Glow Your Own – Session 3

For a reminder of how to access Tinkercad and get started, and send your code to a physical Arduino, see the 'How To' sheets for Sessions 1 and 2. There are instructions for how to upload your code to a physical Arduino from a Chromebook later in this How To sheet.

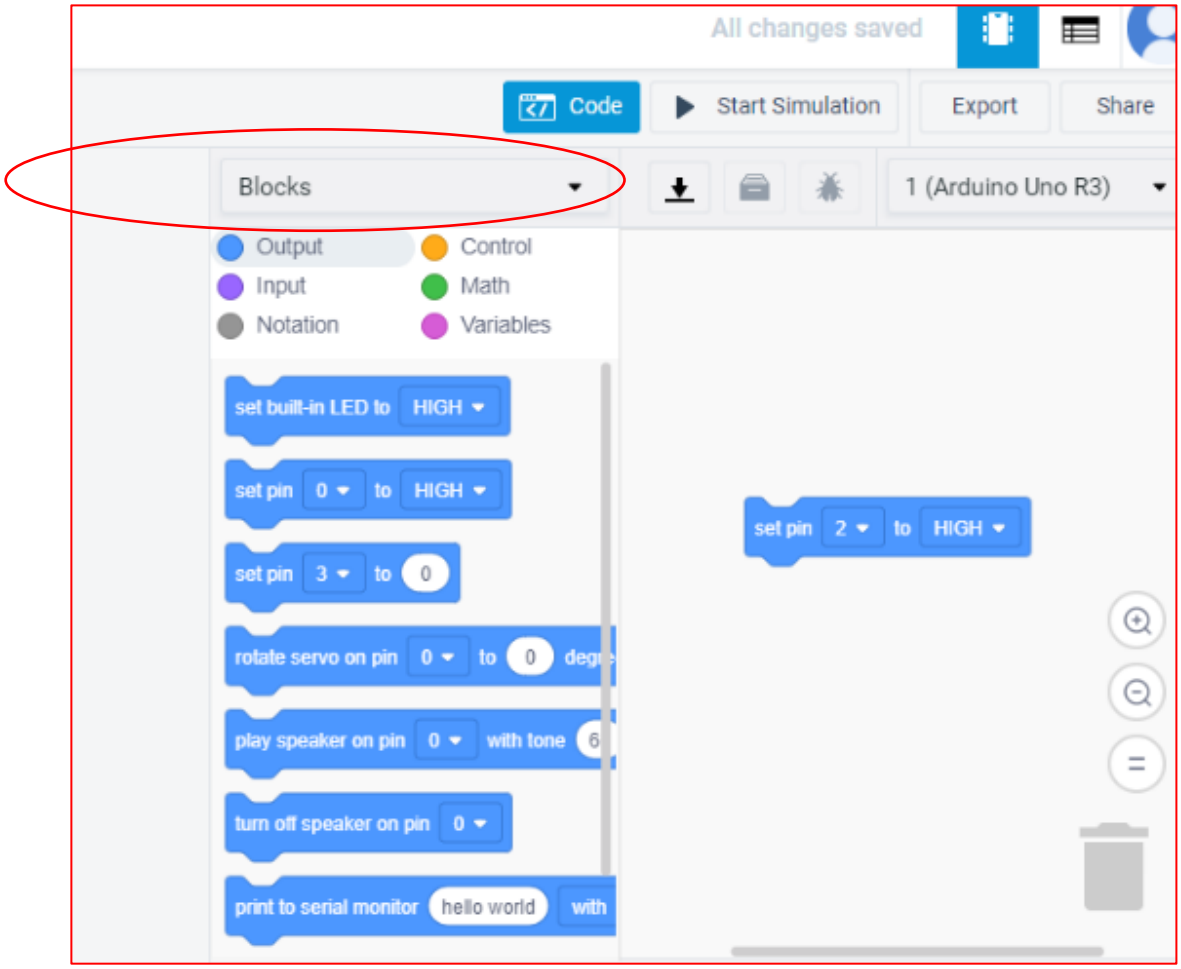
Remember, if you're using building physical circuits with your real Arduino:

- Always remember to unplug your Arduino when you're changing components.
- Make sure all of the small components are kept out of reach of young children.
- Make sure that all components are tidied up at the end of the session, and none are left on the floor or table.

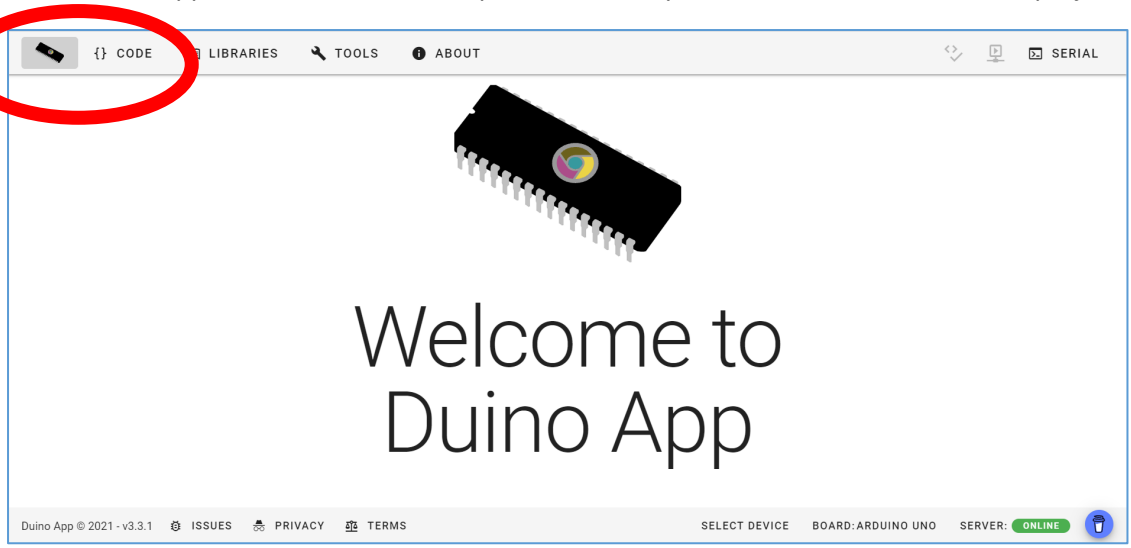
Sending your code to the physical Arduino from a Chromebook

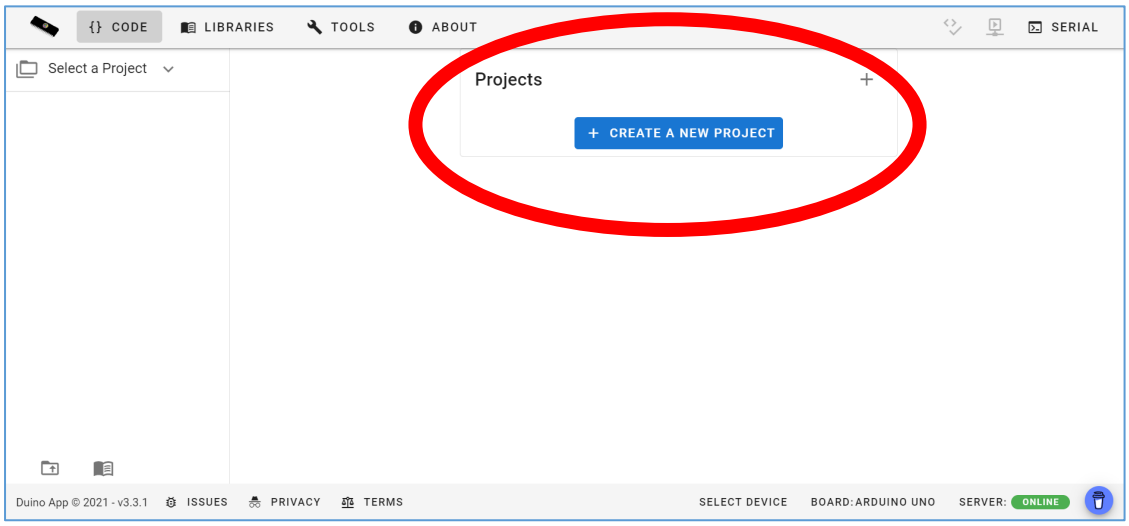
To send (or "upload") your code to your physical Arduino, the best option if you are using a Chromebook is to use the **Duino App: https://duino.app/#/**. This will also work from a Chrome browser.

Once you've built your circuit, use the USB cable to connect your Arduino to your computer. If you need to change your electrical components, always remember to unplug your Arduino from your computer first! Then in Tinkercad, in the Code section, change "Blocks" to "Blocks + Text"

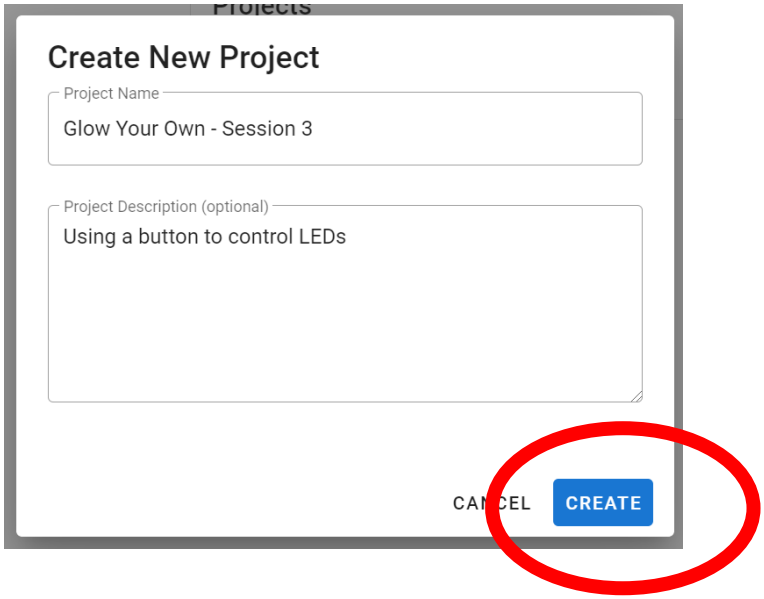


In the Duino App, click “Code”, in the top left corner of your screen, then “Create new project”

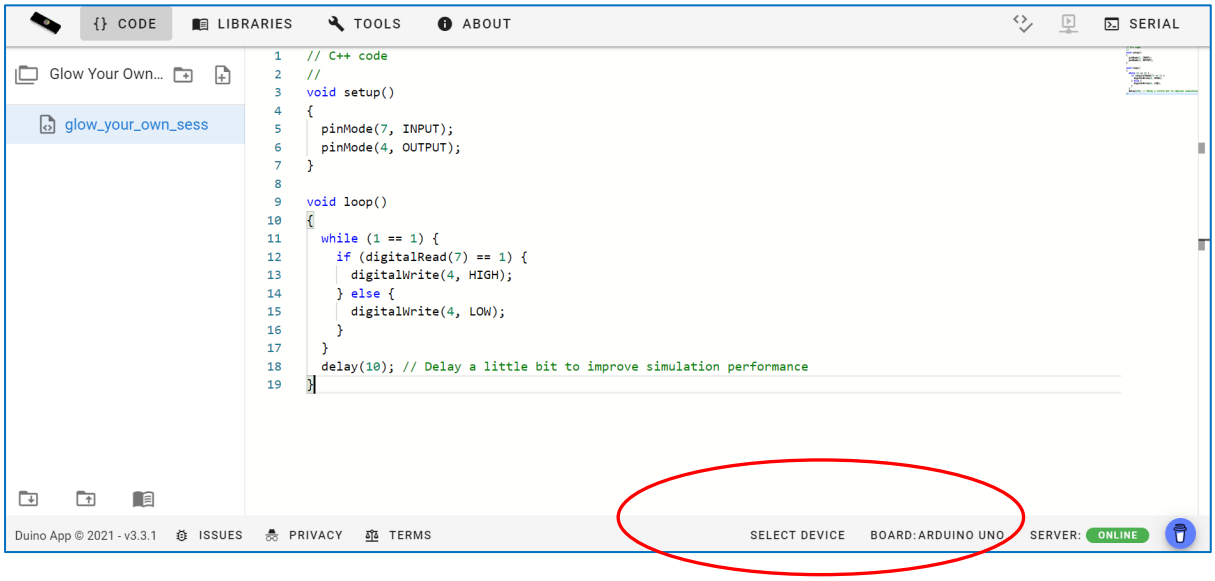




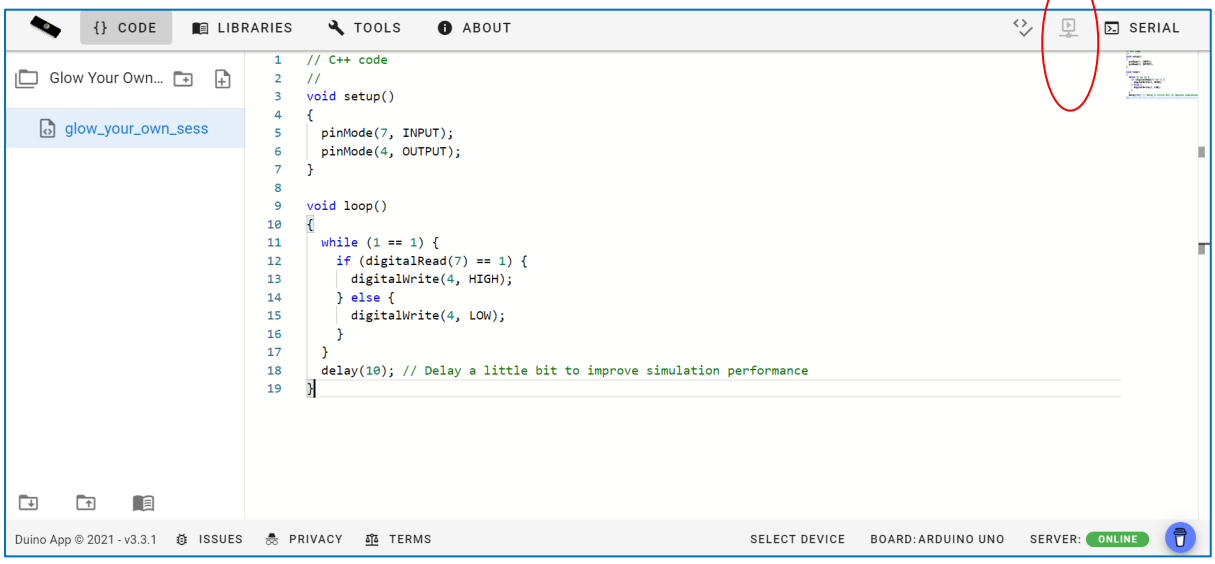
Name your code and write a short description (we’ve named ours “Glow Your Own – Session 3”), then click “Create”.



Delete the standard code that appears, and copy and paste from the Text section of Tinkercad into the Duino App, then check that the Device, in the bottom right corner, is “Arduino Uno”. You can change the device by clicking “Select Device”.



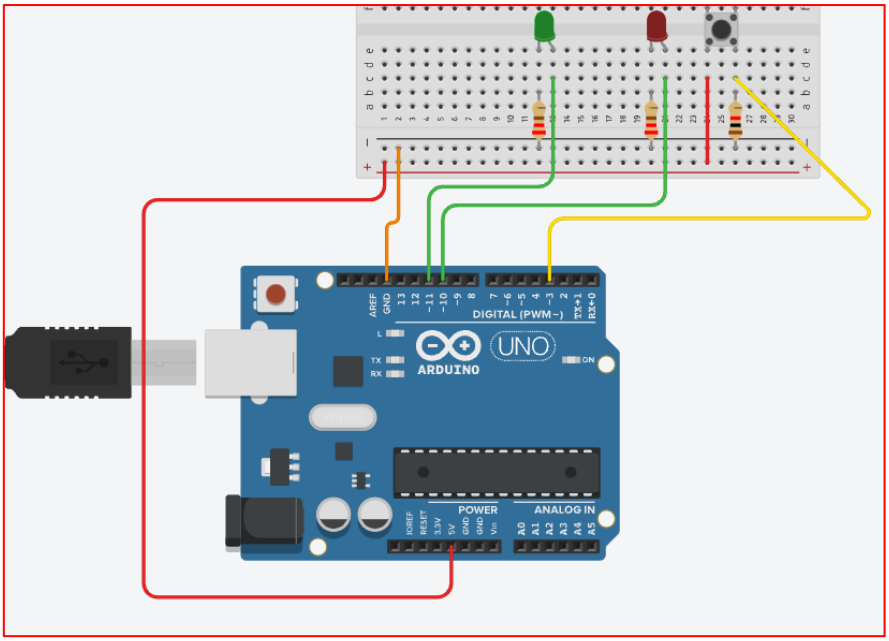
Send your code to your physical Arduino by clicking “Compile and upload”, in the top left corner.



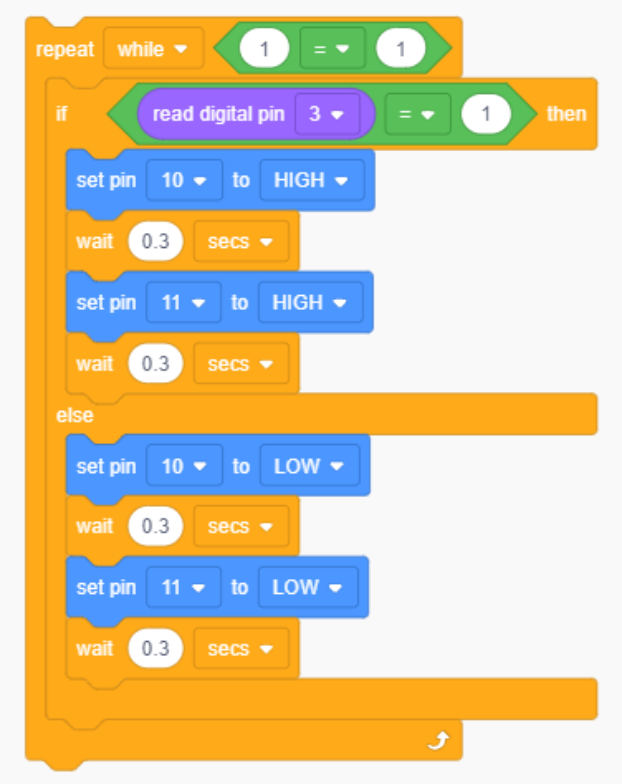
Circuits to build and code

Turning an LED on with a button

Lights that are always on can be a bit annoying! You can control your LED by clicking a button. You need the same type of circuit as with the photoresistor, as below (use a 1 kΩ resistor with your button):



Don’t forget the red wire on the left hand side of the diagram! This links the button to the 5V (power) pin on the Arduino – the button needs this power to work! To code our new circuit, we need to use an “**if then else”** block: this block is used when we want to ask the computer a question. We ask whether the button has been pushed. If it has been pressed, we instruct the Arduino to turn the LEDs on (one after the other), if the button is not pressed, we instruct the Arduino to turn the LEDs off (one after another), using the code below.



When you simulate your circuit on Tinkercad, in order to press the button just click-and-hold the button on your screen.

Alternatively, you can copy and paste the below code into your UNO R3 Software (or the Duino App) and export it to your Arduino. *[Make sure the lines of code are displayed as below, by either copying and pasting one line at a time or copying the whole code and adjusting the line spacing using your Enter key whilst in the UNO R3 Software.]* The lines that start "//" are **comments**. They're little explanations in the code, that says what the code is doing. They're really helpful, especially when you're coding with other people!

```
// constants won't change. They're used here to set pin numbers:
const int buttonPin = 3;     // the number of the pushbutton pin
const int ledPin1 =  10;      // the number of the LED pin
const int ledPin2 =  11;      // the number of the LED pin


// variables will change:
int buttonState = 0;         // variable for reading the pushbutton status

void setup() {
  Serial.begin(9600);
  // initialize the LED pin as an output:
  pinMode(ledPin1, OUTPUT);
    pinMode(ledPin2, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT_PULLUP);
}

void loop() {
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed. If it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin1, HIGH);
    delay(300);
    digitalWrite(ledPin2, HIGH);
    delay(300);
  } else {
    // turn LED off:
    digitalWrite(ledPin1, LOW);
    delay(300);
    digitalWrite(ledPin2, LOW);
    delay(300);

}

Serial.println(buttonState);
```

This code can be copied and pasted in the correct format, via this GitHub link:
https://raw.githubusercontent.com/Isioviel/glow-your-own/main/example.txt

Adding a power source

We don't always want our Arduino to be connected to a computer while it's working. We can add an alternative power source to our circuit, which means that it will continue working after being disconnected from the computer (so long as the code works!)

To do this, simply connect your power source to "Vin" and "GND" pins on the Arduino, as in the circuit diagram below.