ARDUINO WORKSHOP SERIES

Lesson plans for a three-session Arduino course, grades 5–8.

This is the Arduino course we run for middle schoolers, written up so your team can run it too. Students work in pairs and build something different in each session. The sessions don't depend on each other, so a student who only makes it to the second one can still finish the reaction game.

It's meant for grades 5–8. Each session takes 55–70 minutes, and group size is limited mostly by how many helpers you have. Nobody needs Arduino experience, and that includes whoever is teaching.

Most students have never used a breadboard. The first twenty minutes of session 1 will feel slow. By session 3 most pairs wire their circuits without asking for help.

Who brings what

We send parts for every pair (Arduino boards, breadboards, LEDs, buttons, buzzers, servos, and jumper wires, about $200 worth), the lesson plans in this packet, and a training call for whoever is teaching. A Spark Gears mentor is on call while you run each session.

The host provides a room with tables and outlets, recruits the students, and sends one or two adults or team members to the training call. Plan on an hour of setup before the first session.

Students bring a laptop. School Chromebooks work, and so do Windows and Mac laptops. Setup for each is below.

What each session covers

Session	Students build	New ideas
1. Traffic light	Three LEDs that cycle green, yellow, red	Digital outputs, resistors, breadboard rows, delay()
2. Reaction game	A game that times how fast you press a button	Reading a button, INPUT_PULLUP, random(), millis(), tone()
3. Servo creature	A cardboard creature with a moving part	Servos, analog input, map()

Laptop setup, a few days before session 1

Most first-day problems are laptop problems, not wiring problems. Test one laptop of every kind students will bring, and test it on the school's own network.

1. **Chromebooks.** Open **app.arduino.cc** in Chrome and sign in, or install the free Arduino Cloud for Chromebook app from Google Play. Nothing else needs to be installed. Plug in the board, click Detect Device, choose its

port, and upload.

2. **Windows and Mac.** Install Arduino IDE 2 from **arduino.cc/en/software**. The browser editor also works on these laptops, but it needs Arduino's Cloud Agent installed first.
3. On every laptop, upload the Blink example (File, Examples, 01.Basics, Blink). If the small on-board LED starts blinking, that laptop is ready.
4. Count parts into one bag per pair and label the bags. Cleanup goes much faster.
5. If students are under 13, ask the school how it handles Arduino accounts. Some teachers sign in once and share one account across the room.

Chromebook boards. Arduino only supports uploading from a Chromebook to some boards. The Uno R3 and Uno R4 are on the list, and they're what we send. Many inexpensive Uno clones won't upload from a Chromebook even though they work fine on a Windows laptop.

Serial Monitor. Some schools have found the Serial Monitor stays blank on their Chromebooks. Session 2 prints reaction times there. The buzzer still gives feedback without it, so the game works either way, but check it during your test.

> **Tip from our first workshops**
>
> Have one spare laptop already set up and tested. When a student's computer refuses to see the board, swap it in and keep them building instead of debugging drivers in front of the class.

Session 1 · 55 minutes

Traffic light

Pairs wire a red, a yellow, and a green LED and program them to cycle like a stoplight. Plan for wiring to take most of the period. Once the first LED lights up, the other two usually take a few minutes. The code is short, so there's time to experiment with the timing at the end.

Parts per pair

Part	Qty	Notes
Arduino Uno + USB cable	1	Make sure the cable carries data. Some only charge.
Half-size breadboard	1	
LEDs (red, yellow, green)	1 each	The longer leg is positive
220-ohm resistors	3	Bands are red, red, brown
Male-to-male jumper wires	6–8	Extras for the challenge

Timing

Time	What happens	Notes for the teacher
0–8 min	Plug in, choose the board and port, upload Blink	Walk the room. Every pair should see the on-board LED blink before moving on.
8–25 min	Wire the three LEDs	Show how breadboard rows connect. Each LED goes pin, resistor, long leg. The short leg goes to GND.
25–50 min	Code the sequence and adjust the timing	Start with green only, then add yellow and red. Let pairs choose their own timing.
50–55 min	Show a few working lights, then clean up	Parts back in the bag. Count the LEDs.

Wiring

LED	Pin	Path
Red	8	Pin 8 to a 220-ohm resistor to the LED's long leg. Short leg to the GND rail.
Yellow	9	Pin 9 to a 220-ohm resistor to the LED's long leg. Short leg to the GND rail.
Green	10	Pin 10 to a 220-ohm resistor to the LED's long leg. Short leg to the GND rail.
GND rail	GND	One wire from any GND pin to the blue rail on the breadboard.

Starter code

```
// Traffic light: green, then yellow, then red, forever
const int RED = 8;
const int YELLOW = 9;
const int GREEN = 10;

void setup() {
  pinMode(RED, OUTPUT);
  pinMode(YELLOW, OUTPUT);
  pinMode(GREEN, OUTPUT);
}

void loop() {
  digitalWrite(GREEN, HIGH);   // go
  delay(4000);
  digitalWrite(GREEN, LOW);

  digitalWrite(YELLOW, HIGH);  // slow down
  delay(1500);
  digitalWrite(YELLOW, LOW);

  digitalWrite(RED, HIGH);     // stop
  delay(4000);
  digitalWrite(RED, LOW);
}
```

If it doesn't work

- **An LED stays dark.** It's probably in backwards. Turn it around so the long leg faces the resistor.
- **Upload fails.** Check the port, then try a different USB cable. Charge-only cables are common.
- **Two LEDs light at once.** A wire or resistor is sharing a breadboard row with the wrong LED.

For pairs who finish early

- Add a second set of LEDs for the cross street. When one direction is green, the other has to be red.
- Add a white walk light that only comes on while both directions are red.
- Make the yellow light blink three times before it turns red.

Session 2 · 55 minutes

Reaction-time game

The LED comes on after a random wait and you hit the button as fast as you can. The Arduino times the press and plays a high beep if you were quick, a low one if you weren't. Reading a button is the new skill this week. Leave time at the end, because most pairs keep playing once it works.

Parts per pair

Part	Qty	Notes
Arduino Uno + USB cable	1	
Half-size breadboard	1	
LED and 220-ohm resistor	1 each	Any color
Four-leg tactile button	1	No resistor needed, see below
Passive piezo buzzer	1	Passive buzzers can play different pitches
Jumper wires	6	

Timing

Time	What happens	Notes for the teacher
0–8 min	Plug in and upload Blink as a test	New students may need help finding the port.
8–18 min	Wire the LED to pin 9	Pin 9, resistor, long leg. Short leg to GND.
18–30 min	Wire the button to pin 2 and GND	The button straddles the center gap. One leg to pin 2, the diagonal leg to GND.
30–50 min	Write the game and test it	Open the Serial Monitor at 9600 baud to see the times.
50–55 min	Class tournament, then clean up	Fastest reaction in the room wins bragging rights.

Why the button has no resistor. INPUT_PULLUP switches on a resistor inside the Arduino. The pin reads HIGH until the button is pressed, then reads LOW. Students find this backwards at first. Say it out loud a couple of times. Pressed means LOW.

Starter code

```
// Reaction game: wait for the light, then press as fast as you can
const int LED = 9;
const int BUTTON = 2;
const int BUZZER = 8;

void setup() {
  pinMode(LED, OUTPUT);
  pinMode(BUTTON, INPUT_PULLUP);   // pressed = LOW
  pinMode(BUZZER, OUTPUT);
  Serial.begin(9600);
  randomSeed(analogRead(A0));      // different random waits every time
}

void loop() {
  Serial.println("Get ready...");
  delay(random(2000, 5000));       // wait 2 to 5 seconds

  digitalWrite(LED, HIGH);
  unsigned long start = millis();
  while (digitalRead(BUTTON) == HIGH) {
    // wait here until the button is pressed
  }
  unsigned long reaction = millis() - start;
  digitalWrite(LED, LOW);

  Serial.print("Your time: ");
  Serial.print(reaction);
  Serial.println(" ms");

  if (reaction < 300) {
    tone(BUZZER, 1200, 300);       // fast: high tone
  } else {
    tone(BUZZER, 500, 300);        // slow: low tone
  }
  delay(2000);
}
```

If it doesn't work

- **The game ends instantly with a time of 0.** The button is turned the wrong way, so it always reads as pressed. Rotate it 90 degrees.
- **No sound.** Check that the buzzer's + leg goes to pin 8. An active buzzer plays only one pitch, which is fine for this.
- **The Serial Monitor shows garbage.** Set the baud rate to 9600.

Changes to try

- Play three rounds and print the average.
- Add a false-start penalty. If the button is already down when the LED comes on, buzz and restart.
- Shrink the 300 ms target a little every round.
- Two-player mode with a second button on pin 3. First press wins.

Session 3 · 70 minutes

Servo creature

Half of this session is cardboard and tape. Each pair builds a creature around a micro servo and then decides how it should move, using either a button that steps through poses or a knob for smooth control. It runs longer than the other two because the building takes a while, and nobody wants to stop.

Parts per pair

Part	Qty	Notes
Arduino Uno + USB cable	1	
Half-size breadboard	1	
SG90 micro servo with horns	1	Brown is GND, red is 5V, orange is signal
Button or 10k potentiometer	1	Each pair picks one
Jumper wires	6	
Cardboard, foam, tape, markers	Shared	Set up a craft table

Timing

Time	What happens	Notes for the teacher
0–10 min	Upload the Sweep example (File, Examples, Servo, Sweep)	This proves the servo works before anyone writes code.
10–25 min	Wire the SG90	Brown to GND, red to 5V, orange to pin 9.
25–35 min	Pick a control and wire it	Button to pin 2 and GND. For the knob, middle leg to A0 and the outer legs to 5V and GND.
35–50 min	Code the movement	The button steps through poses. The knob sets the angle directly.
50–70 min	Build the creature and tune it	Tape the servo horn to a jaw or a wing. Adjust angles until it looks alive.

Starter code, button version

```
// Button version: each press moves the creature to its next pose
#include <Servo.h>

Servo creature;
const int BUTTON = 2;
int poses[] = {20, 90, 160};   // angles in degrees, change these
int pose = 0;

void setup() {
  creature.attach(9);
  pinMode(BUTTON, INPUT_PULLUP);
  creature.write(poses[pose]);
}

void loop() {
  if (digitalRead(BUTTON) == LOW) {
    pose = (pose + 1) % 3;           // 0, 1, 2, back to 0
    creature.write(poses[pose]);
    delay(250);
    while (digitalRead(BUTTON) == LOW) { }   // wait for release
  }
}
```

Starter code, knob version

```
// Knob version: turn the potentiometer, the creature follows
#include <Servo.h>

Servo creature;

void setup() {
  creature.attach(9);
}

void loop() {
  int knob = analogRead(A0);              // 0 to 1023
  int angle = map(knob, 0, 1023, 0, 180); // convert to 0 to 180 degrees
  creature.write(angle);
  delay(15);
}
```

If it doesn't work

- **The servo twitches or the Arduino keeps resetting.** The servo is drawing too much power. Make sure nothing is jamming the horn, and run only one servo from a laptop's USB port.
- **The servo buzzes at the end of its swing.** It's being pushed past its limit. Keep angles between about 10 and 170.
- **The knob only works for part of its turn.** An outer leg isn't connected, or the outer legs are swapped.

Creature ideas

- A crocodile whose jaw snaps shut on every button press.
- A bird that flaps faster as the knob turns. Change the delay, not the angle.

Logistics and safety

Grades	5–8. Younger students can do sessions 1 and 3 with more adult help.
Group size	As many pairs as you have helpers for. One helper for every five or six pairs works for us.
Length	55–70 minutes each. Run them on three days in a row or once a week.
Safety	Everything runs on 5 volts from USB. There's no soldering and no power tools. Remind students not to bend LED legs back and forth, because they snap.
Cleanup	Every part goes back in its labeled bag. Count the LEDs and servos before students leave.

Questions? Email **sparkgearsinitiative@gmail.com** or call **571-429-9220**. You can request a mentor or a supply kit at sparkgearsinitiative.org/for-teams.html.