RX) and 1 (TX) as well as with the computer via USB. Thus, if you use these functions, you cannot also use pins 0 and 1 for digital input or output.

You can use the Arduino environment's built-in serial monitor to communicate with an Arduino board. Click the serial monitor button in the toolbar and select the same baud rate used in the call to begin().

Traffic light with 3 LED Circuit



Components required for the ckt

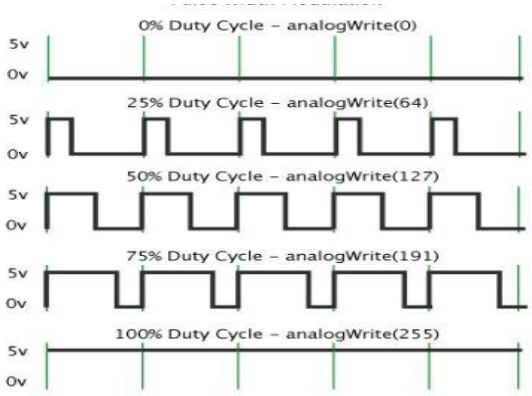
Name	Quantity	Component
R1, R2, R3	3	1 kâ„¦ Resistor
D1	1	Red LED
D2	1	Green LED
D3	1	Yellow LED
U1	1	Arduino Uno R3

Working principal

It is a simole Arduino circuit which shows the traffic light red green and yellow as per set time in the program . As per the programm green light remains on for the 15sec and the turns yellow light for 3sec seconds and after red turns on for 15 sec

Analog read and analog write (input and output) in Arduino

Pulse Width Modulation, or PWM, is a technique for getting analog results with digital means. Digital control is used to create a square wave, a signal switched between on and off. This on-off pattern can simulate voltages in between full on (5 Volts) and off (0 Volts) by changing the portion of the time the signal spends on versus the time that the signal spends off. The duration of "on time" is called the pulse width. To get varying analog values, you change or modulate, that pulse width. If you repeat this on-off pattern fast enough with an LED, for example, the result is as if the signal is a steady voltage between 0 and 5v controlling the brightness of the LED.



In the graphic below, the green lines represent a regular time period. This duration or period is the inverse of the PWM frequency. In other words, with Arduino's PWM frequency at about 500Hz, the green lines would measure 2 milliseconds each. A call to analogWrite() is on a scale of 0 – 255, such that analogWrite(255) requests a 100% duty cycle (always on), and analogWrite(127) is a 50% duty cycle (on half the time) for example.

Analog read

Syntax

analogRead(pin)

Reads the value from the specified analog pin. Arduino boards contain a multichannel, 10-bit analog to digital converter. This means that it will map input voltages between 0 and the operating voltage(5V or 3.3V) into integer values between 0 and 1023. On an Arduino UNO, for example, this yields a resolution between readings of: 5 volts / 1024 units or, 0.0049 volts (4.9 mV) per unit. See the table below for the usable pins, operating voltage and maximum resolution for some Arduino boards. The input range can be changed using analogReference(), while the resolution can be changed (only for Zero, Due and MKR boards) using analogReadResolution().On ATmega based boards (UNO, Nano, Mini, Mega), it takes about 100 microseconds (0.0001 s) to read an analog input, so the maximum reading rate is about 10,000 times a second.

For arduion uno board there are total 5 analog read and write pins
A0 to A5 which has maximum resolution 10 bits

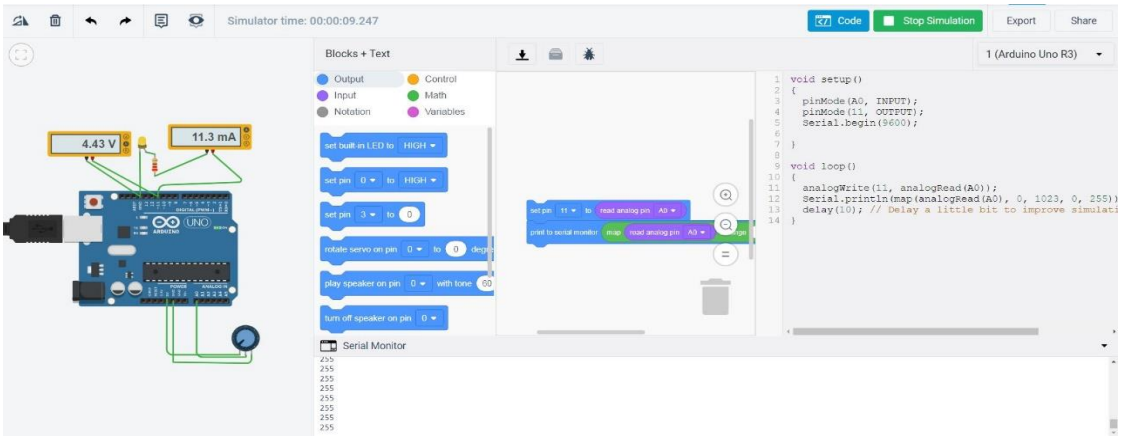
Analog write

Syntax

analogWrite(pin, value)

Writes an analog value (PWM wave) to a pin. Can be used to light a LED at varying brightnesses or drive a motor at various speeds. After a call to analogWrite(), the pin will generate a steady rectangular wave of the specified duty cycle until the next call to analogWrite() (or a call to digitalRead() or digitalWrite()) on the same pin.
To show the proper working of the analog read and write here are some circuits which will elaborate the concept .

Led Brightness controller using pot meter on Arduino board



Circuit components list

In the a above given image there is a circuit program code and a block code are given
As we move the pot meter to right side that increasing its resistance the brightness of the led reduces because the duty cycle given to the analog pin **A0** also reduces which can be seen in the serial monitor

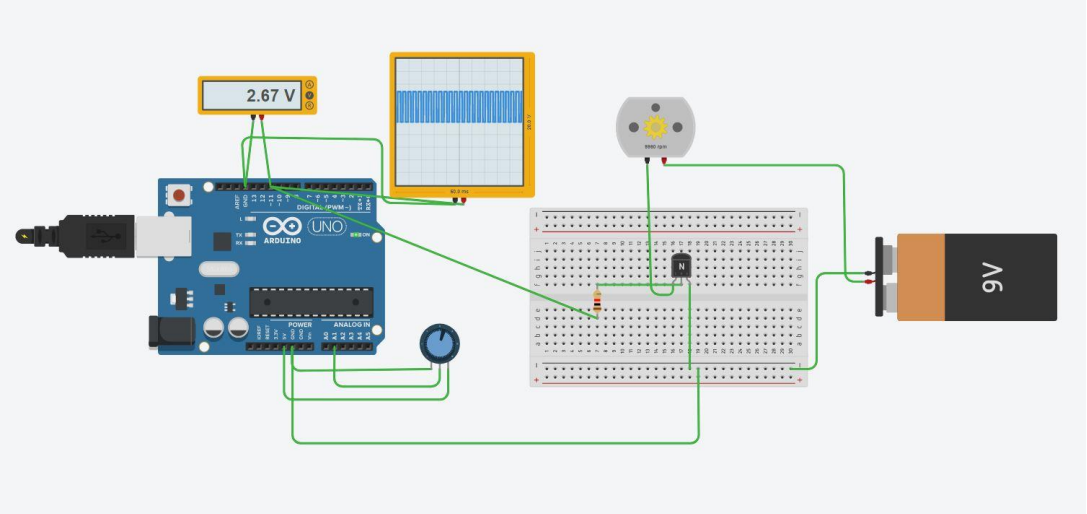
Working principal

As we move the pot meter to right side that increasing its resistance the brightness of the led reduces because the duty cycle given to the analog pin **A0** also reduces which can be seen in the serial monitor

The analogRead() command converts the input voltage range, 0 to 1023, to 0 to 5 voltage. After the reading is successful we are able to control the brightness of the LED depending on the value that is read. An interesting part of this sketch is the way we convert the reading to a digital number for controlling the LED.

First, we need to consider that PWM uses a range of values between 0 and 255 while analogRead() returns a value between 0 and 1023. Therefore we will divide the value of analogRead() by 4. By doing this we are creating values that work with PWM. If you like, you can change the number 4 to 8 to see the effect.

Motor speed control circuit using analog port of Arduino




```
void setup()
{
  pinMode(A1, INPUT);
  pinMode(11, OUTPUT);
}

void loop()
{
  analogWrite(11, map(analogRead(A1), 0, 1023, 0, 255)
  delay(10); // Delay a little bit to improve simulati
}
```

Circuit components list

Name	Quantity	Component
U1	1	Arduino Uno R3
T1	1	NPN Transistor (BJT)
M1	1	DC Motor
BAT1	1	9V Battery
Rpot1	1	250 kâ„¦ Potentiometer
Meter1	1	Voltage Multimeter
R1	1	1 kâ„¦ Resistor
U2	1	5 ms Oscilloscope

Working Principal

Working is same as the led brightness circuit . Increasing pot resistance leads to reduce in the duty cycle of the PWM pin 11whin in terns reduce the speed of the motor

Lecture 2 (21-12-2020)

Content

- **Introduction of scratch .**
- **Layout .**
- **Circuit design with scratch programming .**
- **Introduction of micro bit .**
- **Circuit design using micro-bit with scretch progeamming .**

Introduction of scratch

Our newest feature in Tinkercad Circuits is visual code blocks–powered by Scratch Blocks!

Scratch programming

Scratch is a block-based visual programming language . Users of the site can create online projects using a block-like interface .
Code blocks let you program Arduinos using drag-and-drop blocks. The blocks allow you to automatically build the text-based code in real time, so you can see exactly how Arduino code is formatted and then easily export your sketch to upload directly to an Arduino board.

scratch is a block-based visual programming language and website targeted primarily at children 8-16 to help learn code. Users of the site can create projects on the web using a block-like interface. Wikipedia

- Influenced: ScratchJr
- Implementation language: Squeak (Scratch 0.x, 1.x); ActionScript (Scratch 2.0); JavaScript (Scratch 3.0)
- License: GPLv2 and Scratch Source Code License
- Paradigm: Event-driven, visual, block-based programming language
- OS: Microsoft Windows, macOS, Linux (via renderer), HTML5

Advantages of using scratch programming -:

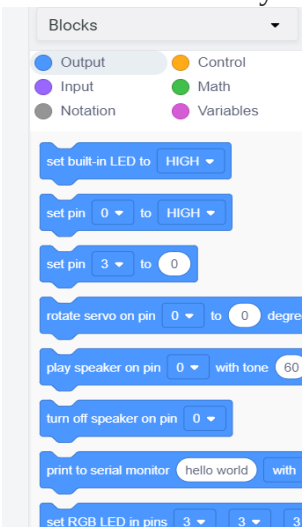
1. Simplify the Arduino programming experience to reduce common syntax errors (such as mistyping the name of a function or forgetting a semi-colon)
2. Focus on the desired interaction (what you want to sense and control), while scaffolding all the necessary setup required to do so
3. Bridge the worlds of visual programming and text-based programming to make everything created in Tinkercad Circuits extensible to real, physical hardware using the Arduino IDE

Tikercad scratch code layout

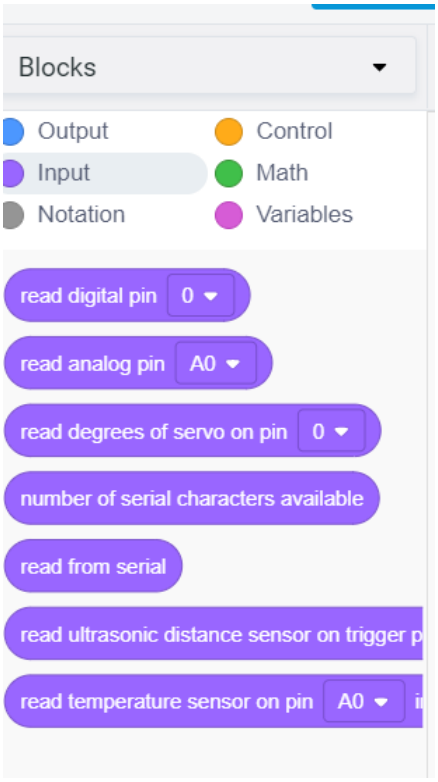
Tinkercad code blocks are visual blocks you can drag-and-drop to create Arduino programs. Using the Tinkercad Circuits simulator, you can test any code you create directly in the browser, before you build and program your devices with real physical components.

Code blocks are organized into several categories:

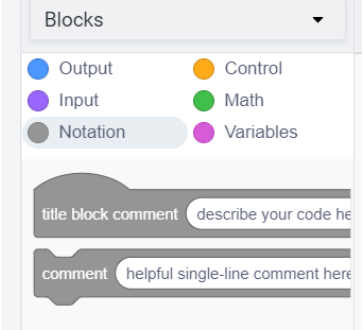
- **Output** — Blocks for controlling actuators connected to your programmable microcontroller .



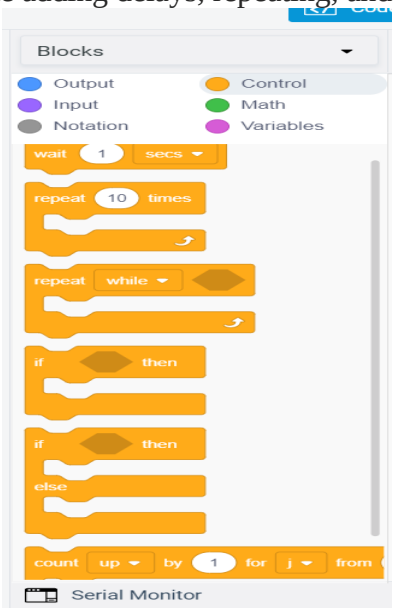
- **Input —** Blocks for reading sensor input .



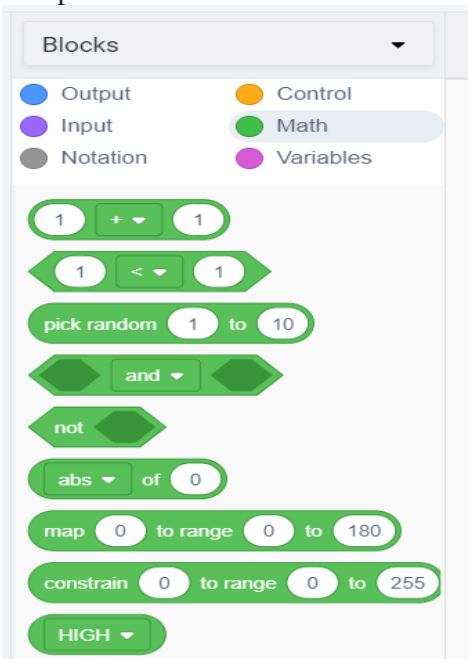
- **Notation** — Blocks for adding comments, both to provide an overview of your code (in the title block) and in-line with your code.



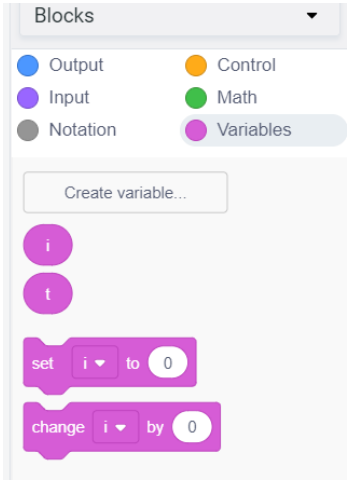
- **Control** — Control structures like adding delays, repeating, and if else statements .



- **Math —** Blocks for logic and computation

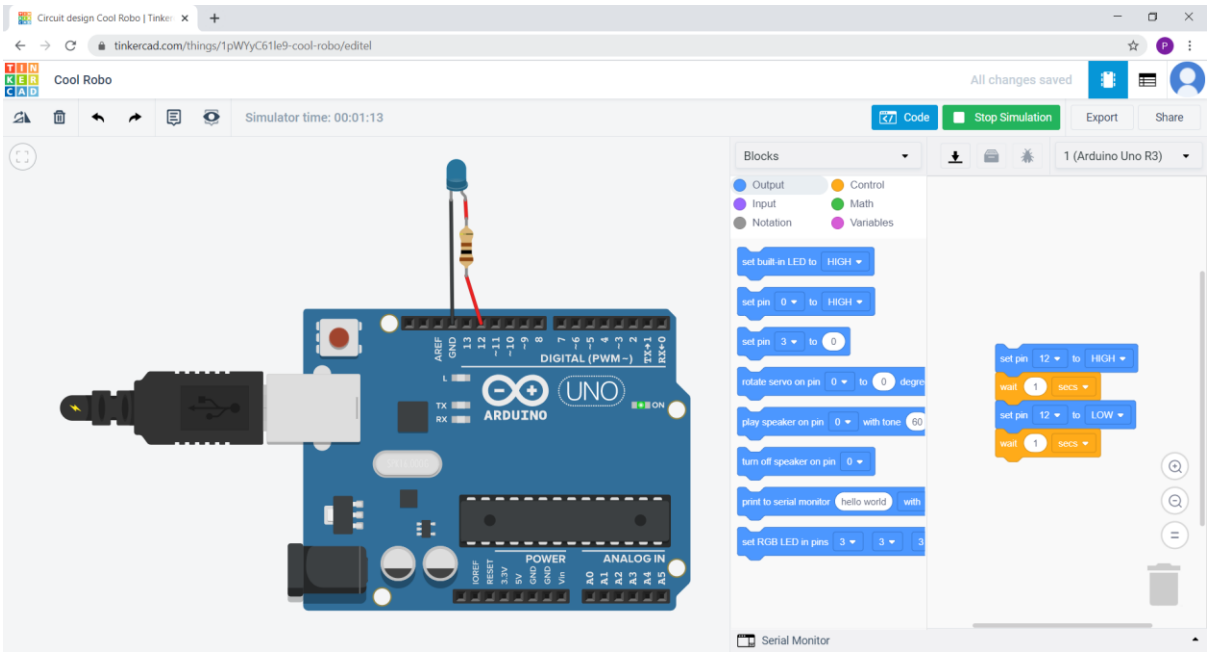


Variables — Custom variables that can be edited in your program



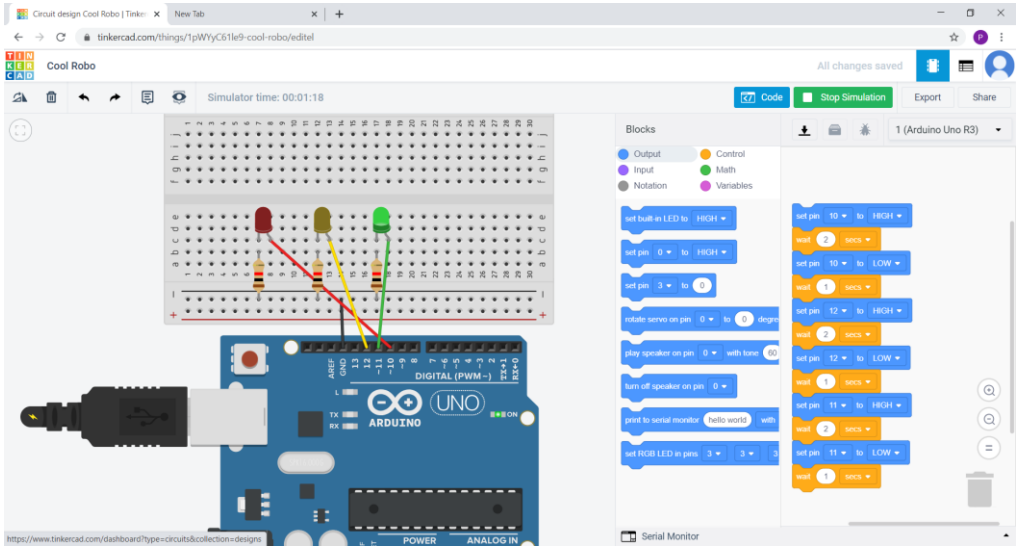
Circuit design with scratch programming .

1. LED blinking circuit with 1 sec delay .



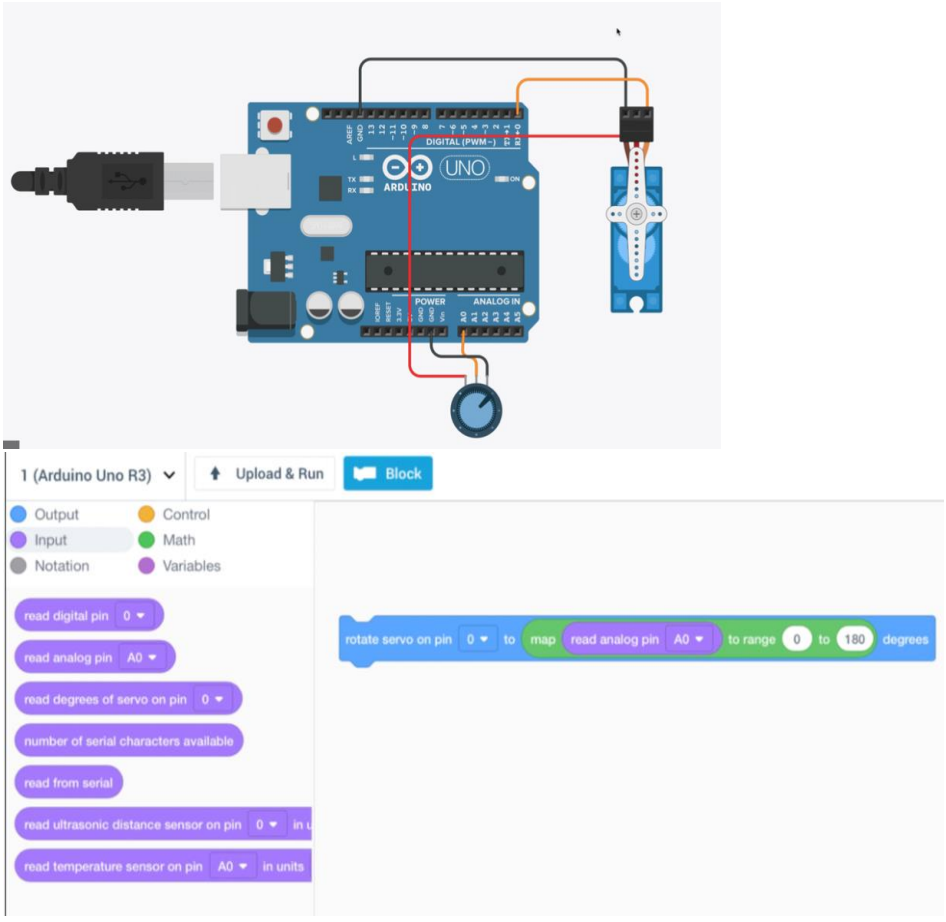
In the program above, you can see that we have placed an Arduino block and a forever block. These two blocks are musts for programming an Arduino. The point of using a forever block is, in an Arduino program, the logic should be such that it would run in a loop indefinitely. In our case, we need to blink the LED again and again, so use of a forever block is a must in many cases, and it makes life a lot easier when programming Arduino. Inside forever, set which digital pin block is used. This block can make a pin voltage high or low. So, if I have an LED connected to pin number 13 of the Arduino (below) and I want to turn it on, I will use “set digital pin 13 output as HIGH”, and my LED will light up.

2. Traffic light circuit .



In this circuit 3 LED different color (red, green, yellow) ,3 resistance (1K ohm), Arduino, breadboard .then set the programe for traffic control . set the 0.25 sec delay every light the program successfully run.then click the simulate.programing this circuit given below .

3. Servo moter with 10sec delay .



All Arduino programs have two main functions: a setup() and a loop(). When you drag out any block into the scripting area, the code gets added to the **loop()** function, and we **dynamically create any lines of code necessary within the setup()**.

For example, when the first block to rotate the servo is added (*rotate servo on pin 0 to 0 degrees*), several things happen **automatically**:

1. The Servo library is added to the code *(#include <Servo.h>)*
2. A Servo instance is created (*Servo servo_3*)
3. In the setup(), the servo is attached to the pin specified *(servo_3.attach(0))*
4. We tell the servo to move to 0 degrees (*servo_3.write(0)*)

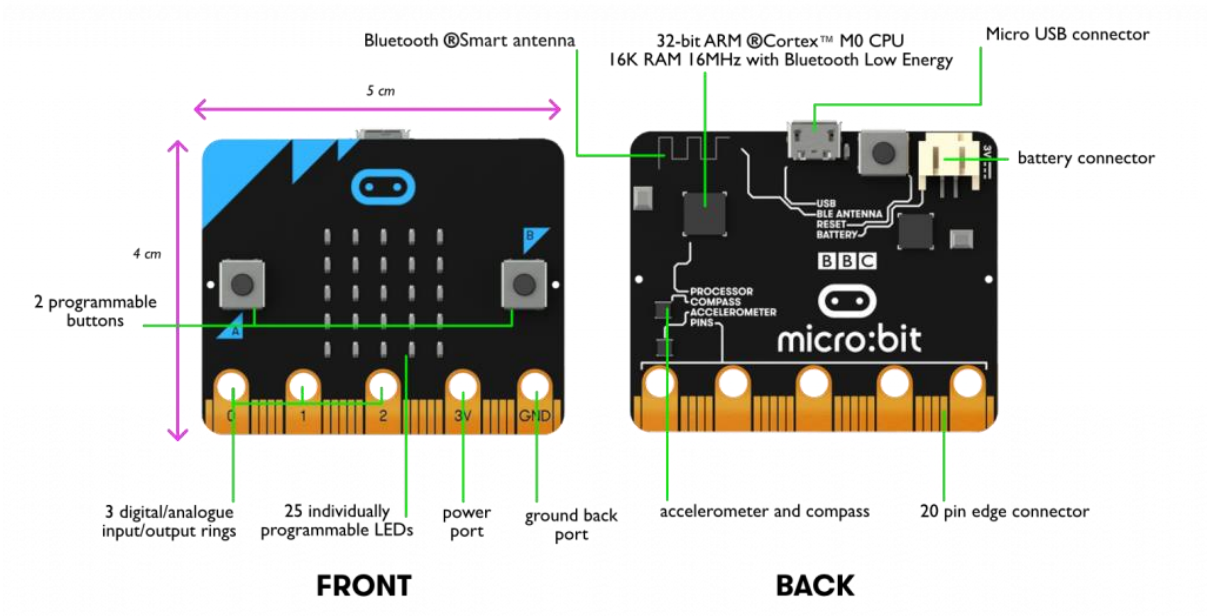
Introduction of micro bit .

The Micro Bit (also referred to as BBC Micro Bit, stylized as micro:bit) is an open source hardware ARM-based embedded system designed by the BBC for use in computer education in the United Kingdom. It was first announced on the launch of BBC's Make It Digital campaign on 12 March 2015 with the intent of delivering 1 million devices to pupils in the UK. The final device design and features were unveiled on 6 July 2015 whereas actual delivery of devices, after some delay, began in February 2016 .

BBC Micro Bit with its original packaging behind it.

Developer	BBC Learning, BBC R&D, ARM Holdings, Barclays, element14, NXP Semiconductors, Lancaster University, Microsoft, Samsung, Nordic Semiconductor, ScienceScope, Technology Will Save Us, Python Software Foundation
Type	Single-board microcontroller
Release date	Schools: September 2015 (delayed) Public: October 2015 (delayed) First deliveries to teachers reported February 10th 2016.
CPU	Nordic Semiconductor nRF51822, 16 MHz ARM Cortex-M0 microcontroller, 256 KB Flash, 16 KB RAM.
Connectivity	Bluetooth LE, MicroUSB, edge connector
Website	microbit.org

BBC micro:bit for all sorts of cool creations, from robots to musical instruments – the possibilities are endless. This little device has an awful lot of features, like 25 red LED lights that can flash messages. There are two programmable buttons that can be used to control games or pause and skip songs on a playlist. BBC micro:bit can detect motion and tell you which direction you're heading in, and it can use a low energy Bluetooth connection to interact with other devices and the Internet.



Moter Circuit design using micro-bit with scretch progeamming .