20-Nov
Friday

Embedded AI & Robotics

Who Am I & What This Is

- I’m **Sahil**, Master of AI in Business & your TA
- My world: **AI + Robotics + Embedded Systems**
- **Contact me through whatsapp**
- This course: learn how **data & models** can control **real hardware**
- Very **hands-on**: less talking, more building

Quick Recap (Week 1)

What we did

Plugged in Arduino
Installed Arduino IDE
Uploaded Blink
Used:

- **Button** as input
- **LED** as output

What confused you the most?
The wiring?
Where GND goes?
Why the resistor?
Digital pins vs 5V?

????

Why Are We Learning Basic Electricity?

So you know:

- **Where** to put the resistor – and **why**
- Why we don’t just use **5V** all the time
- What these pin labels mean (0, 1, ~3, A0, SDA, SCL, RX, TX)

So you can:

- Read simple **schematics**
- Wire safely without guessing
- Understand what’s happening when things **don’t work**

Circuits & Ground

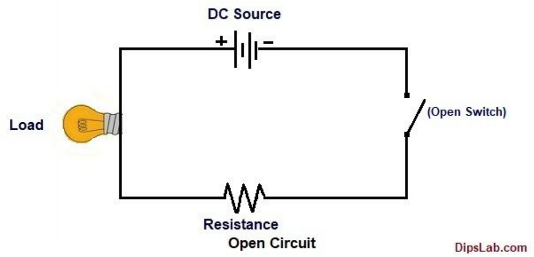
What Is a Circuit?

A circuit is a **closed loop**:
`5V → resistor → LED → GND`

-
- If any part of the loop is open → no current → nothing happens
- If loop is “too easy” (short circuit) → too much current → bad

Ground (GND):

- Our **0V reference**
- All voltages are measured *relative* to GND
- All boards, sensors, servos must share **the same GND**



Why Does the LED Need a Resistor?

If you connect:
`5V → LED → GND`

- → too much current can flow
  → LED can burn, pin can be stressed

We add a **resistor in series**:
`Pin/5V → resistor → LED → GND`

- In a series loop, **same current** flows through all parts
  - Resistor can be **before or after** the LED
  - Important thing: there is **one resistor in series**

What Is a Switch?

- A **switch** either:
    - Connects the circuit (ON, closed)
    - Breaks the circuit (OFF, open)
- Push button in Week 1 was a **mechanical switch**:
    - Press → completes the path → current flows
    - Release → opens path → no current

Used for:

- Turning LEDs on/off
- Selecting modes
- Providing manual input

Switches

Digital Pins as Electronic Switches

Digital Pins = Tiny Built-In SwitchesDigital output pin can be:

  - `LOW` → almost 0V → OFF
  - `HIGH` → ~5V → is ON
- Instead of you pressing a button:'The **code** opens/closes the switch

Example:

- `pinMode(9, OUTPUT);`
- `digitalWrite(9, HIGH); // switch ON (5V)`
- `digitalWrite(9, LOW);  // switch OFF (0V)`

Why Do We Need Transistors?

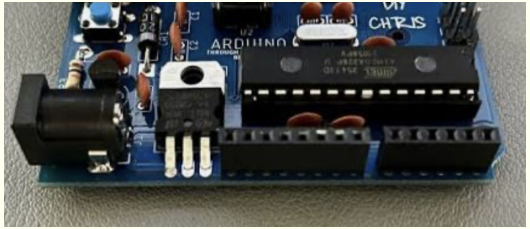
The Limits of a GPIO Pin

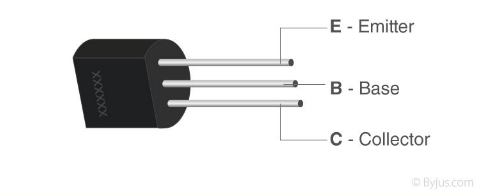
- A digital pin can only supply a **small current** (≈ 20–30 mA)
- Fine for:
  - LEDs
  - Small signals
- NOT fine for:
  - Motors, fans, pumps
  - Bigger loads

Transistor as an Electronic Switch

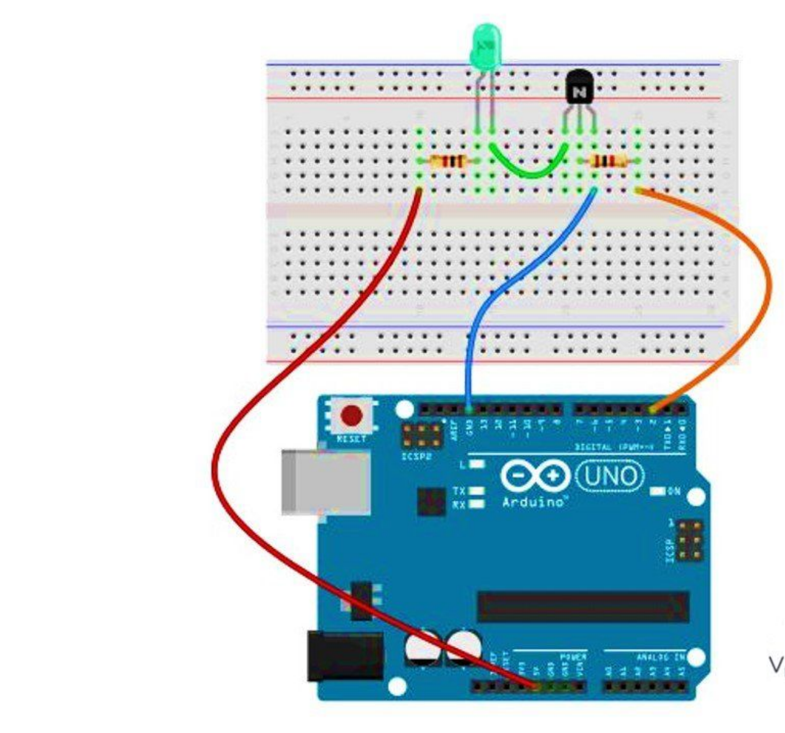
High-level view :

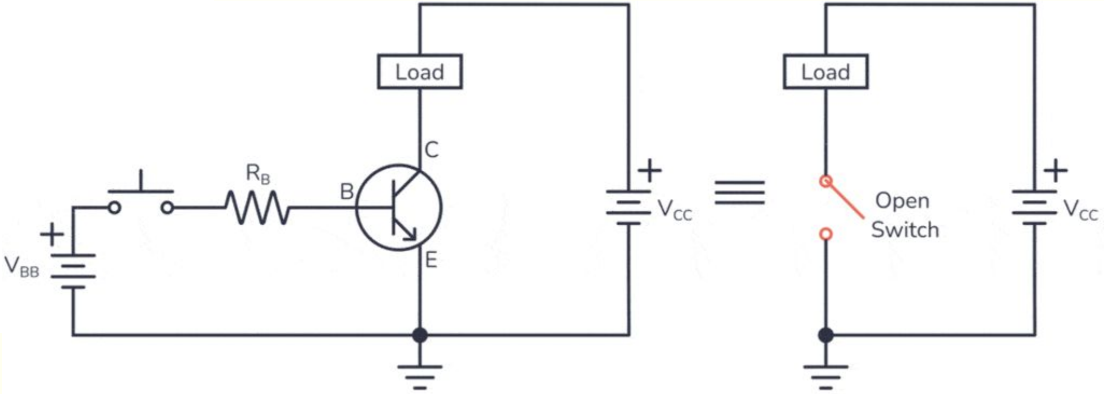
- Transistor has two "sides":
  - **Load side**: where the higher current flows (e.g., motor → GND)
  - **Control side**: tiny signal from Arduino (e.g., pin 9)
- Arduino **does not** directly power the motor:
  - It tells the transistor when to **connect** or **disconnect** the motor from power



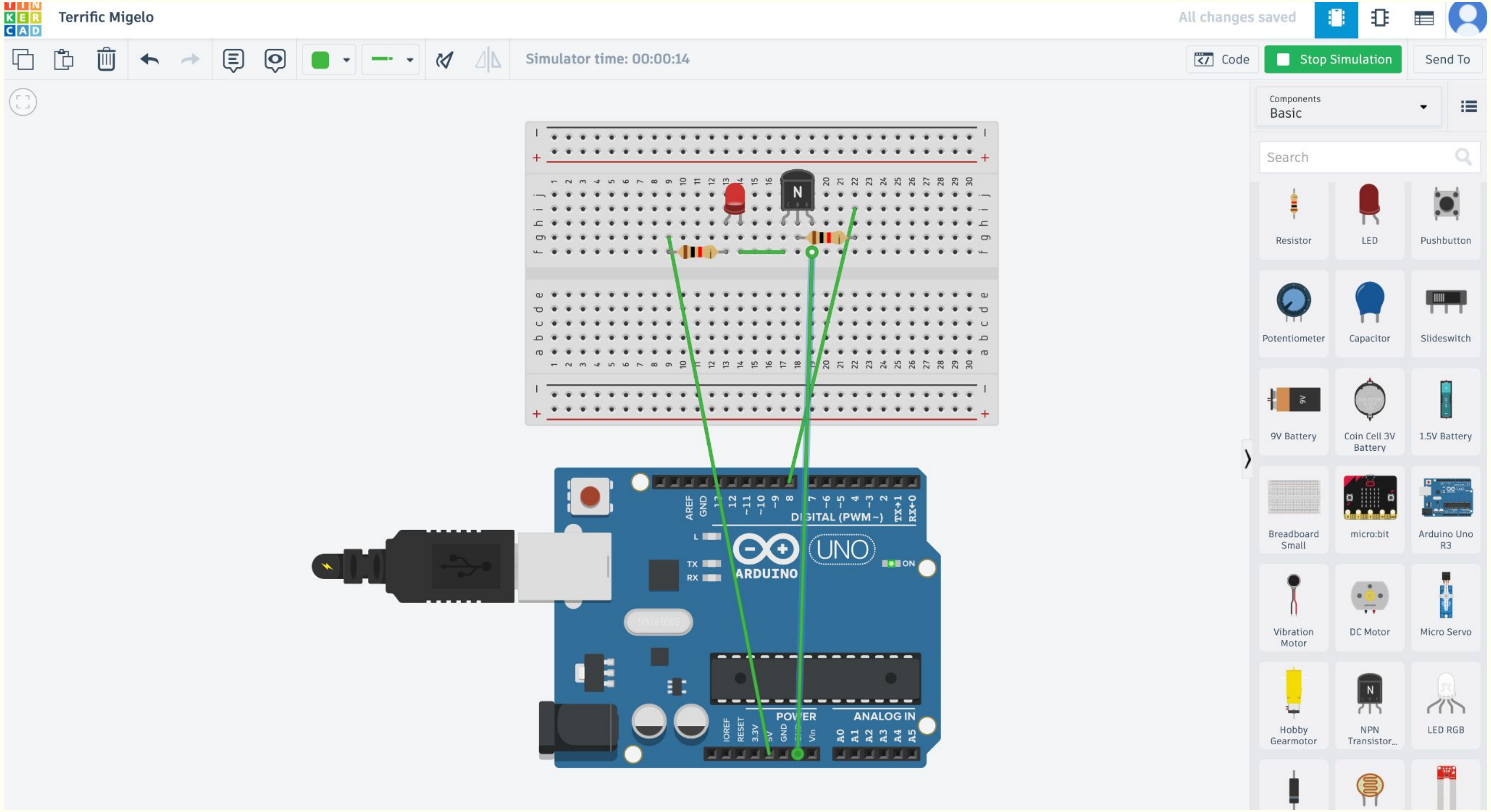


© Byjus.com


RESET
ICSP2
DIGITAL (PWM~)
UNO
Arduino
TX
RX
ON
ICSP
POWER
ANALOG IN


Load
C
B
E
R_B
+
V_{BB}
+
V_{CC}
≡
Load
Open
Switch
+
V_{CC}

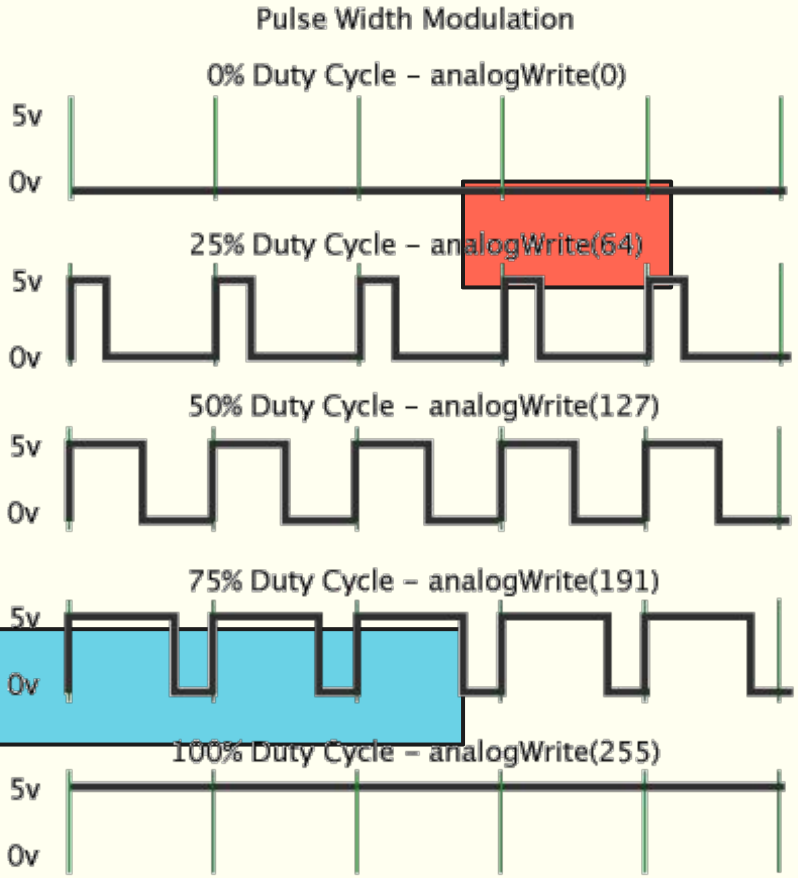
TinkerCad


Terrific Migelo
All changes saved
Simulator time: 00:00:14
Code
Stop Simulation
Send To
Components
Basic
Search
Resistor
LED
Pushbutton
Potentiometer
Capacitor
Slideswitch
9V Battery
Coin Cell 3V Battery
1.5V Battery
Breadboard Small
micro:bit
Arduino Uno R3
Vibration Motor
DC Motor
Micro Servo
Hobby Gearmotor
NPN Transistor...
LED RGB
N
ARDUINO
UNO
DIGITAL (PWM~)
POWER
ANALOG IN

PWM: Why Not Just HIGH or LOW?

Why Do We Need PWM?

- Digital pins are only:
  - `LOW` (0V)
  - `HIGH` (~5V)
- But real world often needs **in between**:
  - Half brightness LED
  - Medium motor speed
- **PWM (Pulse Width Modulation)**:
  - Turns the pin ON and OFF **very fast**
  - Controls the **percentage of time ON** (duty cycle)
  - 0% duty → always OFF
  - 50% duty → feels like "half power"
  - 100% duty → always ON


Pulse Width Modulation
0% Duty Cycle – analogWrite(0)
5v
0v
25% Duty Cycle – analogWrite(64)
5v
0v
50% Duty Cycle – analogWrite(127)
5v
0v
75% Duty Cycle – analogWrite(191)
5v
0v
100% Duty Cycle – analogWrite(255)
5v
0v

PWM Pins on Arduino Uno**

- Marked with ~:
  `~3, ~5, ~6, ~9, ~10, ~11`

RX/TX – Talking to the Computer

Pins **0** and **1** = **Serial** UART pins:

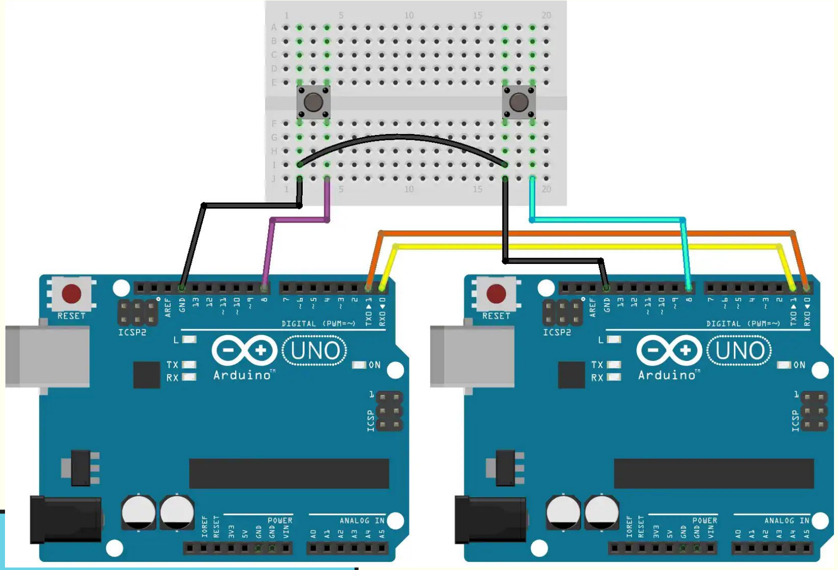
  - **RX (0)** – Receive data
  - **TX (1)** – Transmit data
- Used by:
  - USB connection to your laptop
  - **Serial Monitor** in Arduino IDE

Why it matters:

When you do:

```
Serial.begin(9600);
Serial.println("Hello");
```

- the board is using **RX/TX** under the hood.
- That's why we **avoid using pins 0 and 1** for:
  - LEDs
  - Buttons
  - Other random stuff
    → they can interfere with uploads / Serial communicatio



SDA / SCL and I^{2}C

I^{2}C (“eye-squared-see”) is a **two-wire communication bus** commonly used for:

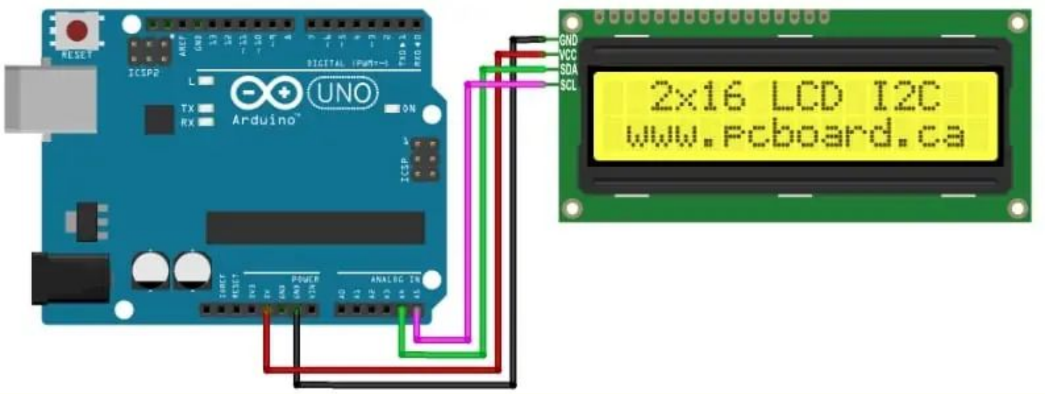
- Sensors (IMUs, temperature/humidity, light)
- Small displays (OLED, some LCDs)
- Other smart peripherals

On Arduino Uno:

- **SDA (data line)** → A4
- **SCL (clock line)** → A5

Key features:

- Only **two wires** (plus GND) for many devices
- Each device has an **address**
- Arduino acts as **controller/master**:
  - Sends clock on SCL
  - Sends/receives data on SDA



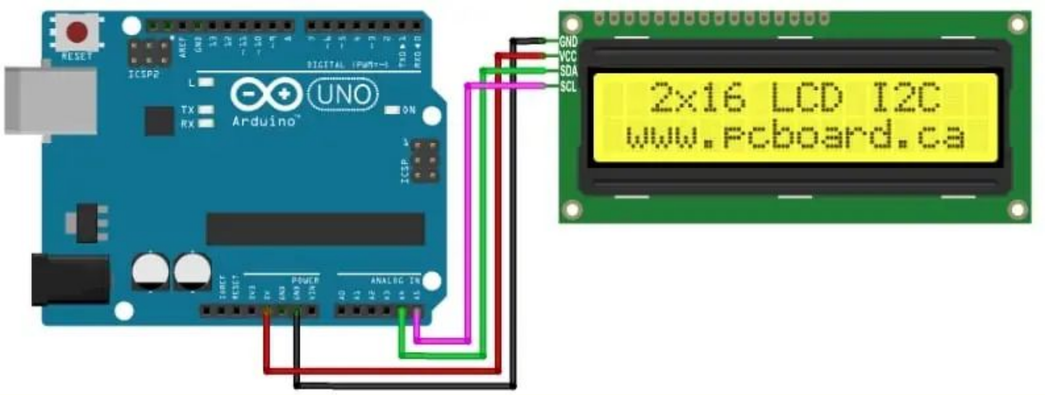
Why You Should Care About I²C as a Data Scientist

As data science students doing embedded AI:

- Real-world AI projects use:
    - IMUs (motion sensors)
    - Environment sensors (temp, humidity, gas)
    - Tiny displays for UI
- Most of these talk via **I²C** over SDA/SCL

So:

- SDA/SCL are how your microcontroller **collects data** from multiple sensors
- Later weeks:
    - You'll treat I²C sensors as **data sources** for:
        - Tiny ML models
        - Anomaly detection
        - Gesture and motion recognition



Putting It Together: Digital, PWM, UART, I^{2}C

- **Digital / PWM**:
  - LEDs, buttons, simple sensors
  - Motors, servos, basic actuators
- **UART (RX/TX)**:
  - Communication with your laptop
  - Prints, debugging, logging
- **I^{2}C (SDA/SCL)**:
  - Multiple smart sensors & devices on two wires
  - Perfect for rich data inputs for AI/ML

In this week’s lab:

- We mostly use **digital** and **PWM** pins
- You now know where **RX/TX** and **SDA/SCL** fit into the bigger picture

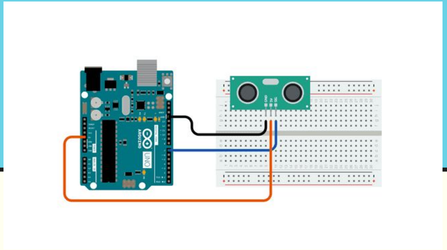
Ultrasonic Sensor: What It Does

Ultrasonic Sensor – Measuring Distance**

- Works like a mini sonar:
  - Sends an ultrasonic **ping**
  - Waits for the **echo**
  - Measures **time** → converts to **distance**
- We use it to detect:
  - Hand distance
  - Obstacles
  - Simple presence

Pins:

- VCC, GND, TRIG, ECHO



Connect:

- **VCC** → 5V
- **GND** → GND
- **TRIG** → digital pin **9**
- **ECHO** → digital pin **10**

Check:

- TRIG/ECHO pin numbers match code
- GND is common between Arduino and sensor

Servo Motor: What It Does

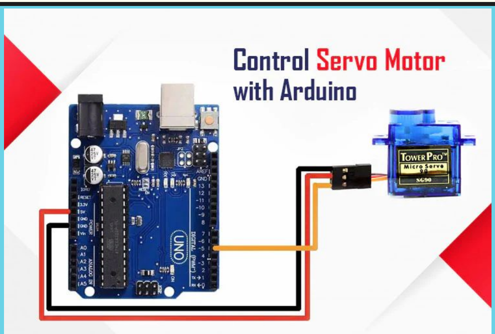
- A servo moves to a **specific angle** (0°–180°)
- Internally:
  - Motor + feedback + control electronics
- We command an **angle**, not just “spin”

Wires:

- Red → 5V
- Brown/Black → GND
- Orange/Yellow → signal from Arduino

This week:

- Servo angle is controlled by **distance**.



Connect:

- **Red** → 5V
- **Brown/Black** → GND
- **Orange/Yellow** → digital pin **3**

Important:

- All GNDs must be common:
  - Arduino GND
  - Sensor GND
  - Servo GND

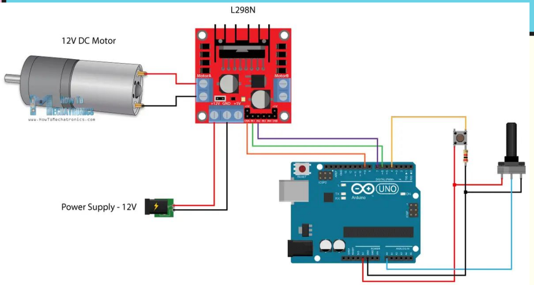
DC Motor: What It Is

DC Motor – Simple Spinning Power

- **DC motor** = basic motor that **spins continuously** when you apply voltage.
- Connect:
  - One side to **+5V / external supply**
  - Other side to **GND** (through a transistor in real circuits)
- Speed depends on:
  - **Voltage** (higher → faster)
  - **Load** (heavier → slower)

Use cases:

- Fans, wheels, pumps, conveyor belts



DC Motor:

- Continuous rotation (spins and spins)
- No built-in angle feedback
- Good for:
  - Wheels, fans, pumps, continuous motion

Servo Motor:

- Moves to and holds a **specific angle**
- Built-in feedback + control
- Good for:
  - Robot joints, camera tilt, pointing things

TODAYS TO DO

1. Connect Arduino & select **Board + Port**
2. Run the **PWM LED fade** (pin 9)
3. Wire & test **Ultrasonic sensor** (TRIG 9 / ECHO 10)
4. Wire & test **Servo motor** (signal pin 3)
5. Observe data on **Serial Monitor**
6. **In-Class Task 1:** Ultrasonic + Servo distance control (in groups)

Lab Requirements & Expectations

What You Need for Lab

- Bring:
  - Laptop (with permission to install software)
  - Arduino kit (board, USB cable, breadboard, basic components)
- In lab you will:
  - Follow **GitHub step-by-step guides**
  - Complete 2-**3 small tasks** per week
  - Ask lots of questions – confusion is normal

Safety & Lab Rules

Work only with **low-voltage** circuits in this course

Do **not** plug random things into mains power

Double-check wiring **before** powering

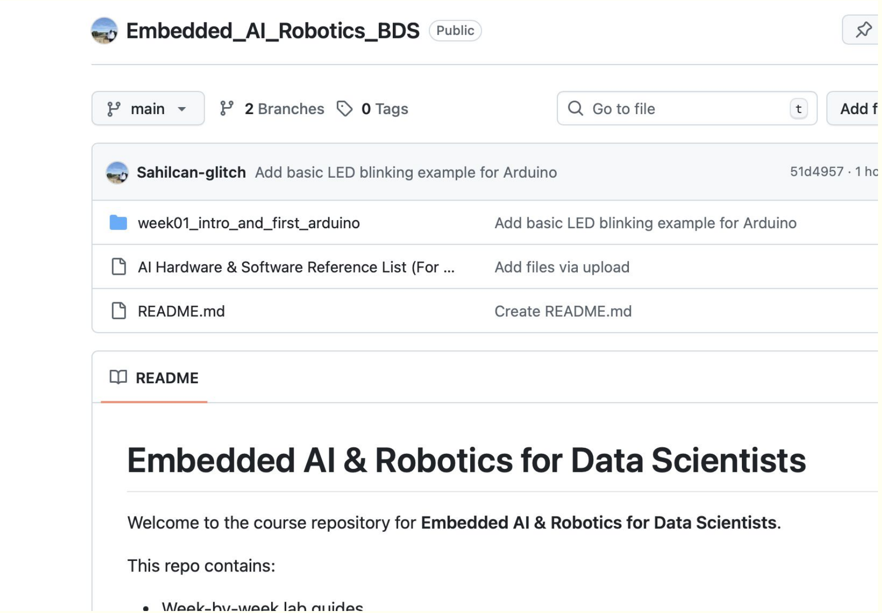
If something smells hot, smokes, or is weird:

Disconnect USB / power immediately

Call me or the instructor

Be kind to the hardware – it’s shared

GitHub Repo for This Course



Embedded_AI_Robotics_BDS

Sahilcan-glitch Add basic LED blinking example for Arduino 51d4957 · 1 hc

week01_intro_and_first_arduino	Add basic LED blinking example for Arduino
AI Hardware & Software Reference List (For ...	Add files via upload
README.md	Create README.md

README

Embedded AI & Robotics for Data Scientists

Welcome to the course repository for **Embedded AI & Robotics for Data Scientists**.

This repo contains:

- Week-by-week lab guides

All weekly content is here:

- `embedded_ai_robotics_bds` (https://github.com/Sahilcan-glitch/Embedded_AI_Robotics_BDS)

For Week 2:

- `week01_intro_and_first_arduino/`
- `README.md` =lab guide
- `Inclass_activity` = example `.ino` files

Q&A

?