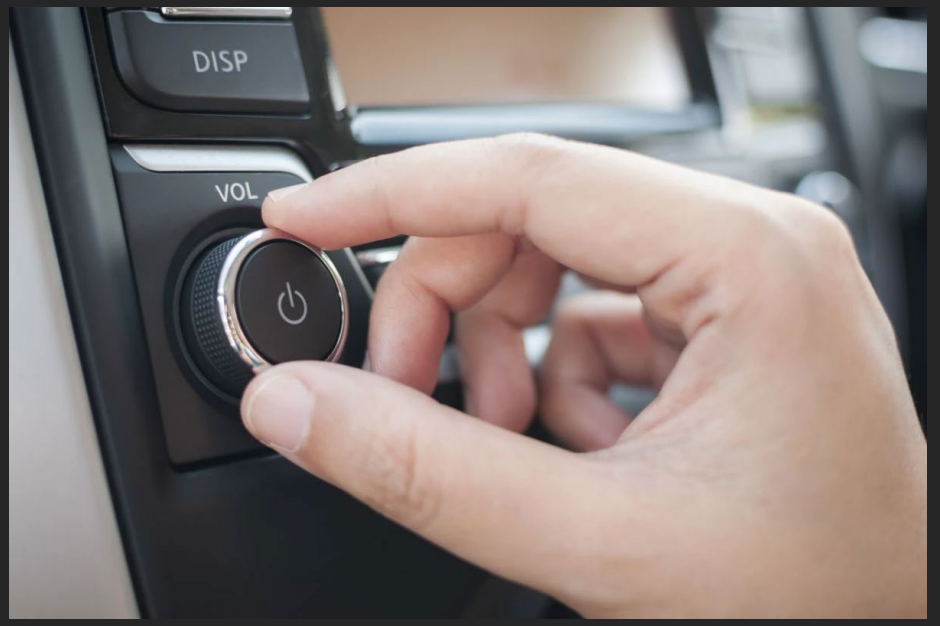
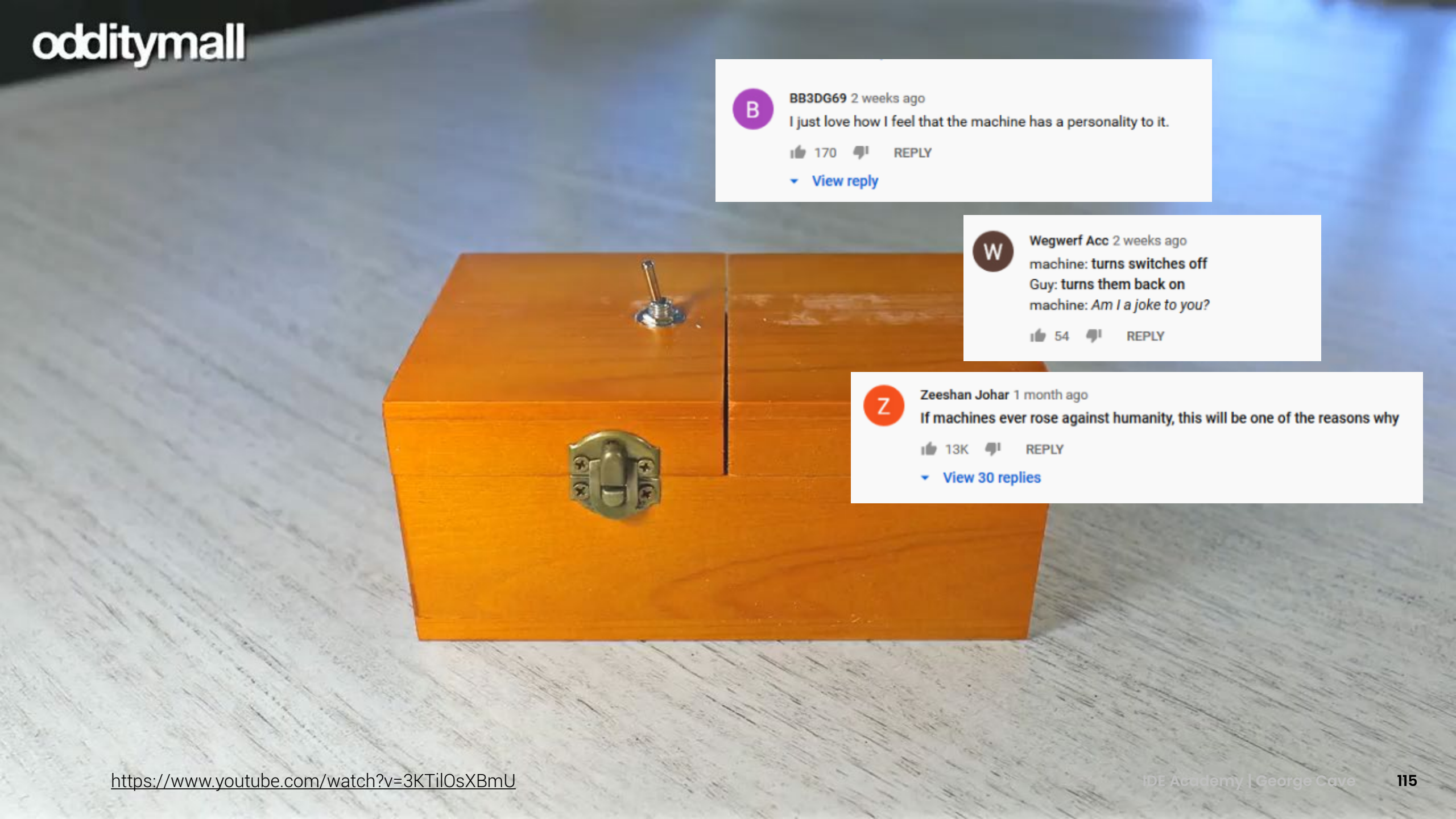

prototyping
the interaction


CD/AUX
RADIO
SCAN
4
5
RPT
6
RDM

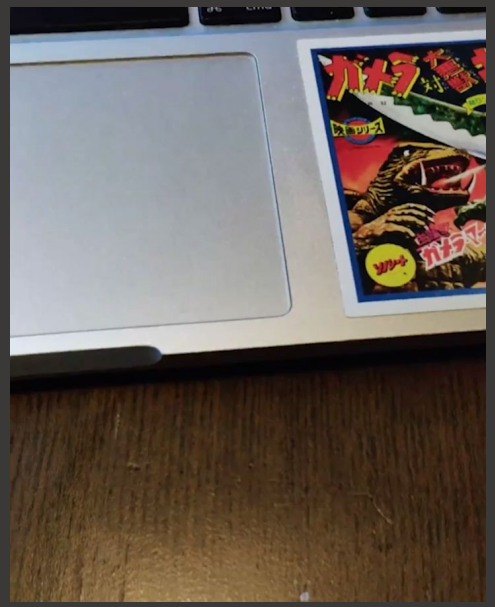

DISP
VOL

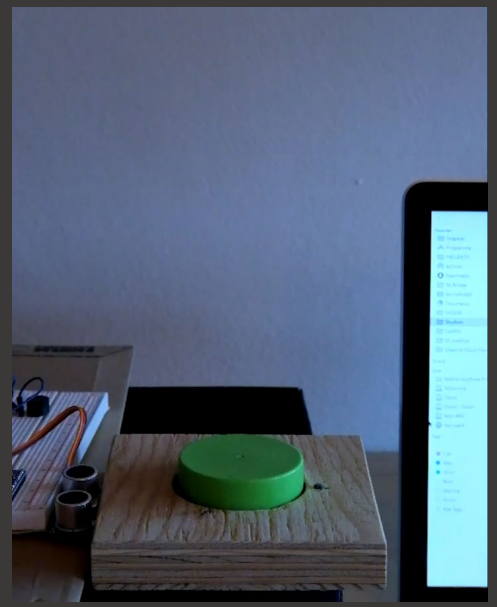
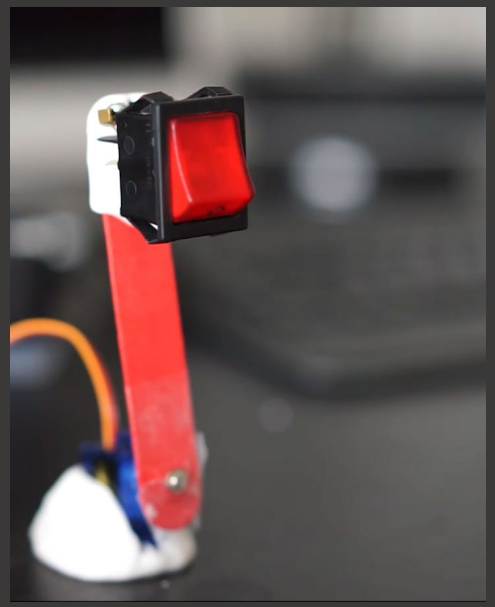

CD IN
RPT
RDM
PWR
VOL
SEEK


SCAN
CD
AUX
VOL
PUSH
SELECT
SETUP



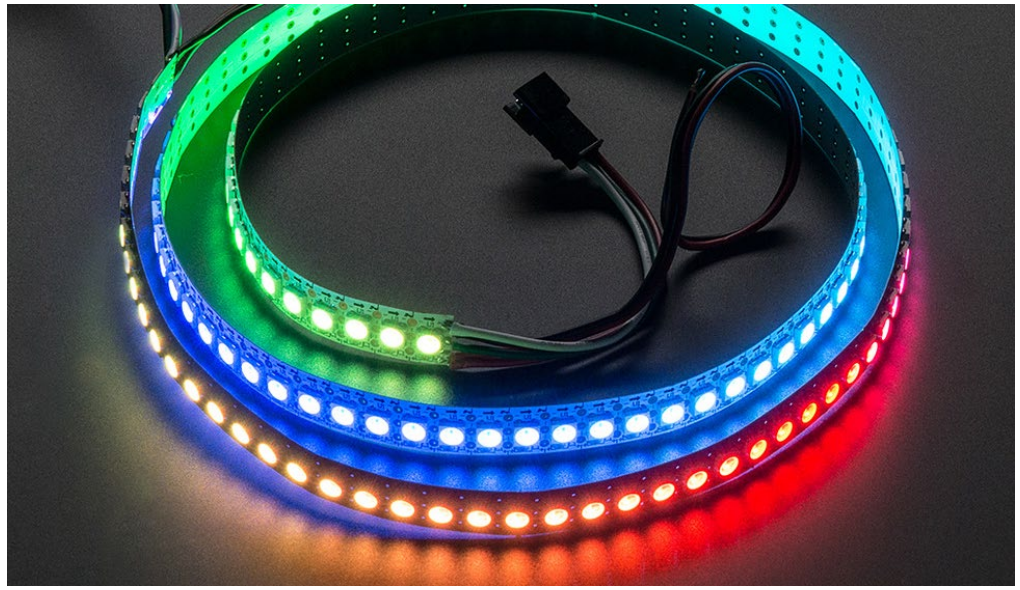
https://www.youtube.com/watch?v=3KTilOsXBmU



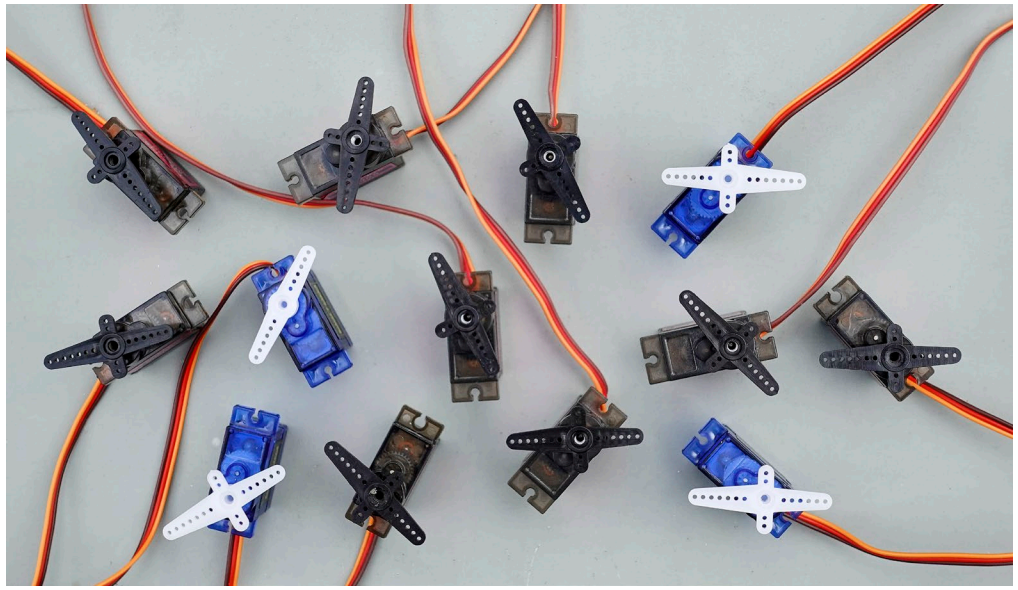



2 fundamental outputs

Light

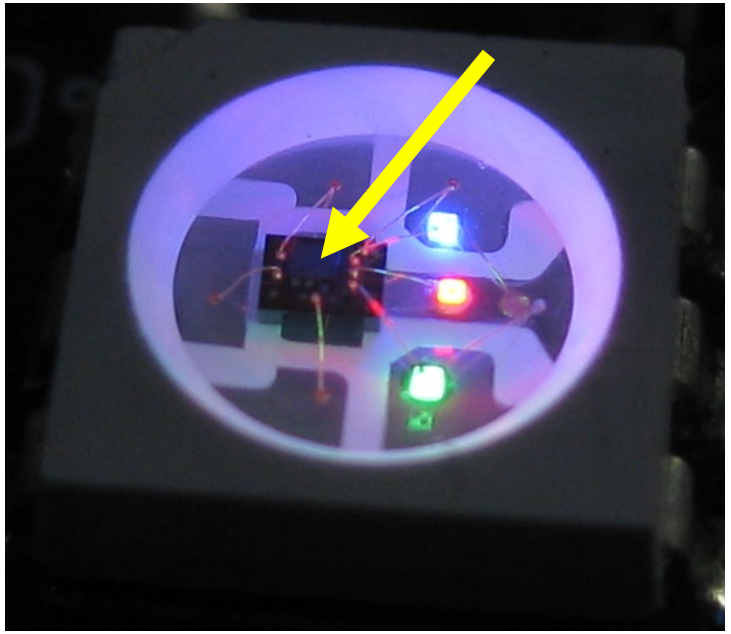


Motion



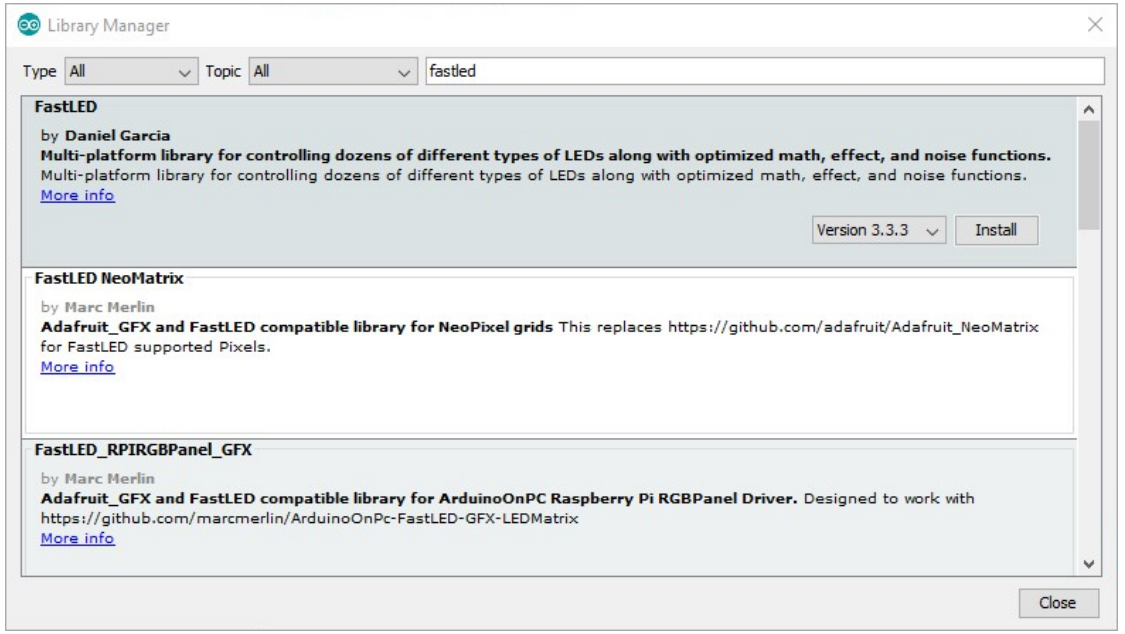
The plan...

RGB LEDs + FastLED



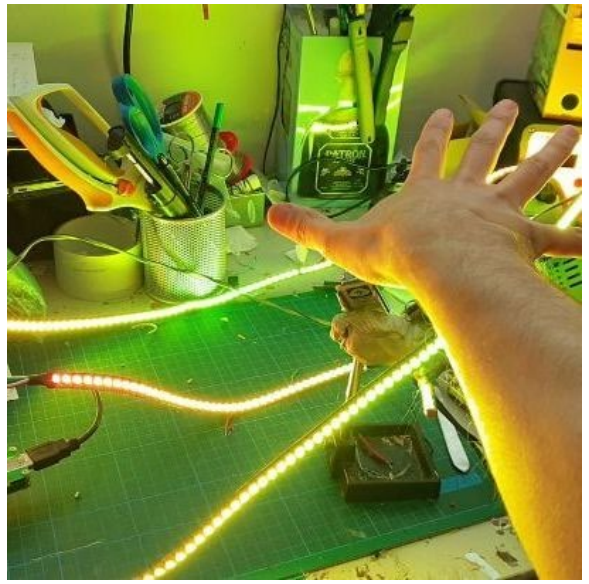
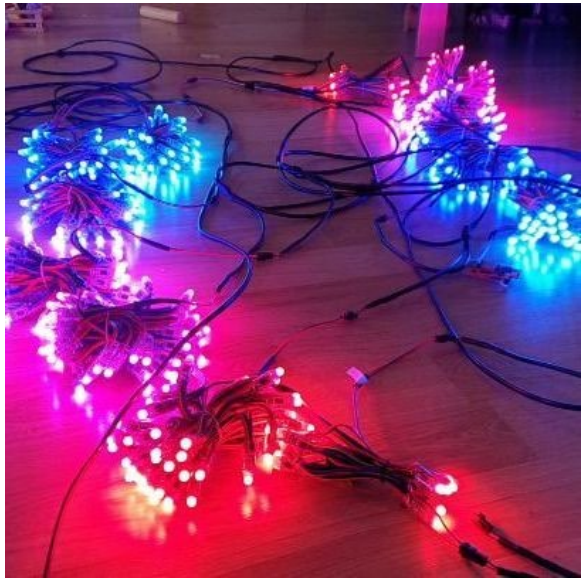
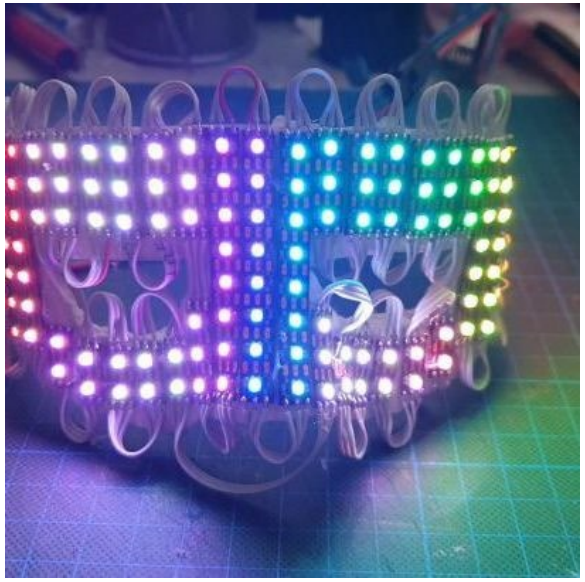
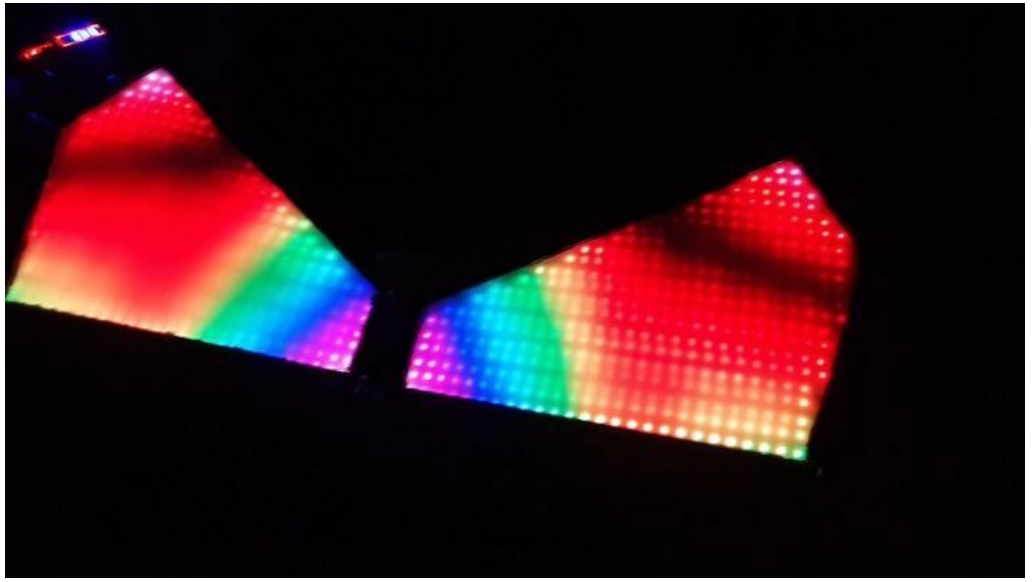
Chainable RGB LEDs

+



FastLED library

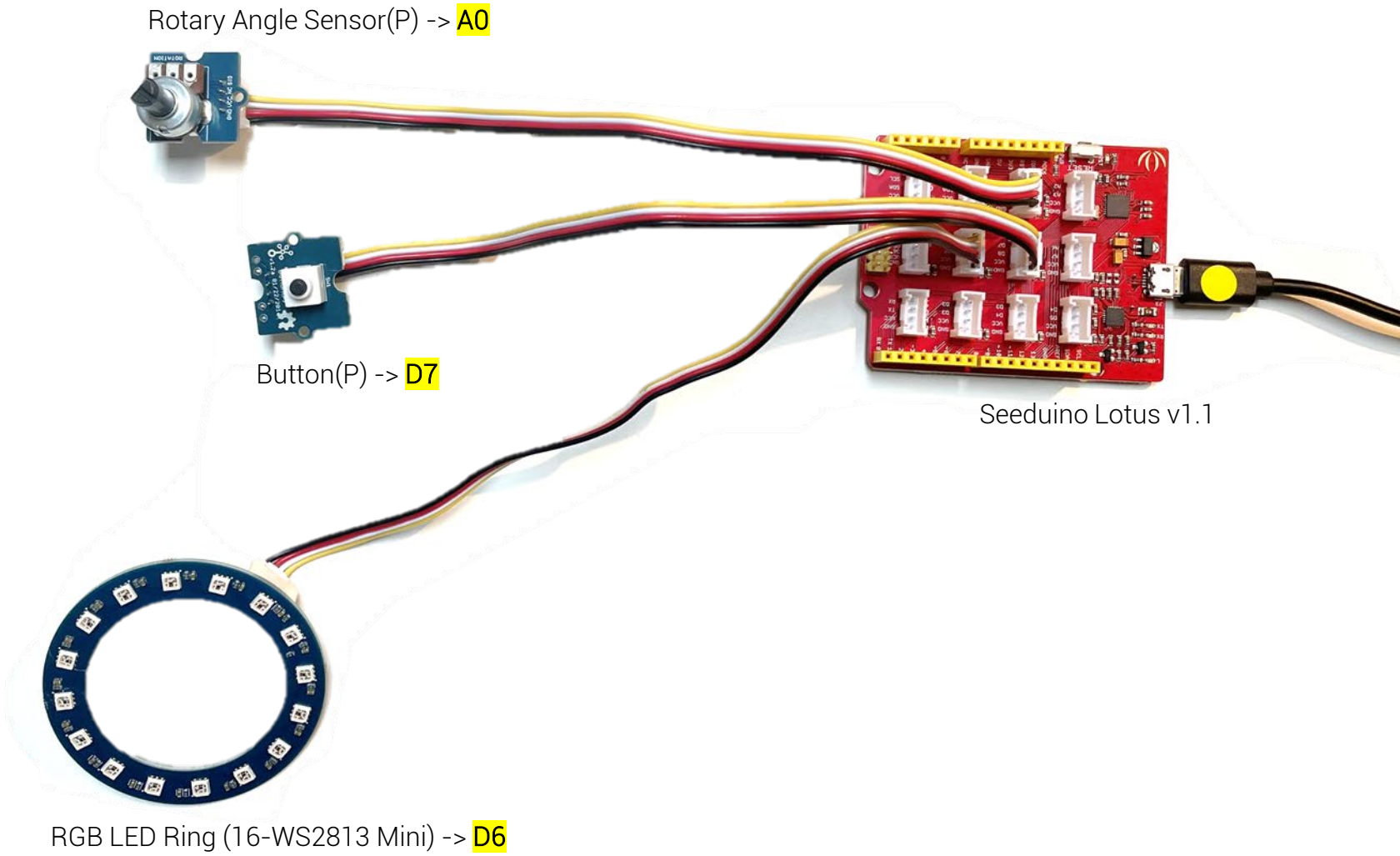


https://www.instagram.com/wow_elec_tron/

Setup

1. Add **FastLED** library by Daniel Garcia to Arduino IDE
2. Connect:
   - LED ring to D6
   - Push button to D7
   - Rotary Angle Sensor to A0



Github: https://github.com/interaction-technologies/IDE_Academy

Basic example

Include library

Define pin number and how many LEDs we have connected

Create an array of colours, one colour for each LED

Setup the LEDS, then turn off all LEDs

Loop through each LED and set its colour

Refresh LEDs to show new values

```
// Include FastLED library by Daniel Garcia
#include <FastLED.h>

// Setup FastLED pins
#define DATA_PIN 6
#define NUM_LEDS 16

// Array of led data
CRGB leds[NUM_LEDS];

void setup() {
   // Neopixel ring is WS2813 chipset, GRB ordering
   FastLED.addLeds<WS2813, DATA_PIN, GRB>(leds, NUM_LEDS);
   FastLED.clear();
}

void loop() {
   // Update colours for ring
   for(int i=0; i<NUM_LEDS; i++){
      leds[i] = CRGB::Green;
   }
   FastLED.show();
}
```

⚠ Always **first** set colours by editing the leds[] array, **then** display them using FastLED.show()

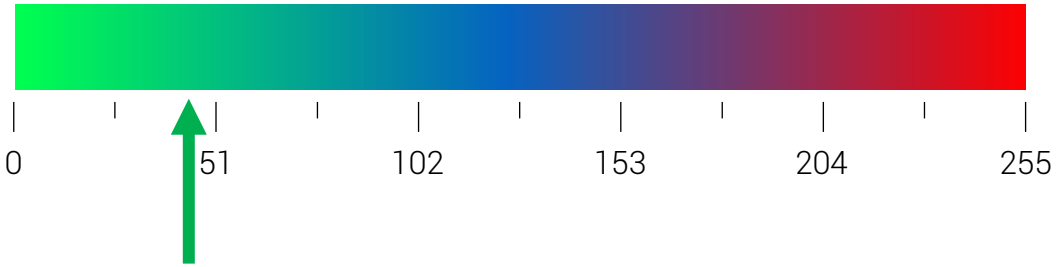
Picking colours

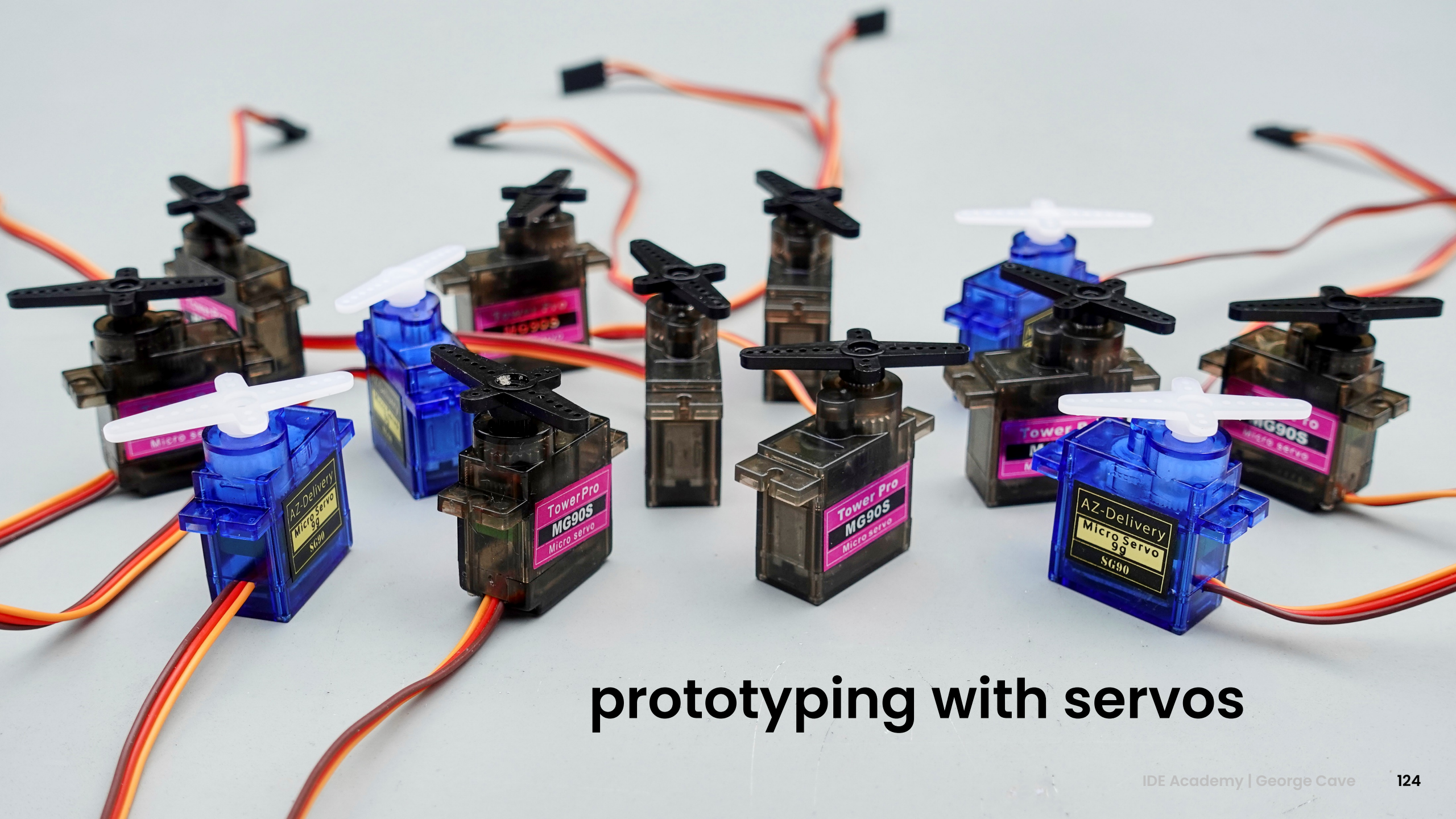
Single colours

```
leds[0] = CRGB::HotPink;

leds[0] = 0xFF44DD;

leds[0].setRGB(255,68,221);

CRGB my_pink = CRGB(255,68,221);
leds[0] = my_pink;
```

Gradients

```
// Define a gradient
CRGBPalette256 my_gradient = CRGBPalette256(
   CRGB::Green,
   CRGB::Blue,
   CRGB::Red
);

// Set LED 0 to colour at index 49
leds[0] = ColorFromPalette(my_gradient, 49);
```

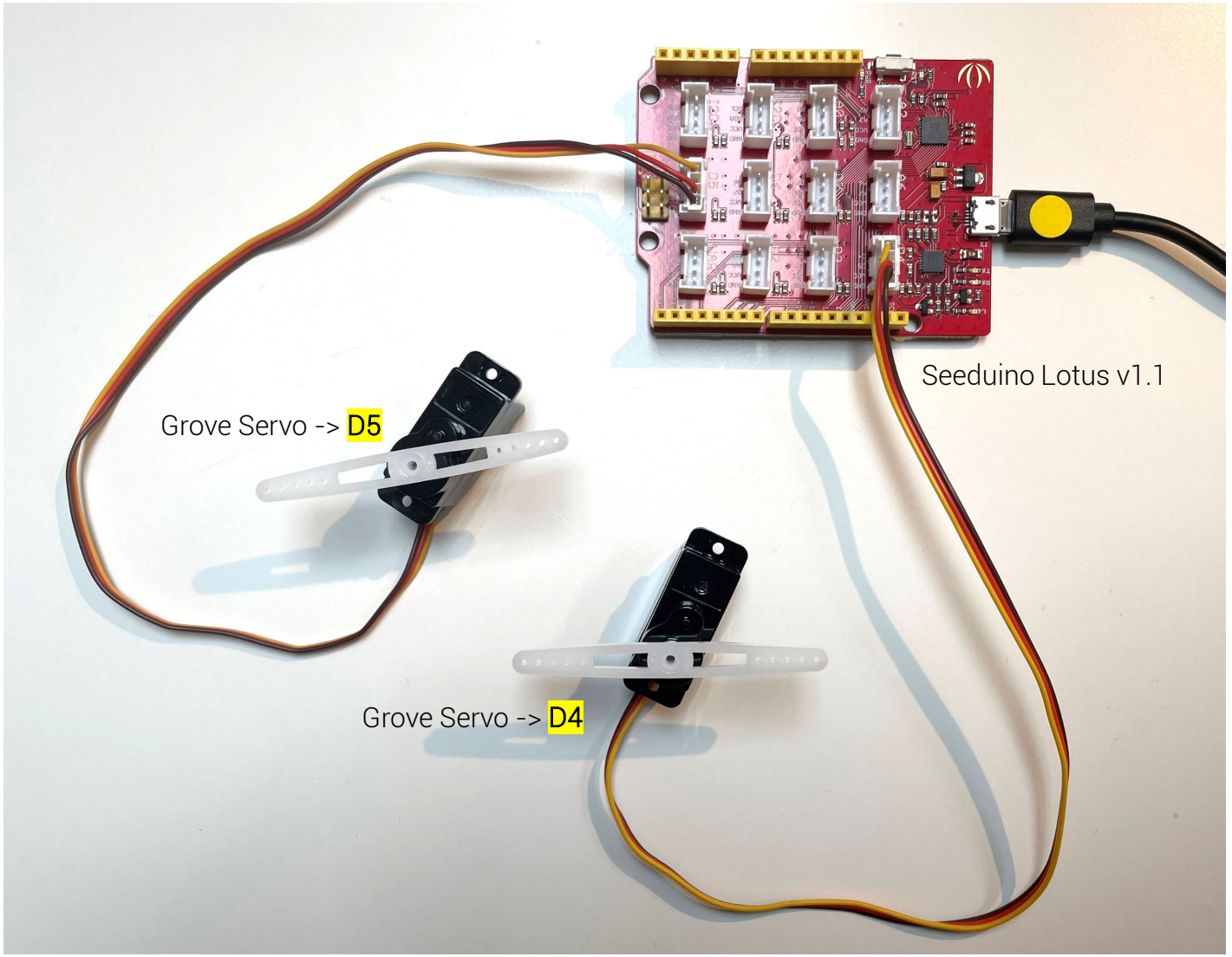




prototyping with servos

Setup

1. No library needed! (part of Arduino core)
2. Connect:
   - 1st Grove Servo to D4
   - 2nd Grove Servo to D5
3. Hint: Servo range is 0°->180°, but keep movement within range 10°->170°



Github: https://github.com/interaction-technologies/IDE_Academy

Basic example

Include libraries

Create Servo object

Attach motor to pin

Variables to keep track of current position, and how much to increase by each time

Set the new position!

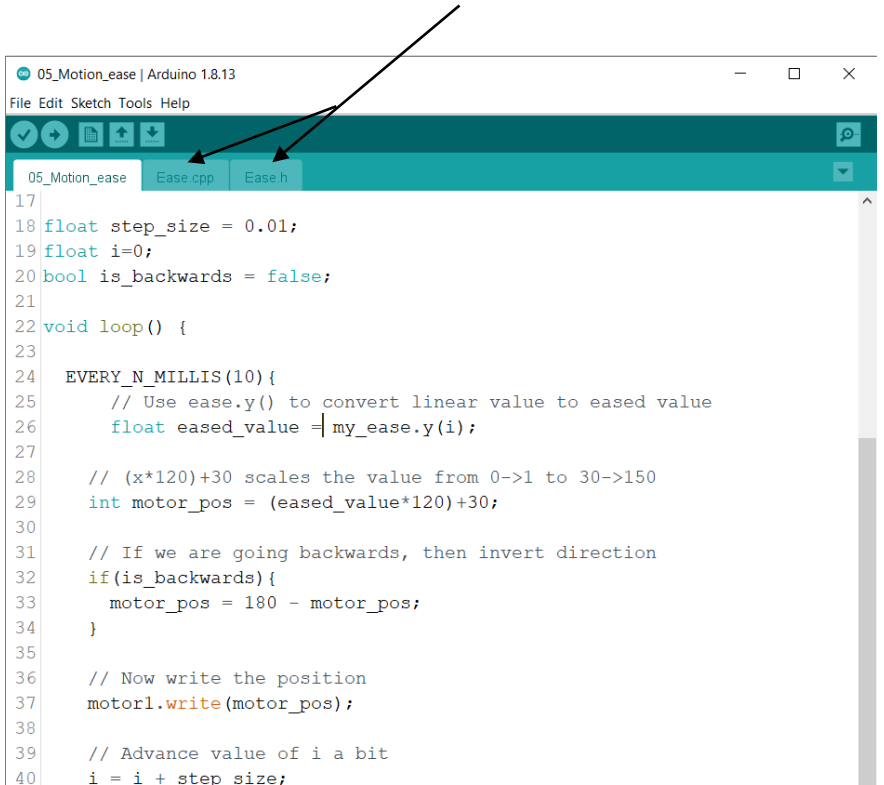
Reverse direction at ends

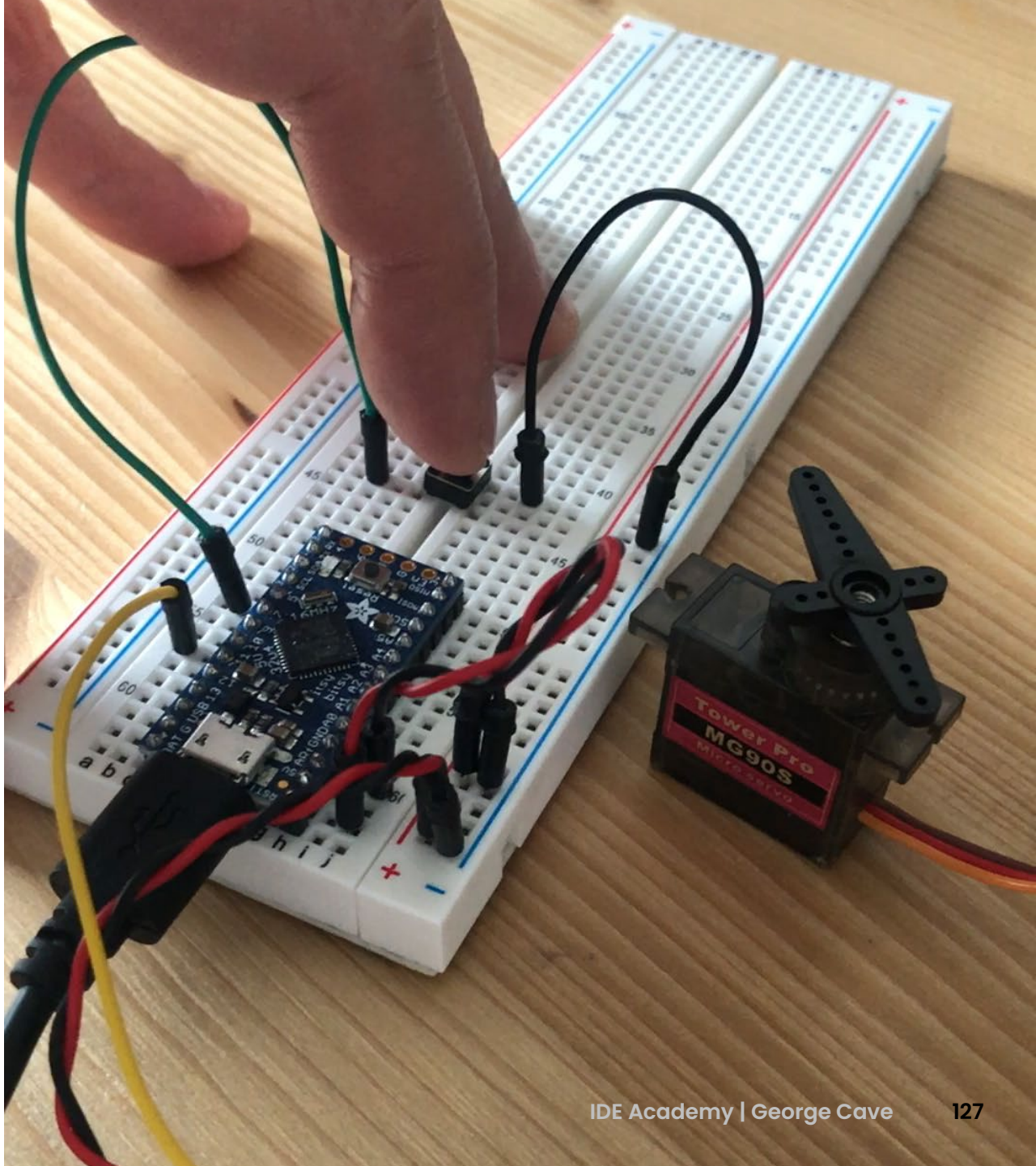
```
#include <Servo.h>
#include <FastLED.h>

// Create Servo object
Servo motor1;
const int motor1_pin = 5;

void setup() {
   motor1.attach(motor1_pin);
}

// Variables to save current positions and step sizes
int motor1_step = 1;
int motor1_pos = 50;

void loop() {
   EVERY_N_MILLIS(25){
      motor1.write(motor1_pos);

      if((motor1_pos >= 170)||(motor1_pos <= 10)){
         motor1_step = -motor1_step;
      }
      motor1_pos = motor1_pos + motor1_step;
   }
}
```

Adding Easing

This demo uses an easing library by George Cave 😉.

Rather than include a new library, the code is visible in the tabs in the project window.





Adding Easing

The local Ease class translates linear values into eased values using my_ease.y().

Try changing the easing type to BOUNCE_OUT or ELASTIC_OUT. Look in Ease.h to see some other examples.

Hint: If you use an easing like ELASTIC, it won't work for values close to the ends as you need to allow some space for the "elastic effect"

Include local Ease.h header

Create new Ease object

Use ease.y(x) to convert value

Scale value from 0->1 up to 30->150

```
#include <Servo.h>
#include <FastLED.h>
#include "Ease.h"

// Create Servo object
Servo motor1;
const int motor1_pin = 5;

// Create a new ease with QUARTIC_IN as the easing type
Ease my_ease(QUARTIC_IN);

void setup() {
  motor1.attach(motor1_pin);
}

float step_size = 0.01;
float i=0;

void loop() {

  EVERY_N_MILLIS(10){
    float eased_value = my_ease.y(i);

    int motor_pos = (eased_value*120)+30;
    motor1.write(motor_pos);

    i = i + step_size;
    if(i>=1){
      i=0; // reset counter
    }
  }
}
```

thank you


@interactionmagic


@George_Cave