SCIENCE NORTH SCIENCE NORD
SUDBURY, ONTARIO, CANADA

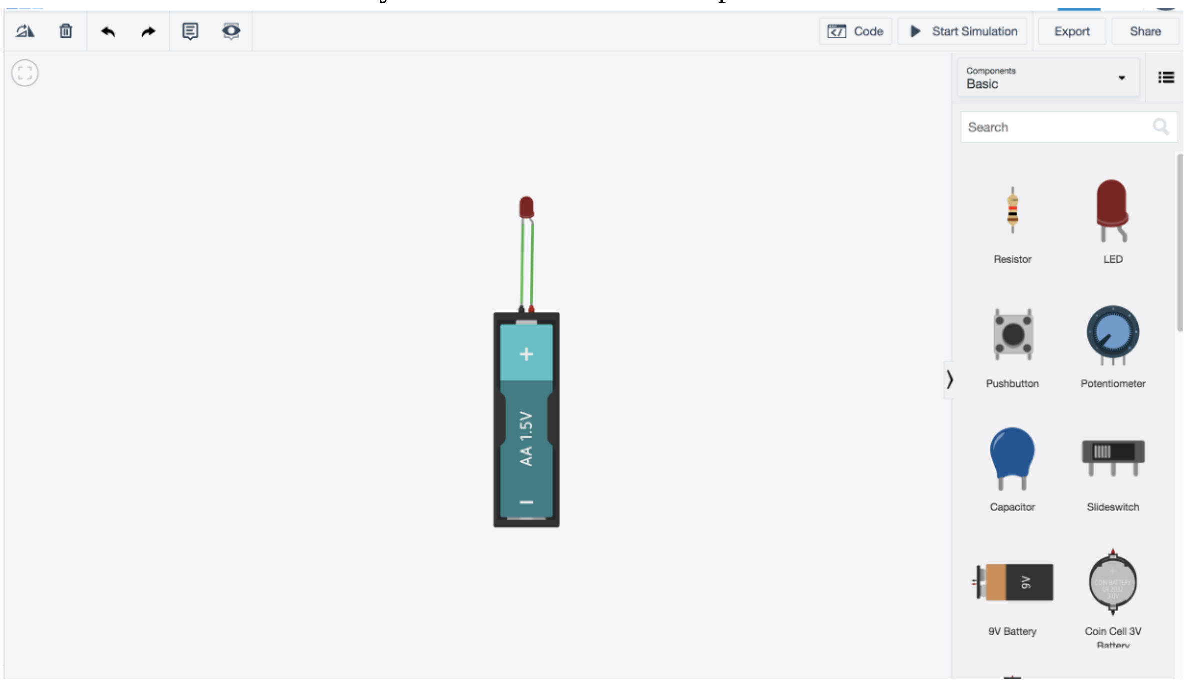
Coding Circuits	Grade 6 Electricity and Electrical Devices

Coding Guide

Activity One: Building a Simple Circuit

First we'll have to locate the components for a simple circuit in the Components > Basic panel and move them into the programming canvas

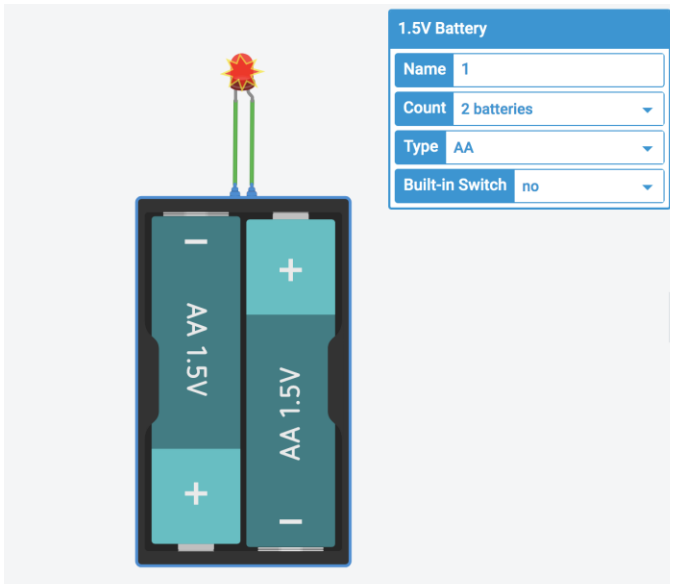
- Find the LED and the 1.5V battery in the components menu and drag them into the programming canvas.
- Draw wires by clicking on the startpoint and endpoint with your mouse or trackpad. A wire will automatically be drawn between these points.



(Descriptive text: Image showing a red LED connected to a AA battery with two wires to form a simple circuit. To the right, a panel is shows a list of components that can be used to build a circuit)

Adding Resistors

We are going to increase the amount of current coming from our power source by adding a second battery. Do this by left-clicking on the power source and updating the 1.5V battery count to **2 batteries.**

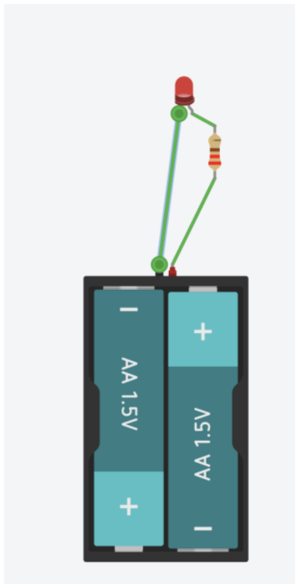


1.5V Battery

Name	1
Count	2 batteries
Type	AA
Built-in Switch	no

(Descriptive text: Image showing a red LED connected to two AA batteries with two wires to form a simple circuit. The LED is covered by a red explosion symbol.)

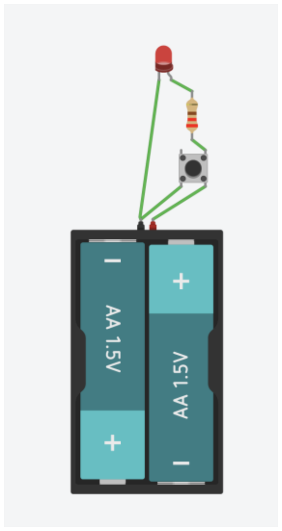
- The current running through the LED is beyond the maximum that our load can support. Overloading a circuit can cause the wiring to overheat, ruin the device that you’re trying to power, or start a fire. To reduce the current from our load, we can add a resistor. Grab a resistor from the components panel and drag it into the programming canvas.
- Reconnect the circuit by redrawing the wires so that a wire connects the negative terminal of the battery pack to terminal 1 of the resistor, and a wire connects terminal 2 of the resistor to the LED.
- Start the simulation. What happens to your circuit?



(Descriptive text: Image showing a simple circuit with a red LED, two green wires, a battery pack, and a resistor. The LED is lit.)

Adding a Switch

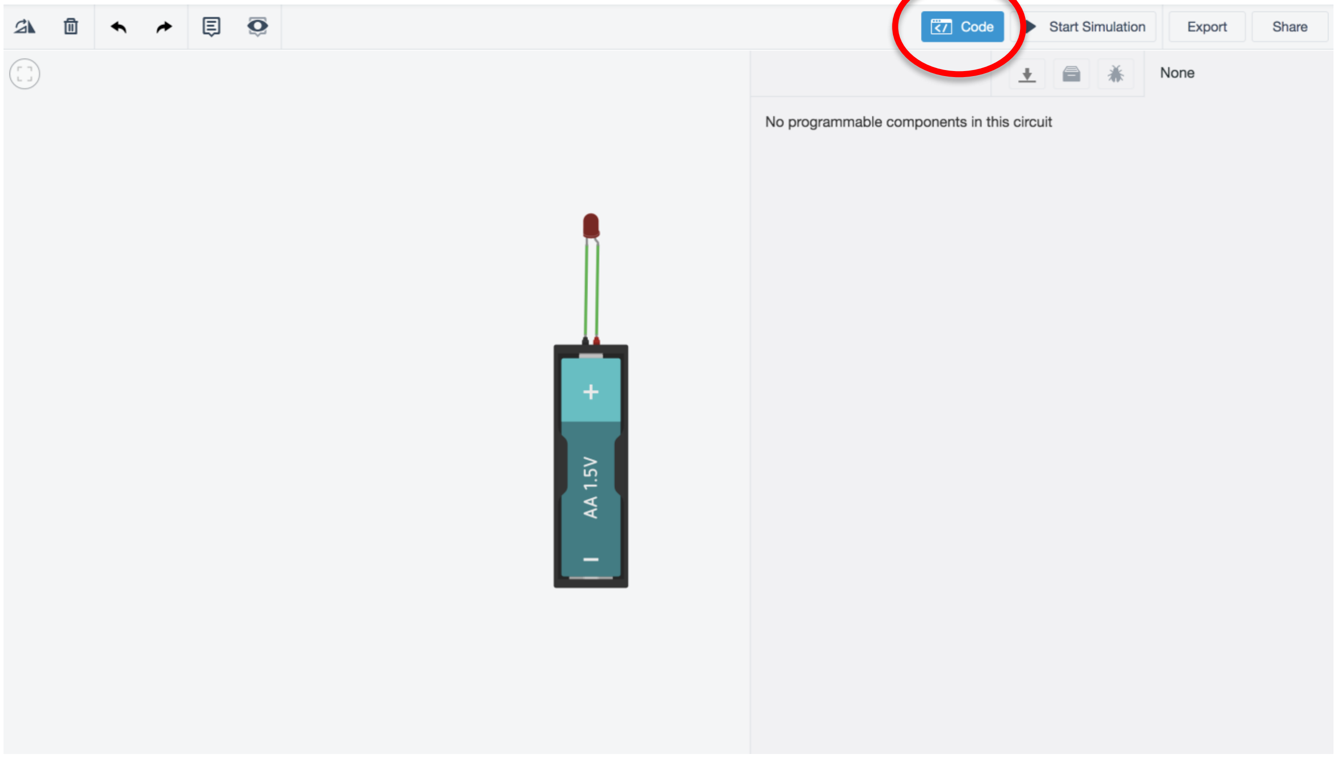
- Let's add a switch that will allow us to control our LED. From the components menu, choose the pushbutton and drag it into the programming canvas. We will want to integrate our button into the circuit by connecting it to both terminals of our power source, as well as to the anode of our LED (via a resistor). See the image below, which illustrates one possible way to connect the switch.



(Descriptive text: Image showing a simple circuit with a red LED, two green wires, a battery pack, and a resistor and a switch. The LED is lit.)

Activity 2: Coding Circuits

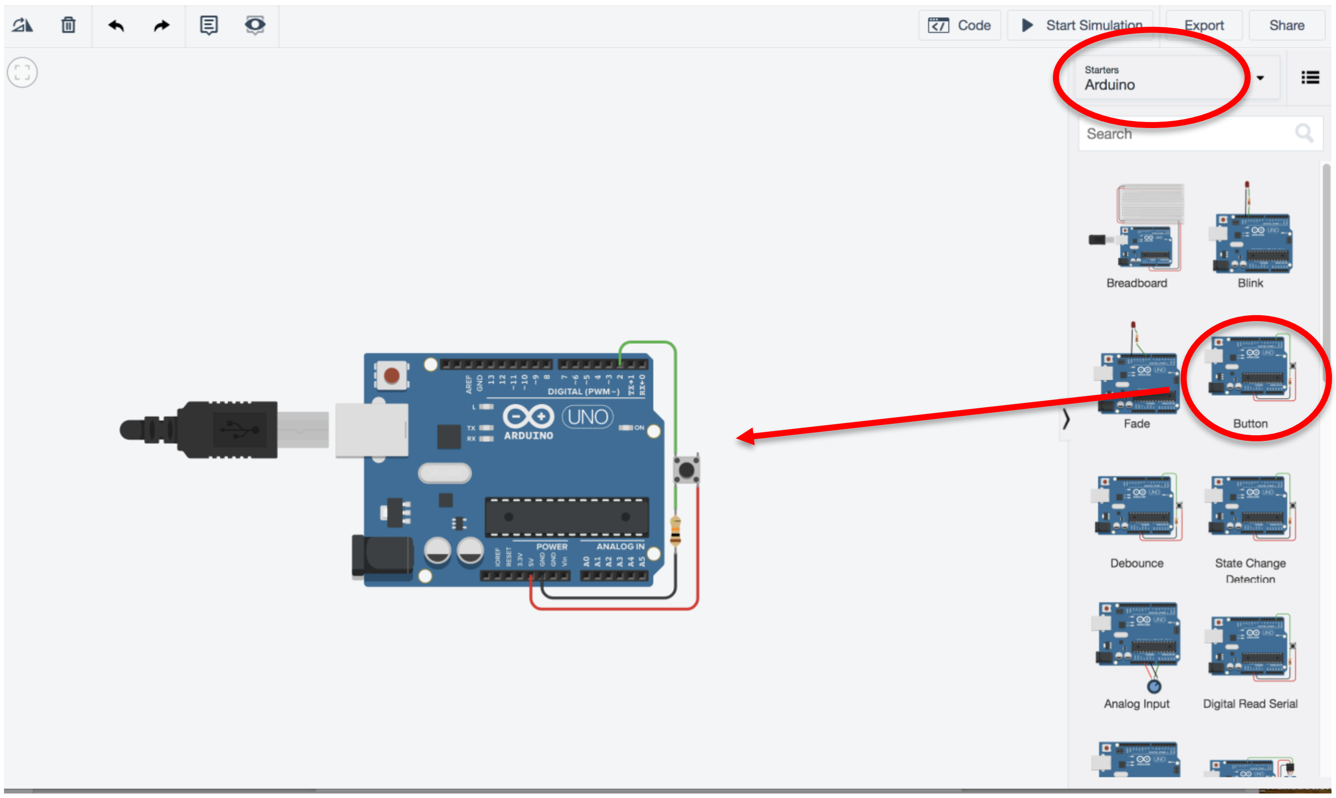
Next to the Start/Stop simulation button, you will notice that there is a **Code** button. Clicking this button will reveal code to control our circuit output (in this case, how our LED lights up). With our current circuit, there is no code because our circuit is not currently programmable (the only control we have is whether our circuit is open (off, or 0), or closed (on, or 1).



No programmable components in this circuit

(Descriptive text: image shows the entire TinkerCAD application screen. On the upper right, a button labelled "Code" is highlighted, and a panel is open so show that there are no programmable components in this circuit)

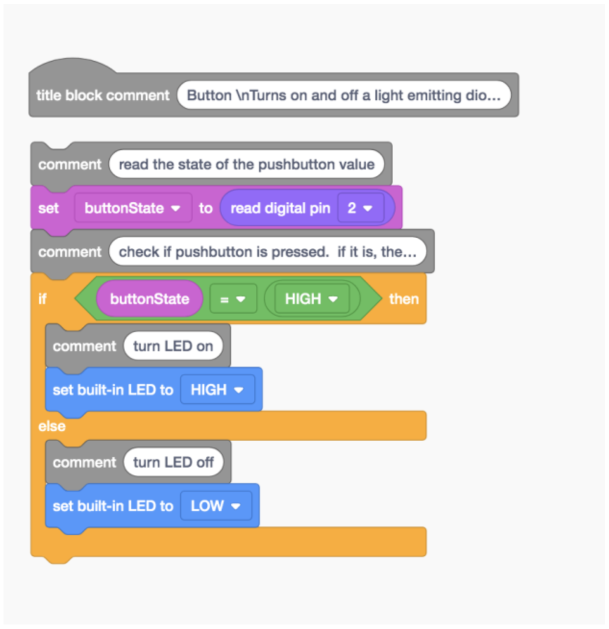
- In the **Components** drop-down menu, choose **Starters** > **Arduino.** This will open programmable component options and pre-built examples. Drag the example **Button** into the programming canvas.
- This circuit looks very similar to the circuit that we made previously. Our Arduino is a programmable microcontroller powered by USB. The current flows through pin 13, is controlled via a resistor to the load (the LED), which is, in this case, a tiny built-in light on the Arduino.



(Descriptive Text: Image shows TinkerCAD Circuits with the Starters > Arduino menu open on the right side of the screen. A sample Arduino circuit is in the programming canvas.)

- If you press the Code button, you will see block code to program the circuit (If you have used Scratch or Blockly before, you may note that this coding format is very similar).

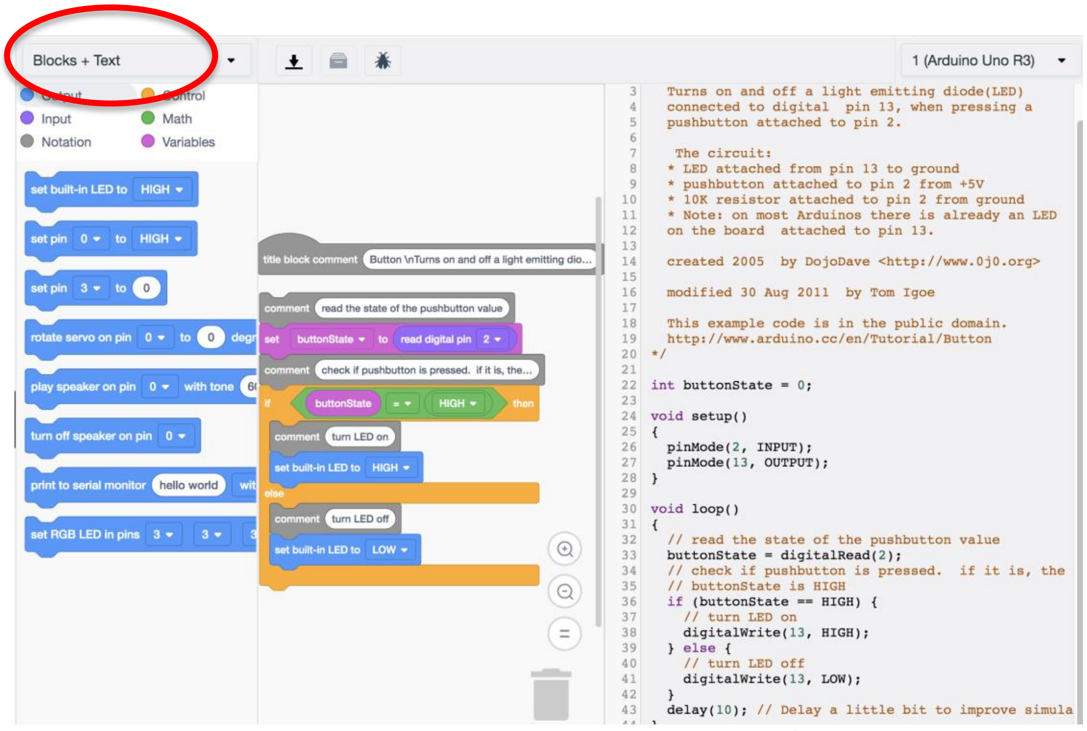
The existing code is a loop (although no loop block exists in this case — if you choose to change your view to Blocks + Text, you will be able to see the loop in C, the coding language used for Arduino) that checks the pin connected to the pushbutton (pin 2) to see if the button is being pressed (HIGH, or 1) or not (LOW, or 0). We then see a conditional statement for both pushbutton states. If the buttonState = HIGH, then the controller sets the pin controlling the LED (pin 13) to HIGH and the LED turns on; else, the pin controlling the LED is set to LOW and the LED is turned off.



```
title block comment  Button \nTurns on and off a light emitting dio...

comment  read the state of the pushbutton value
set  buttonState ▾  to  read digital pin  2 ▾
comment  check if pushbutton is pressed.  if it is, the...
if  buttonState  = ▾  HIGH ▾  then
    comment  turn LED on
    set built-in LED to  HIGH ▾
else
    comment  turn LED off
    set built-in LED to  LOW ▾
```

(Descriptive Text: Image showing block code that describes the program that uses Arduino and a pushbutton to control an on-board LED light)

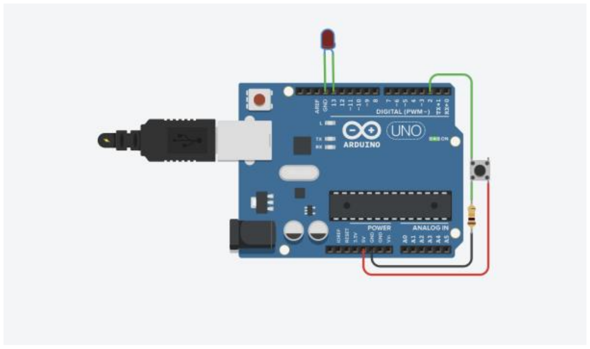
To compare your block code program to the same program as written in the C programming language, select Blocks + Text from the dropdown menu above your coding blocks.


Blocks + Text
Output
Control
Input
Math
Notation
Variables
set built-in LED to HIGH
set pin 0 to HIGH
set pin 3 to 0
rotate servo on pin 0 to 0 degr
play speaker on pin 0 with tone 6
turn off speaker on pin 0
print to serial monitor hello world wit
set RGB LED in pins 3 3 3
1 (Arduino Uno R3)

```
  Turns on and off a light emitting diode(LED)
  connected to digital  pin 13, when pressing a
  pushbutton attached to pin 2.

   The circuit:
  * LED attached from pin 13 to ground
  * pushbutton attached to pin 2 from +5V
  * 10K resistor attached to pin 2 from ground
  * Note: on most Arduinos there is already an LED
  on the board  attached to pin 13.

  created 2005  by DojoDave <http://www.0j0.org>

  modified 30 Aug 2011  by Tom Igoe

  This example code is in the public domain.
  http://www.arduino.cc/en/Tutorial/Button
*/

int buttonState = 0;

void setup()
{
  pinMode(2, INPUT);
  pinMode(13, OUTPUT);
}

void loop()
{
  // read the state of the pushbutton value
  buttonState = digitalRead(2);
  // check if pushbutton is pressed.  if it is, the
  // buttonState is HIGH
  if (buttonState == HIGH) {
    // turn LED on
    digitalWrite(13, HIGH);
  } else {
    // turn LED off
    digitalWrite(13, LOW);
  }
  delay(10); // Delay a little bit to improve simula
```

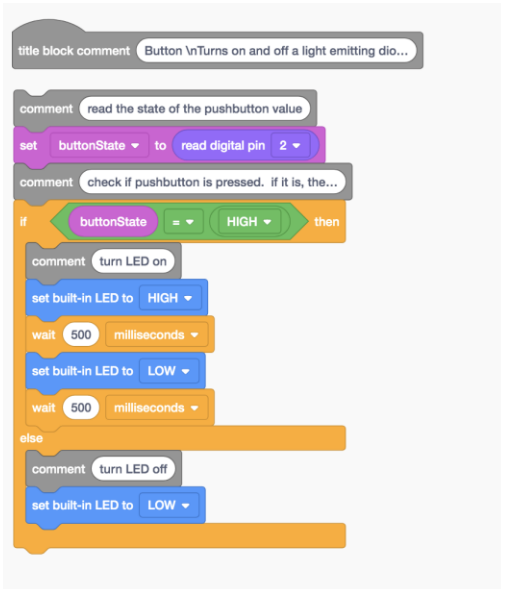
(Descriptive Text: Image showing a side-by-side comparison of a block code program and the same program written in C.)

- Let’s add an external LED to see the effects more clearly. We can drag an LED to the programming canvas and connect the cathode to the Ground (GND) pin and the anode to pin 13 (the LED pin on the Arduino board). Now, when you push the button, the red LED should light up. The LED can be found by navigating to the **Components** > **Basic** menu.



(Descriptive text: Image showing an Arduino board with wires connecting to a pushbutton and a red LED.)

- Let’s change the output of our conditional statement so that the LED blinks on an off on its own while the pushbutton is pushed. We will do that by adding a loop within the if statement of our conditional. The loop will set the LED to HIGH, wait for 0.5 seconds (500 milliseconds), set the LED to LOW, wait for 0.5 seconds (500 milliseconds), and repeat as long as the button is pushed.



```
title block comment  Button \nTurns on and off a light emitting dio...

comment  read the state of the pushbutton value
set  buttonState ▾  to  read digital pin  2 ▾
comment  check if pushbutton is pressed.  if it is, the...
if  buttonState  = ▾  HIGH ▾  then
    comment  turn LED on
    set built-in LED to  HIGH ▾
    wait  500  milliseconds ▾
    set built-in LED to  LOW ▾
    wait  500  milliseconds ▾
else
    comment  turn LED off
    set built-in LED to  LOW ▾
```

(Descriptive text: Image showing block code that describes the program that uses Arduino and a pushbutton to control an LED light to make it blink.)

- Try experimenting with the parameters of your code. Can you create your own blinking pattern by adjusting the wait blocks?