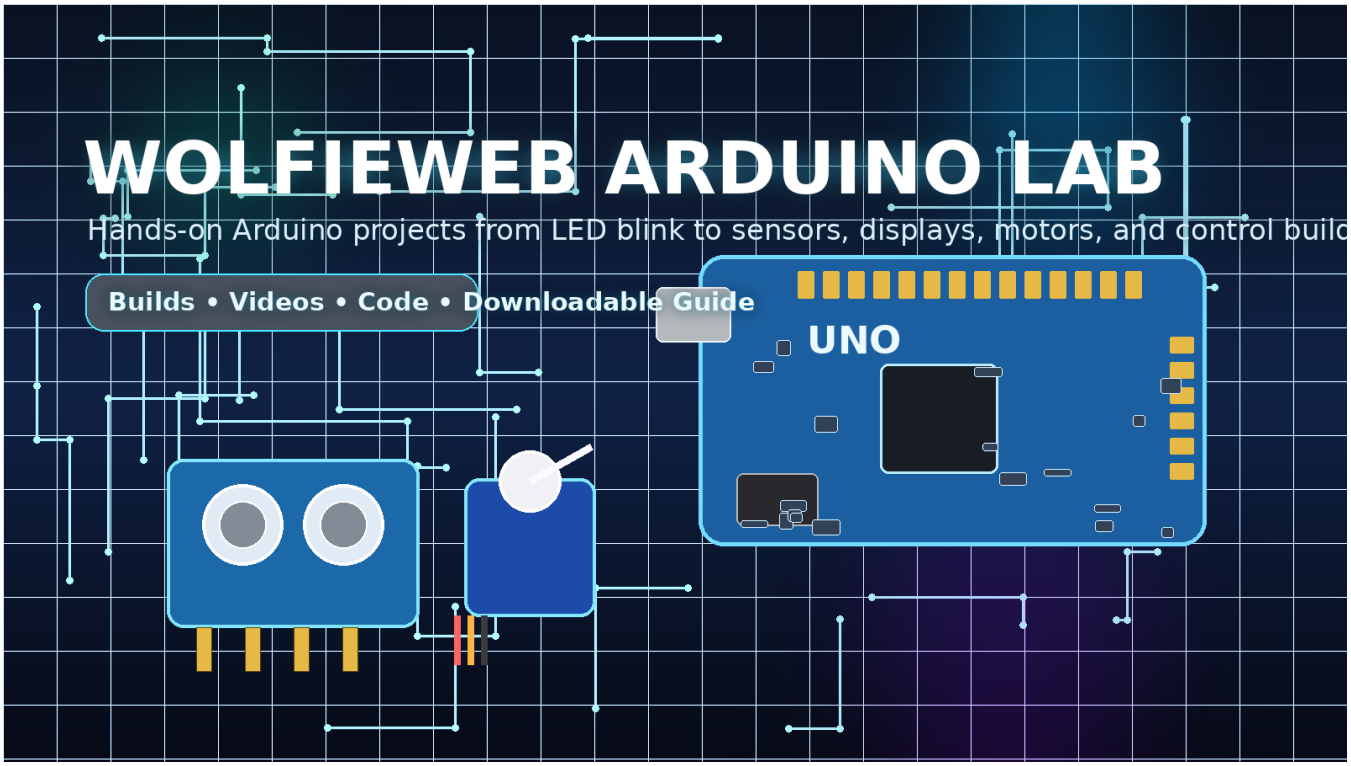
WolfieWeb Arduino Project Guide

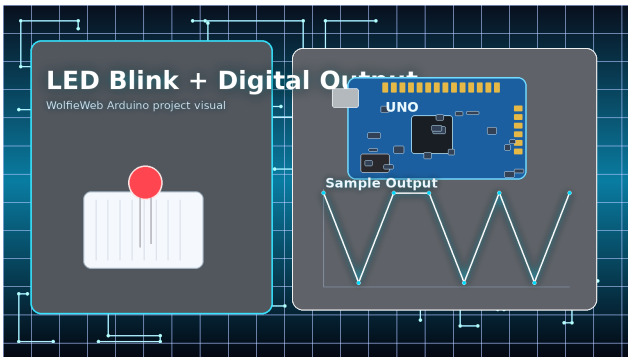
A printable Arduino starter pack with project visuals, wiring references, practical code, and troubleshooting notes.

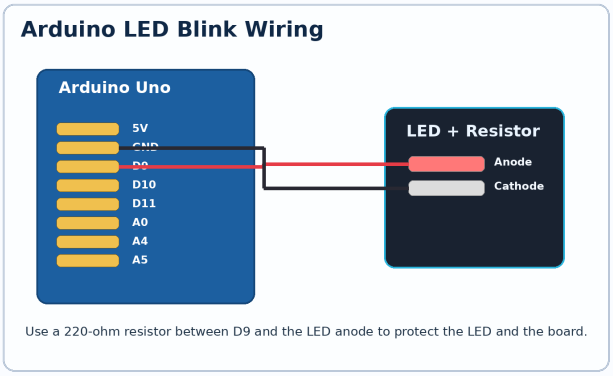


What this pack includes	High-res page graphics, working embedded-video layout, and a downloadable PDF with Ardui
Best beginner path	Start with LED Blink, then Ultrasonic Sensor, then Servo Sweep, then LCD Display, then Pote
What beginners usually get wrong	Wrong pin mapping, missing grounds, using motors that draw too much current from the board

1. LED Blink + Digital Output

The first Arduino win. It teaches digital output, safe LED wiring, and how a simple sketch controls hardware.





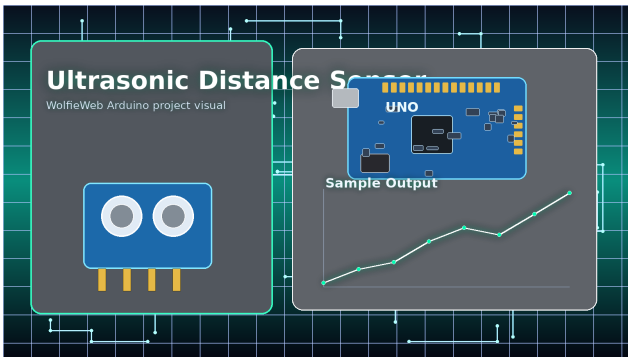
Starter code

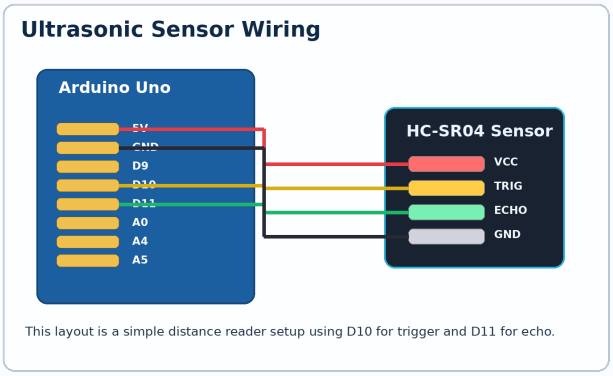
```
void setup() {
  pinMode(9, OUTPUT);
}

void loop() {
  digitalWrite(9, HIGH);
  delay(1000);
  digitalWrite(9, LOW);
  delay(1000);
}
```

Troubleshooting: If the LED does not light, flip the LED orientation, verify the resistor is in series, and make sure the wire is really on D9 and not the wrong row.

2. Ultrasonic Distance Sensor

This project turns the Arduino into a distance reader. It is useful for obstacle sensing, parking alarms, and robot builds.





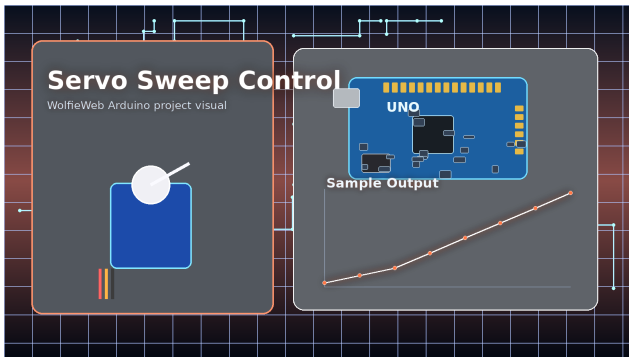
Starter code

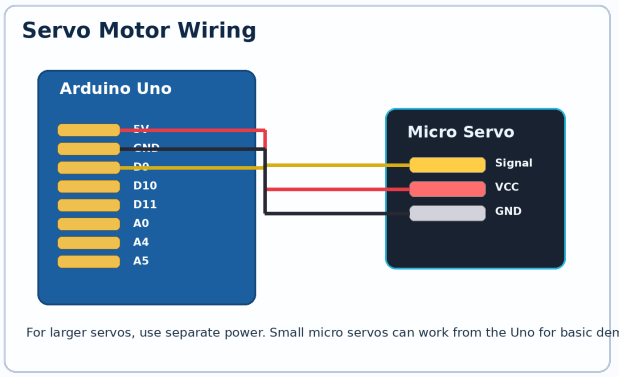
```
const int trigPin = 10;
const int echoPin = 11;

void setup() {
  Serial.begin(9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
}

void loop() {
  long duration;
  float distance;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = duration * 0.034 / 2;
  Serial.println(distance);
  delay(300);
}
```

Troubleshooting: If readings are erratic, recheck trigger and echo pins, avoid loose jumper wires, and test with a flat object in front of the sensor.

3. Servo Sweep Control

A strong motion-control project. It teaches PWM-driven servo positioning and opens the door to robotic arms and moving mechanisms.





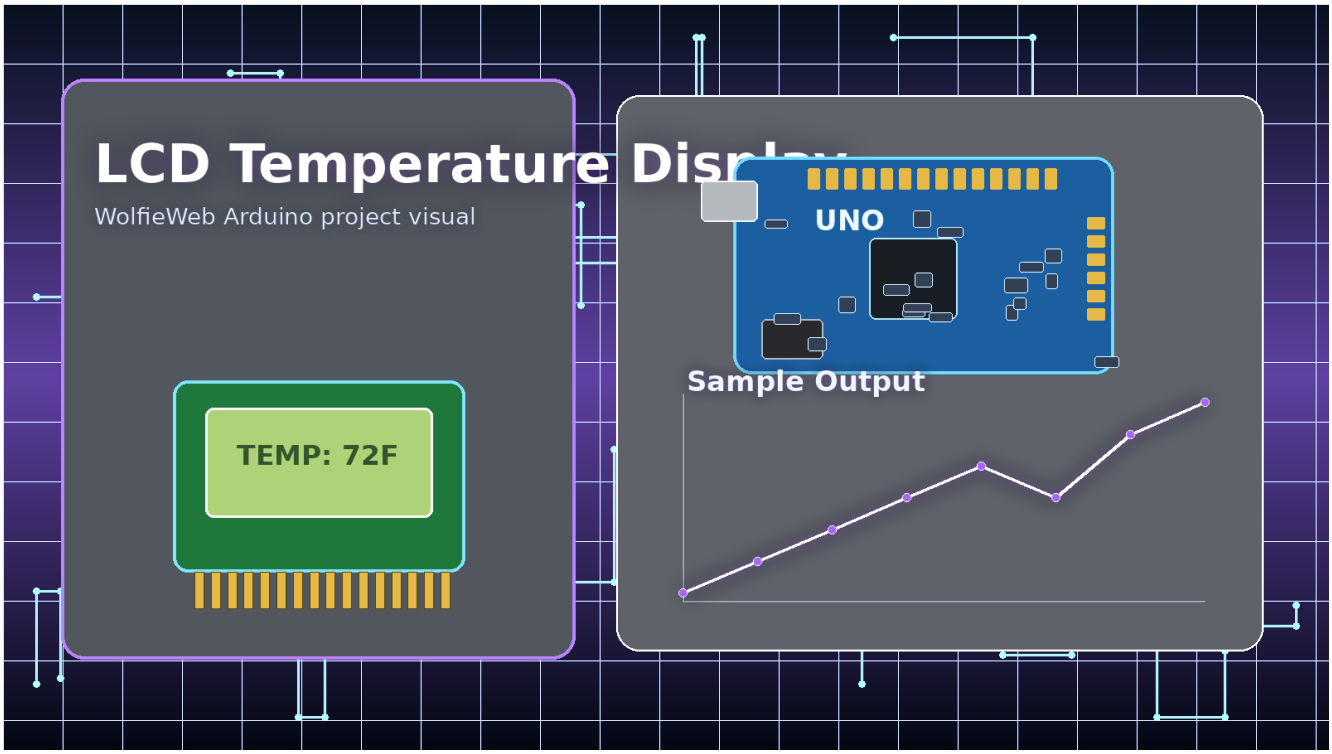
Starter code

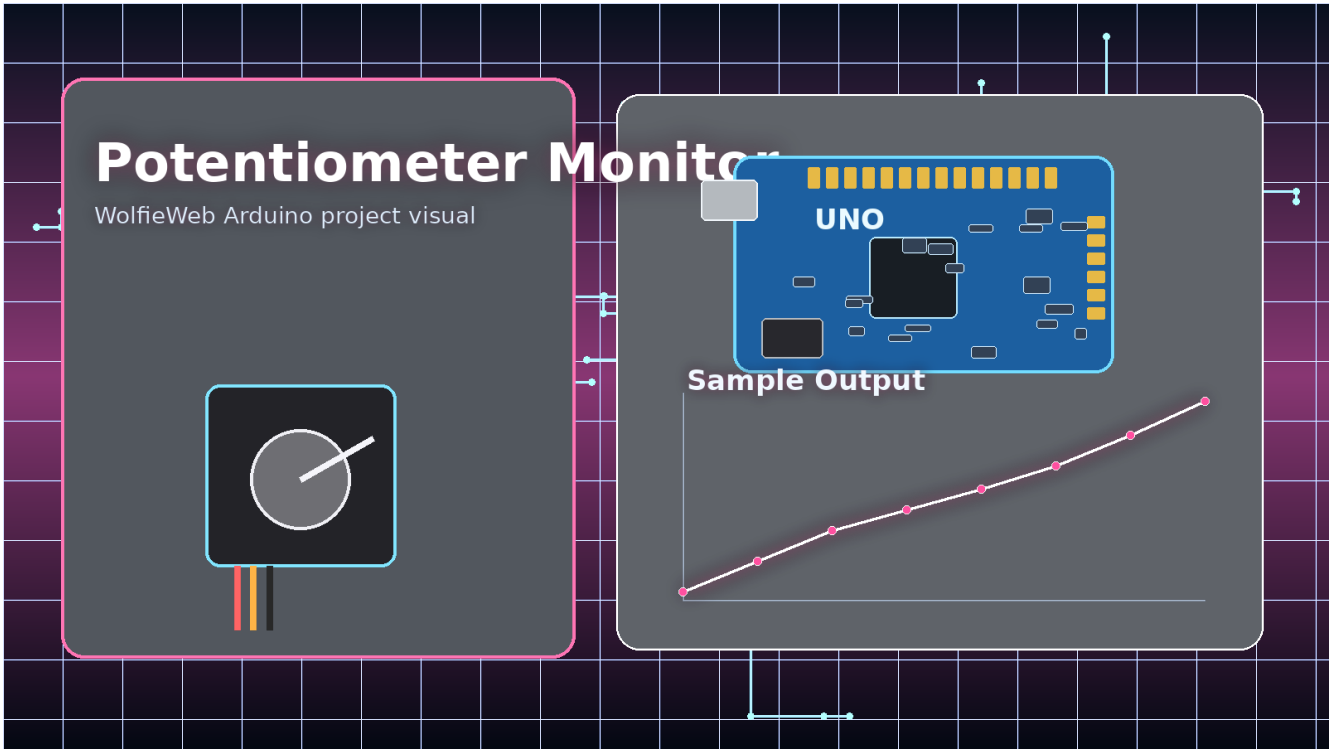
```
#include <Servo.h>

Servo myServo;

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

void loop() {
  for (int pos = 0; pos <= 180; pos++) {
    myServo.write(pos);
    delay(15);
  }
  for (int pos = 180; pos >= 0; pos--) {
    myServo.write(pos);
    delay(15);
  }
}
```

Troubleshooting: If the servo jitters, use a stable power source and keep the ground shared. Cheap servos often shake when power is weak.

LCD and Analog Input Notes

The LCD project shows how Arduino can present live values in a clean display instead of only sending them to the serial monitor. The potentiometer monitor is the easiest way to understand analog input, because you can turn the knob and instantly see values change. Those two builds make a good bridge between basic starter projects and more advanced Arduino systems.





1	Install Arduino IDE	Set board type and COM port before uploading
2	Map the pins first	Write down digital, analog, power, and ground connections
3	Protect LEDs	Use a resistor instead of wiring LEDs directly
4	Power motors carefully	Do not overload the Uno with large servo or motor current
5	Test one part at a time	Build in small steps instead of guessing