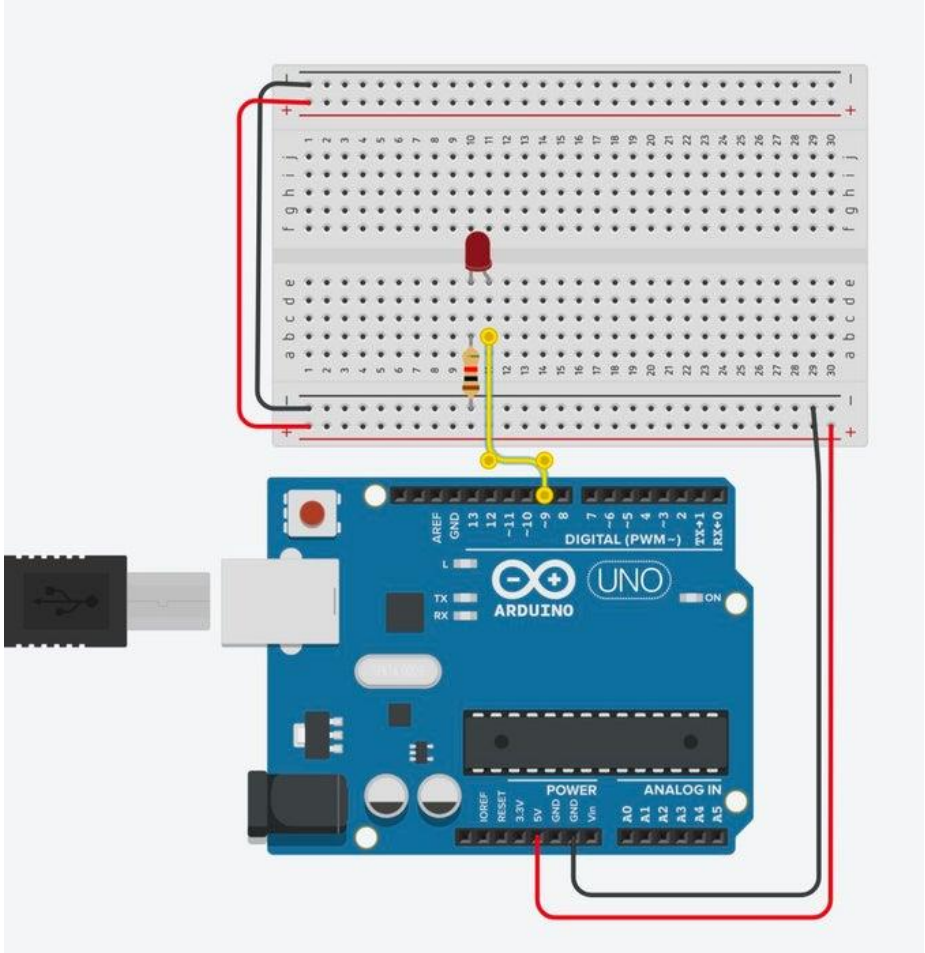
CSE423: Embedded System
Summer-2020
Simulation with TinkerCAD
(Online Simulator-2)

Todays Lecture

- *Understanding TinkerCAD in detail*
- *LED dimmer using PWM supported pin*
- *Tutorial available at:* https://youtu.be/X8dHbdhnGKY

Simulate a LED Dimmer step by step

Step-1: Connect a LED Circuit like the regular one in the breadboard.



Simulate a LED Dimmer step by step

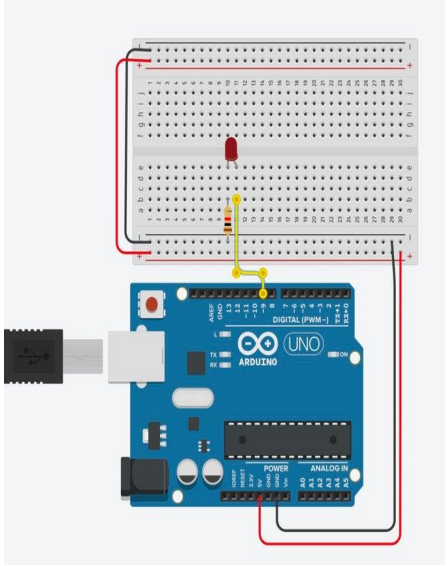
- The LED is connected in series with a resistor between Arduino pin 9 and ground.

Follow the steps for connections:

- Breadboard power (+) and ground (-) rails to Arduino 5V and round(GND)respectively.

- LED cathode (negative, shorter leg) to one leg of a resistor (anywhere from 100-1K ohms is fine)

- Other resistor leg to ground LED anode (positive, longer leg) to Arduino pin 9



Simulate a LED Dimmer step by step

- Step 2: Build Brightness Adjustment Code With Blocks

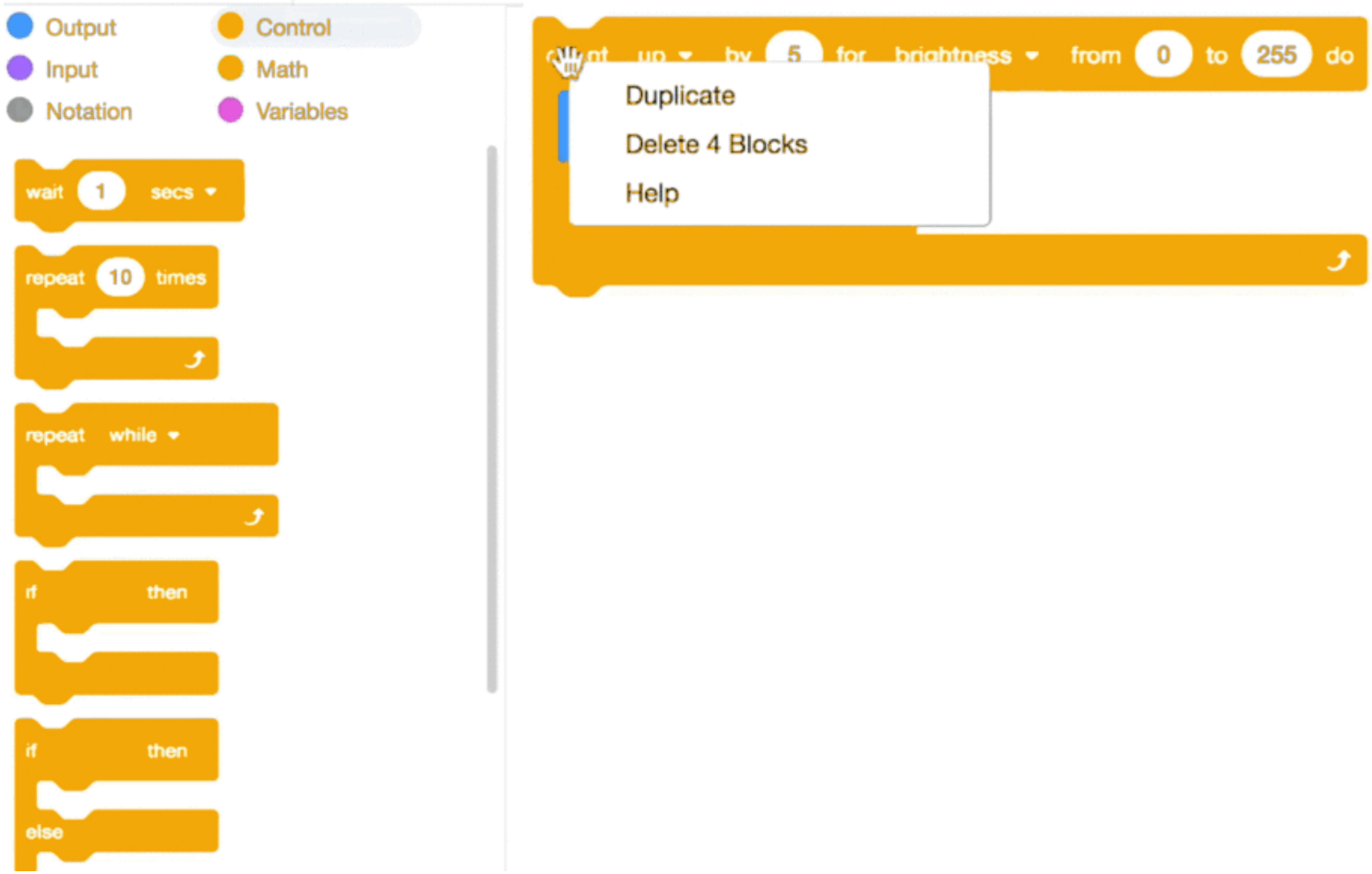
  Click Code Editor to open the code blocks editor

- Start with a control block that counts. Set it to count up by five. Click the dropdown next to "for" and select "rename variable...", then rename it to "brightness". Adjust the "from" and "to" values to 0 and 255, respectively.

- Inside the counting loop, add an output block to set one of the special pins, and adjust it to pin 9. Navigate to Variables and drag the brightness block to the output block to set pin 9 to the current value of brightness, which changes over the course of the counting loop.

- Add a wait block, and set it to 30 milliseconds. This gives time for the light to shine at each brightness level so you have time to see it before it changes. The duration of this block can be changed to slow down or speed up the fading effect.

Simulate a LED Dimmer step by step

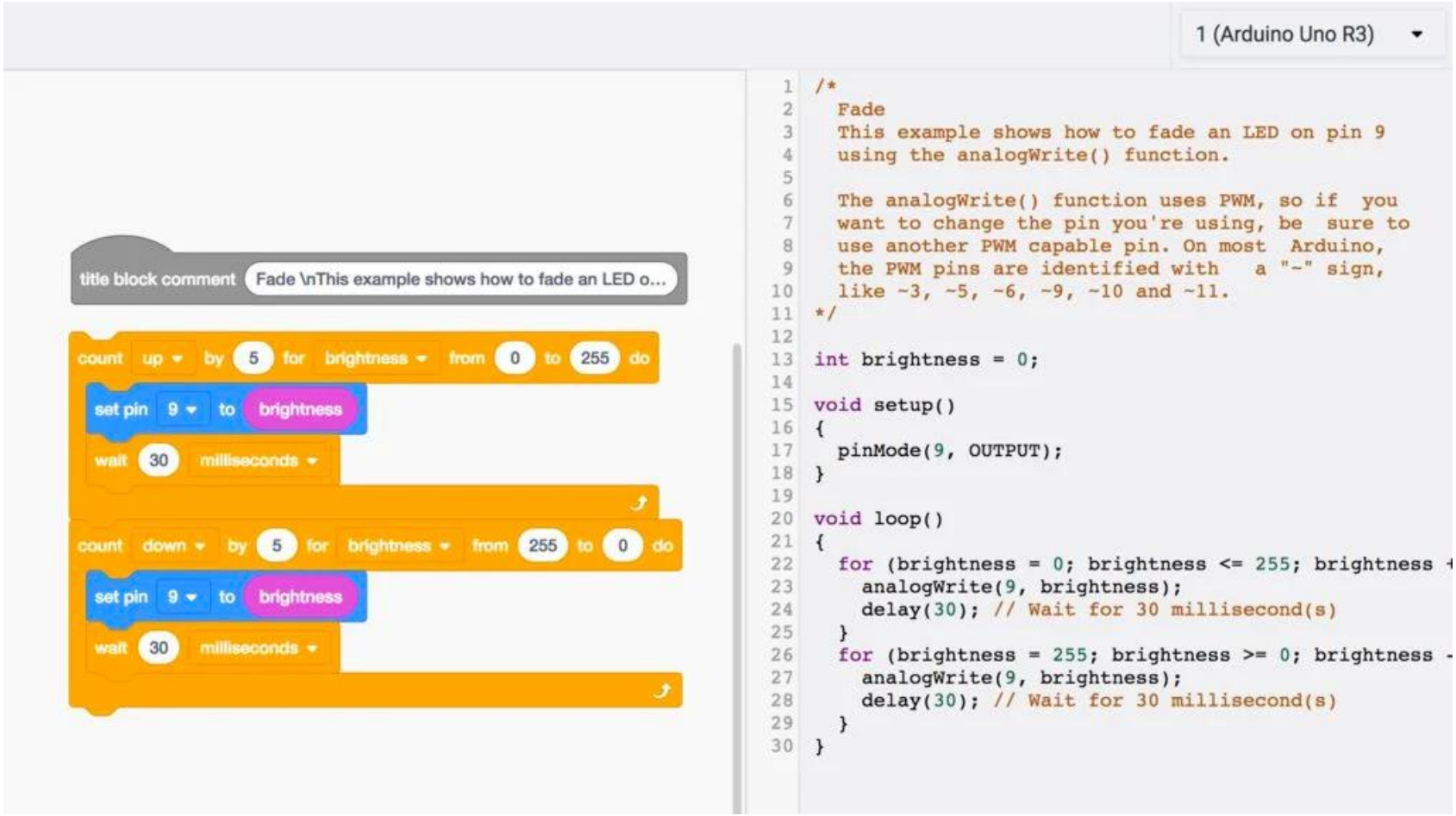


Simulate a LED Dimmer step by step

- **Step 3: Brightness Adjustment Arduino Code Explained**

The counting loop you created fades the LED from off to all the way on. To fade the LED back off again, we have to create another counting loop. Either drag a new counting loop into the editor, or duplicate this one, and this time change it to count down, start with 255, and go down to zero.

Simulate a LED Dimmer step by step



Simulate a LED Dimmer step by step

/* Fade

This example shows how to fade an LED on pin 9 using the analogWrite() function.The analogWrite() function uses PWM, so if you want to change the pin you're using, be sure to use another PWM capable pin. On most Arduino, the PWM pins are identified with a "~" sign, like ~3, ~5, ~6, ~9, ~10 and ~11. */

```
int brightness = 0;

void setup()
{
  pinMode(9, OUTPUT);
}

void loop()
{
  for (brightness = 0; brightness <= 255;
  brightness += 5)
  {
    analogWrite(9, brightness);
    delay(30);          // Wait for 30 millisecond(s)
  }
  for (brightness = 255; brightness >= 0;
  brightness -= 5) {
  analogWrite (9, brightness);
  delay(30);          // Wait for 30 millisecond(s)
  }
}
```

Simulate a LED Dimmer step by step

- The program's loop uses two for loops to count up from 0 to 255 by increments of 5. The analogWrite() function takes two arguments: the Arduino pin number (9 in our case), and a value between 0 (off) and 255 (all the way on).

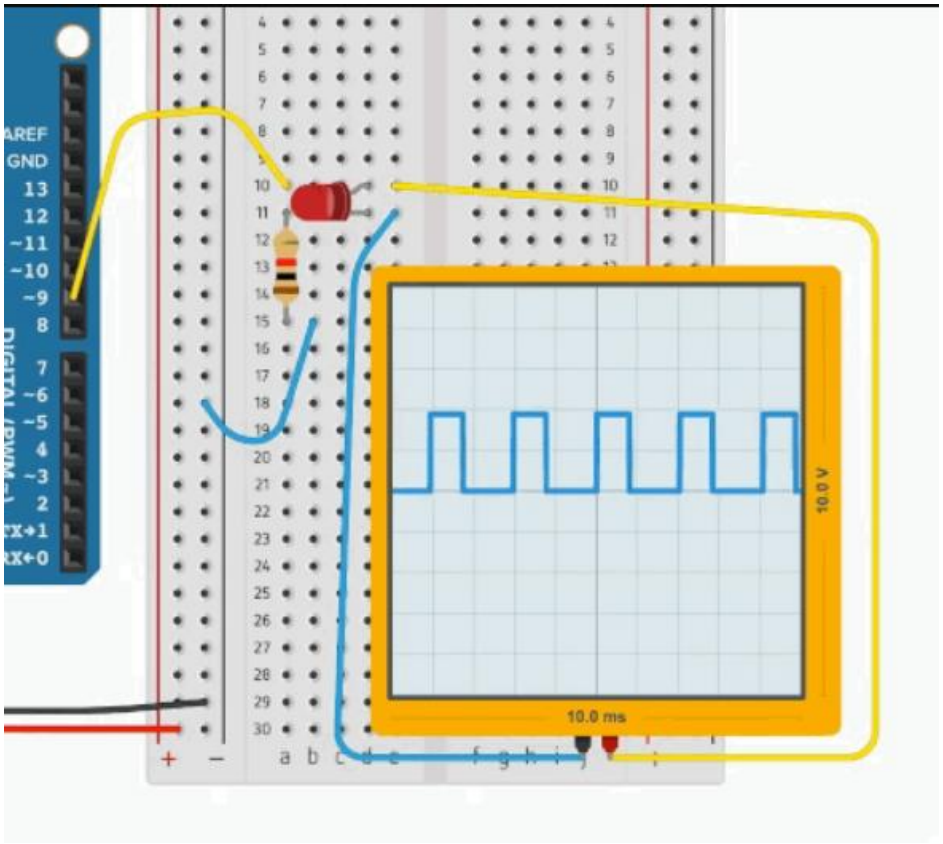
Simulate a LED Dimmer step by step

- This circuit is also available as a fade circuit starter. You can use this circuit starter anytime you want to fade an LED.
- Grab this circuit and code combo any time using the starter available in the components panel (dropdown menu -> Starters -> Arduino).
- Scroll to find the Arduino starter labeled **Fade**, and double click it to add it to the workplane (you can also click and drag instead).

- Notice how the resistor in this version is "upstream" of the LED, connected between power and the LED instead of the LED and ground. These two circuits both make the same connections, linking up the LED to a signal pin and ground, through a current limiting resistor, which will function on either side of the LED.

Simulate a LED Dimmer step by step

- Task: Check it with Pulse Width Modulation (PWM)



Simulate a LED Dimmer step by step

- Pulse Width Modulation

The Arduino board is only capable of generating digital signals (HIGH and LOW), but analogWrite(); simulates the appearance of brightnesses between on and off using pulse width modulation (PWM). The LED flashes on and off very quickly, and your eye interprets a dimmer light. The ratio of time the LED spends on vs. off determines how bright or dim the LED appears.

Pulse width modulation (PWM) creates an oscillating digital signal, alternately driven high and low in a repeating pattern. Each high to low to high period of time is called a cycle.

You can see an oscillating digital signal on the oscilloscope like the animation above.

Notice that for each cycle, the width of the HIGH and LOW portions of the graph are changing, hence the term pulse width modulation, or PWM for short.

Identify the other digital pins on the Arduino Uno capable of PWM, marked with a ~: 3, 5, 6, 9, 10, and 11.