

A blue Arduino Uno microcontroller board is shown from a top-down perspective, angled slightly. A black USB cable is connected to the board's USB port. The board features a central ATmega328P microcontroller chip, various surface-mount components like resistors and capacitors, and a breadboard-style pin header. A dark rectangular overlay with rounded corners is centered on the board. Inside this overlay, the text "ARDUINO CRASHCOURSE" is written in large, white, sans-serif capital letters, followed by a smaller line of text "- ZERO TO HERO!" in the same style.

ARDUINO CRASHCOURSE

- ZERO TO HERO!

WHO AM I?



- **Jacob Bechmann Pedersen**

- Electronics Engineer (AU, 2019)
- Started using Arduino in 2014
- Volunteer in Coding Pirates 2016-2018
- Teaching at MakerCamp since 2018
 - 12-16 y/o "Inventors"
- Taught Arduino for IDA, StudyNow, Herning Municipality, etc.
- Full-time embedded electronics engineer at DTU
 - Robots and autonomous systems

PURPOSE OF THIS WORKSHOP

- Give you a **QUICK** understanding of the basics and essentials of Arduino
 - Hands-on successes
 - Experience some difficulties
- Inspiration on how to use Arduino for yourself
- Try some IoT capabilities of ESP32 on the Arduino platform

BEFORE WE GET STARTED

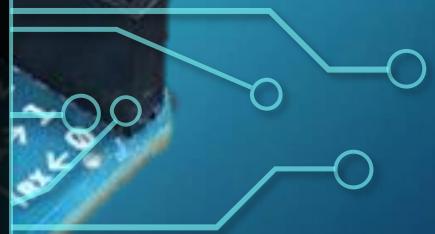
- The code and examples throughout the workshop are available on:
 - <https://github.com/iakop/ArduinoCrashcourseloT>
 - It's a good idea to keep this site open throughout

AGENDA

- 16:00-17:30 : Arduino Basics
- 17:30-18:00 : Setup of ESP32 on Arduino
- 18:00-18:30 : Dinner - Pizza and softdrinks
- 18:30-19:45 : IoT on ESP32
- 19:45-20:00 : Summary, evaluation, and thanks for now!



ARDUINO BASICS



WHAT IS ARDUINO?

- Most used platform for MCU prototyping
 - MCU is short for Micro Controller Unit
 - A tiny computersystem, controlling electronics
- Simple, intuitive programming
- Many code examples available
 - Many software libraries too!

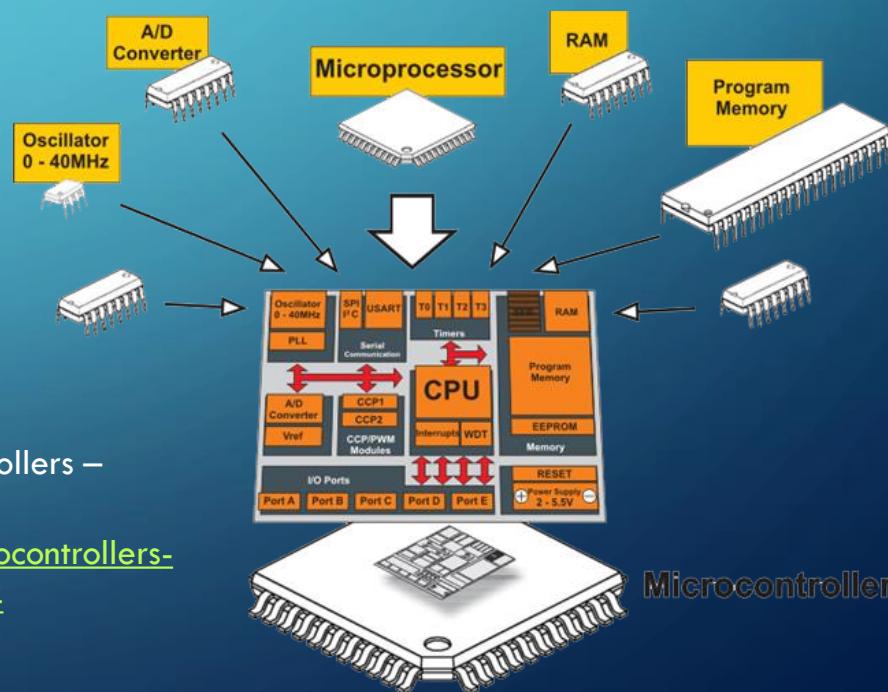
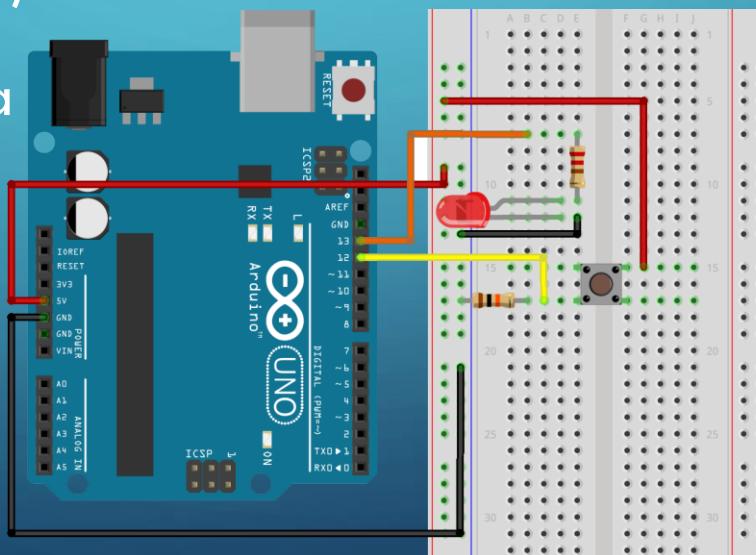


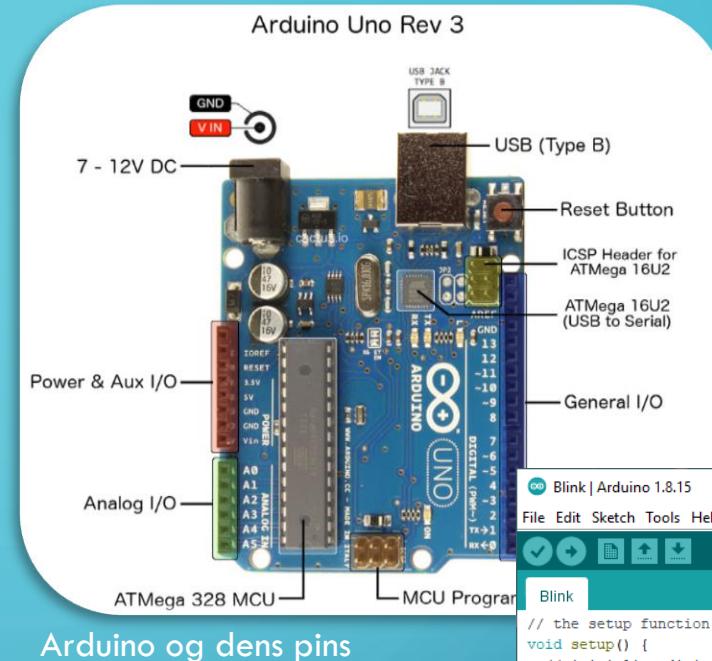
Figure: Introduction to the World of MicroControllers –
Mikroelektronika
URL: <https://www.mikroe.com/ebooks/pic-microcontrollers-programming-in-c/introduction-to-the-world-of-microcontrollers>

THE ARDUINO PLATFORM

- Arduino connects to electronics through its "pins"
- Pin signals are controlled through programs (turn on/off, etc.)
- Often connected through a "breadboard"



Arduino forbundet til elektronik gennem breadboard



Arduino og dens pins

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);    // turn the LED on (HIGH is the voltage level)
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);     // turn the LED off by making the voltage LOW
  delay(1000);                      // wait for a second
}
```

Arduino Uno on COM4

Det officielle Blink program i Arduino IDE

HVORFOR ARDUINO?

- Compare the following two programs
 - Which makes the most sense upon reading?

```
1 //Physical LED pin
2 #define LEDWRITEPORT PORTB
3 #define LEDPIN 5
4 //Physical button pin
5 #define BTNREADPORT PIND
6 #define BTNPIN 3
7
8 int main() {
9
10    //Initialize pins
11    DDRB |= (1 << LEDPIN); //ledPin set to output
12    DDRD &= ~(1 << BTNPIN); //btnPin set to input
13
14    while(1){
15        //If button is pressed LED on
16        if((PIND & (1 << BTNPIN)) >> BTNPIN == 1){
17            PORTB |= (1 << LEDPIN);
18        }
19        //Else, LED off
20        else{
21            PORTB &= ~(1 << LEDPIN);
22        }
23    }
24 }
```

```
1 const int ledPin = 13; //Physical LED pin
2 const int btnPin = 3; //Physical button pin
3
4 void setup() {
5     //Initialize pins
6     pinMode(ledPin, OUTPUT); //ledPin set to output
7     pinMode(btnPin, INPUT); //btnPin set to input
8 }
9
10 void loop() {
11     //If button is pressed LED on
12     if(digitalRead(btnPin) == HIGH){
13         digitalWrite(ledPin, HIGH);
14     }
15     //Else, LED off
16     else{
17         digitalWrite(ledPin, LOW);
18     }
19 }
```

GETTING STARTED:

- Download and install Arduino IDE
- Get the version for your platform
 - For your own sake, do NOT get the Microsoft Store version!!!
- Examples from this workshop are on Github:
<https://github.com/iakop/ArduinoCrashcourseIoT>



Arduino IDE 1.8.16

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the [Getting Started](#) page for Installation instructions.

SOURCE CODE

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

DOWNLOAD OPTIONS

Windows Win 7 and newer
Windows ZIP file

ALDRIG HENT DENNE HER

Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits

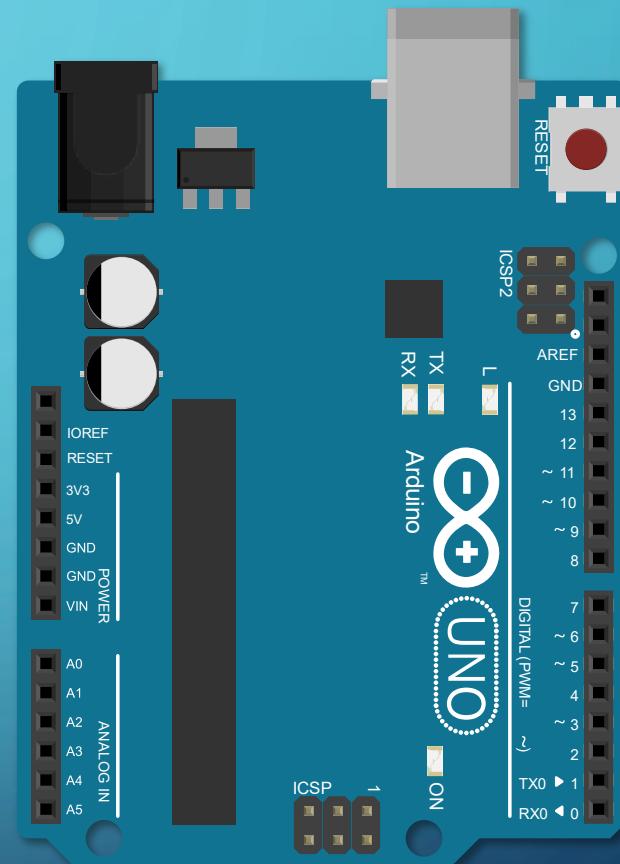
Mac OS X 10.10 or newer

[Release Notes](#) [Checksums \(sha512\)](#)

<https://www.arduino.cc/en/software>

EXAMPLE 0: BLINK

- To test if the Arduino works
- We'll compile and upload the program located in the IDE:
"File > Examples > 01.Basics > Blink"



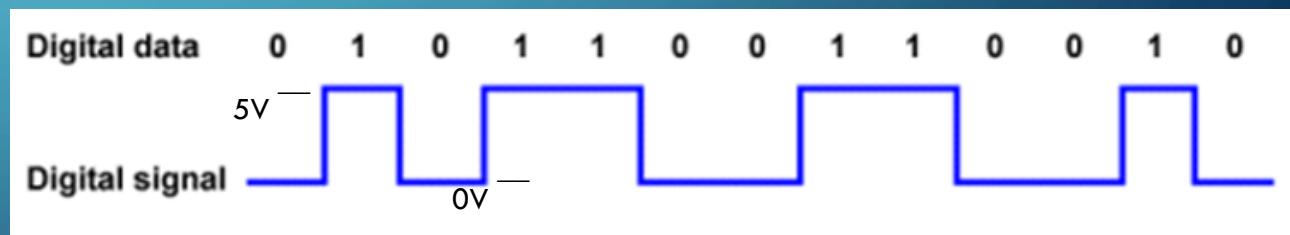
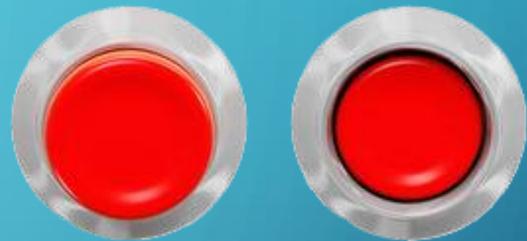
ABOUT DIGITAL COMPONENTS

- Blink example used a built-in LED
 - As a digital output
- Digitale outputs are for example:
 - LED's
 - Motors
 - Relays
- Digital when they are solely operating as on/off
- Digitale inputs are f.eks.
 - Buttons
 - PIR sensors
 - Tilt sensors



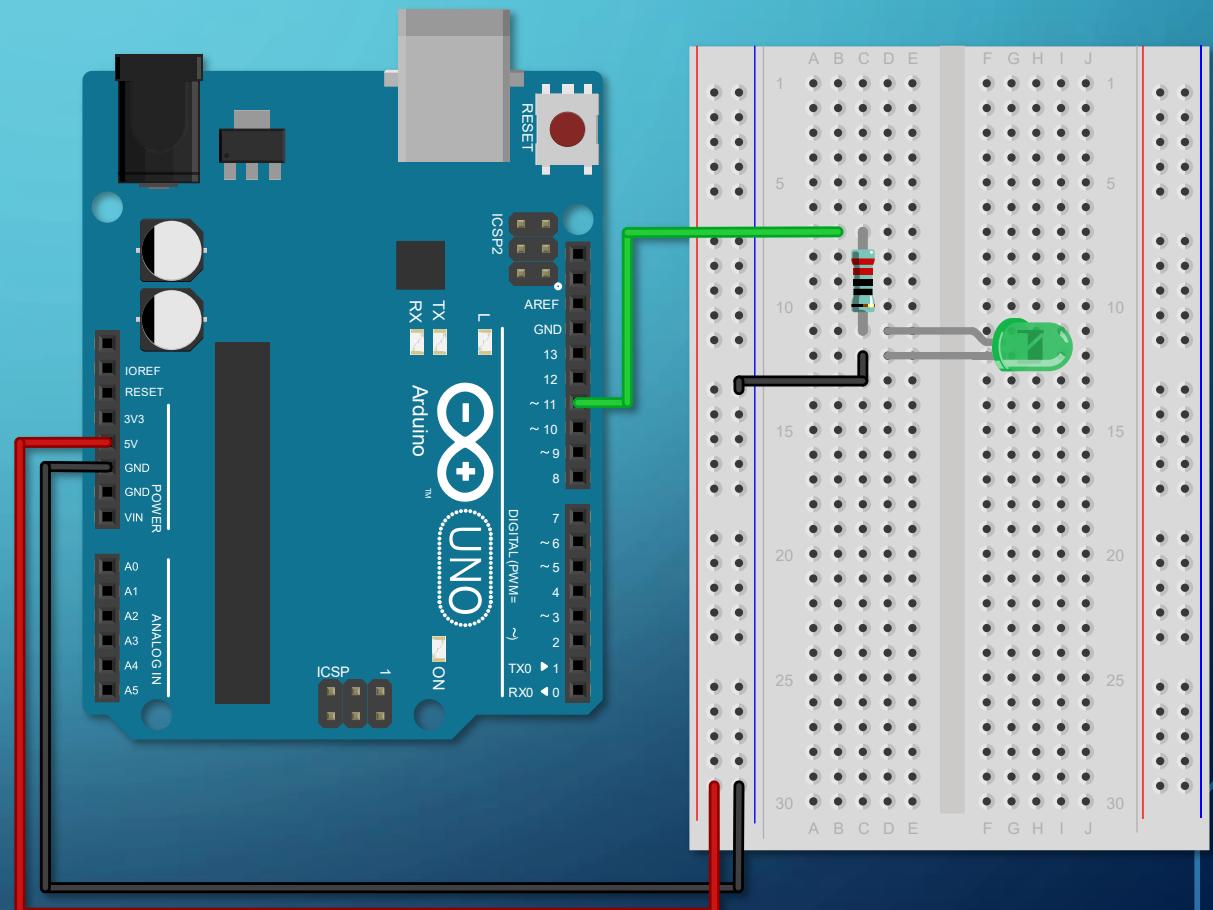
DIGITAL ON ARDUINO

- Digital data are "boolean" values;
 - Either "True" or "False"
 - Examples:
 - "Is the door open"
 - "Is the button pushed"
- Arduino gets the data from real voltages:
 - 5V for True
 - 0V for False
- Represented in code as HIGH or LOW



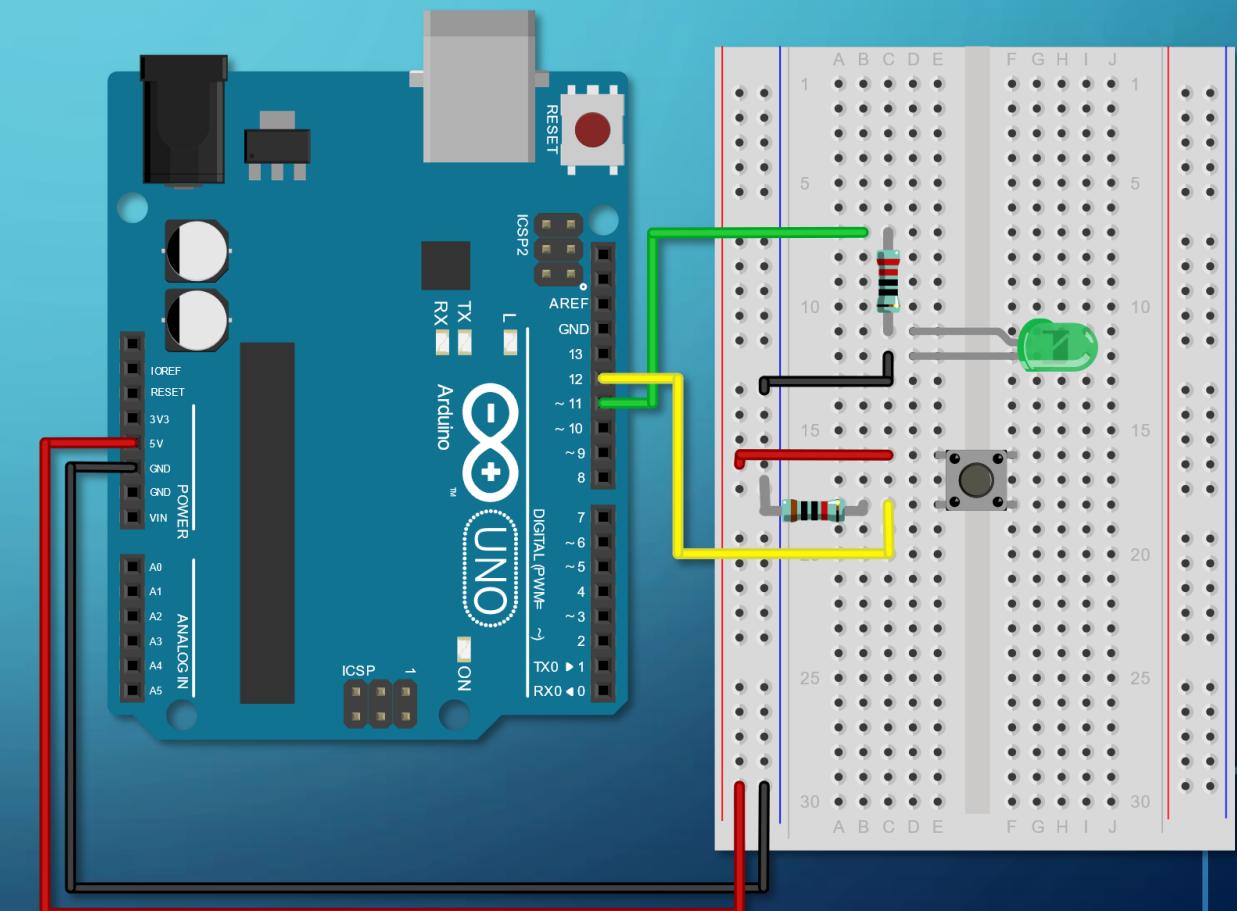
EXAMPLE 1: BLINKY

- We'll make our own Blink, "Blinky"
- Done with external circuit
- And writing the code ourselves



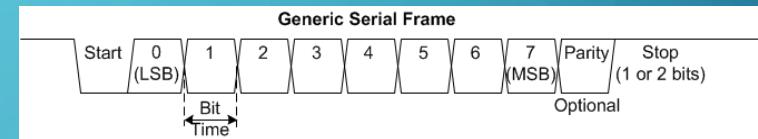
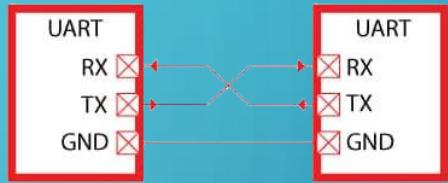
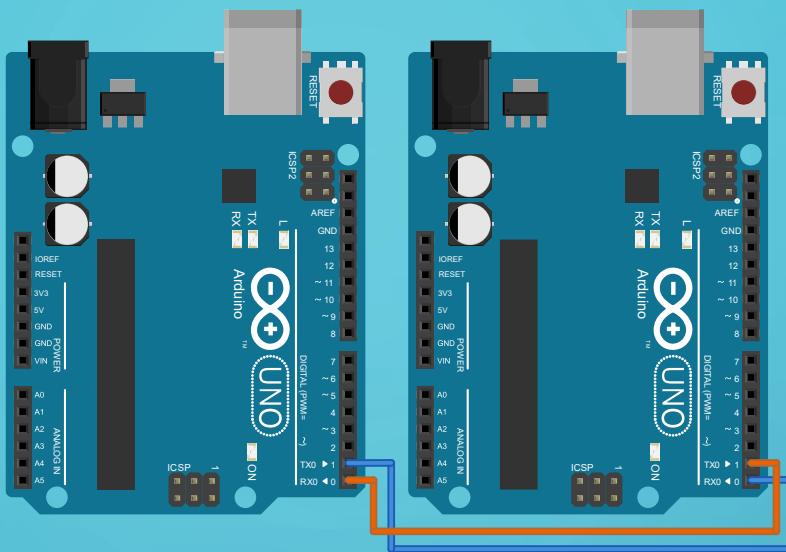
EXAMPLE 2: BUTTON DEBOUNCE

- We attach a button to the Blinky example
- Will make the LED toggle on or off
- Needs some mechanical phenomena to be handled in code

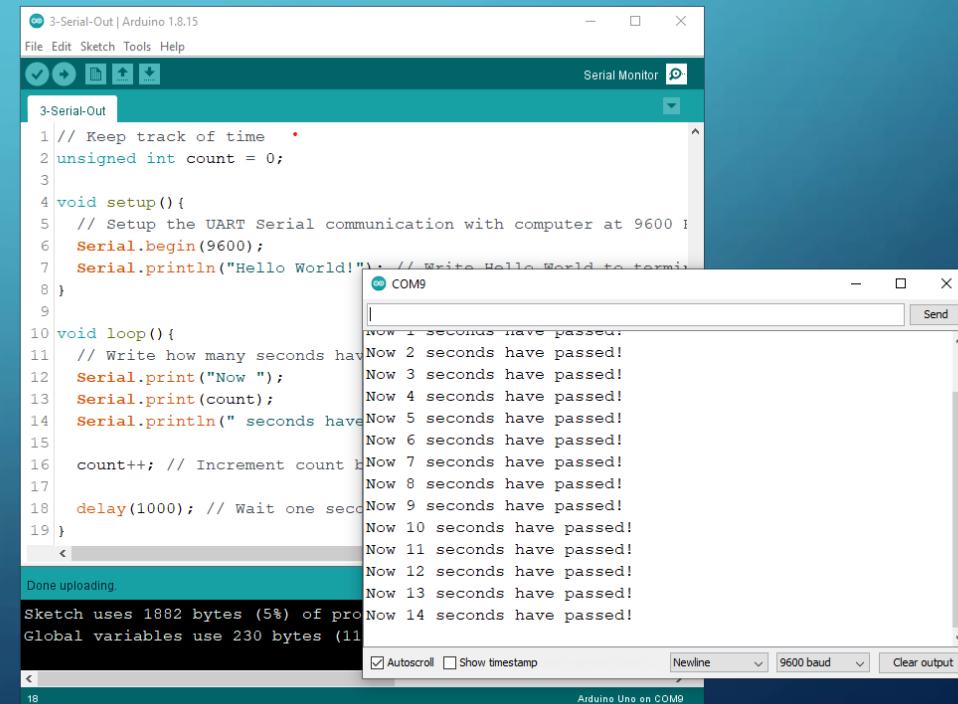


SERIAL - UART

- Serial is communication through a "series" of characters
 - Opposite to parallel
- Arduino can talk to the computer through Serial
 - Specifically UART
 - Universal Asynchronous Receiver/Transmitter
 - Data sent through TX/RX pins through converter to USB

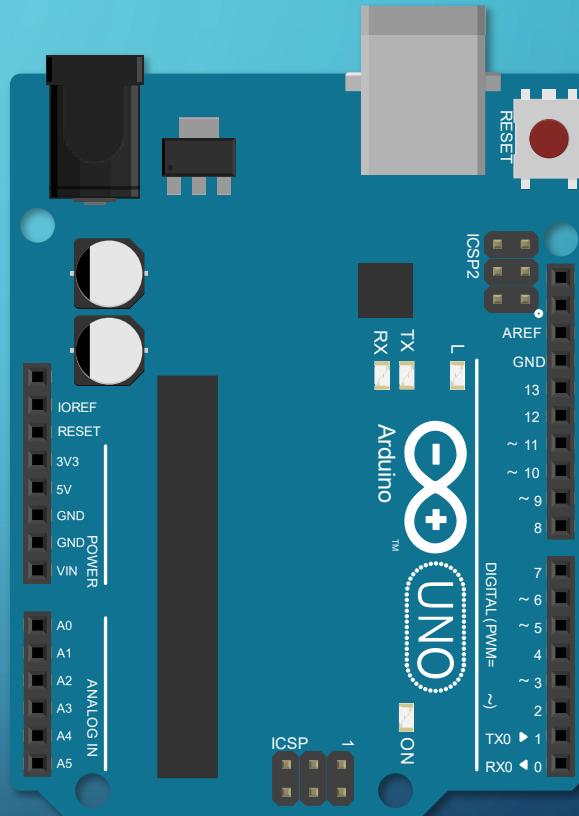


Billede: UART forbindelse og dataframe
URL: <https://microcontrollerslab.com/uart-communication-working-applications/>



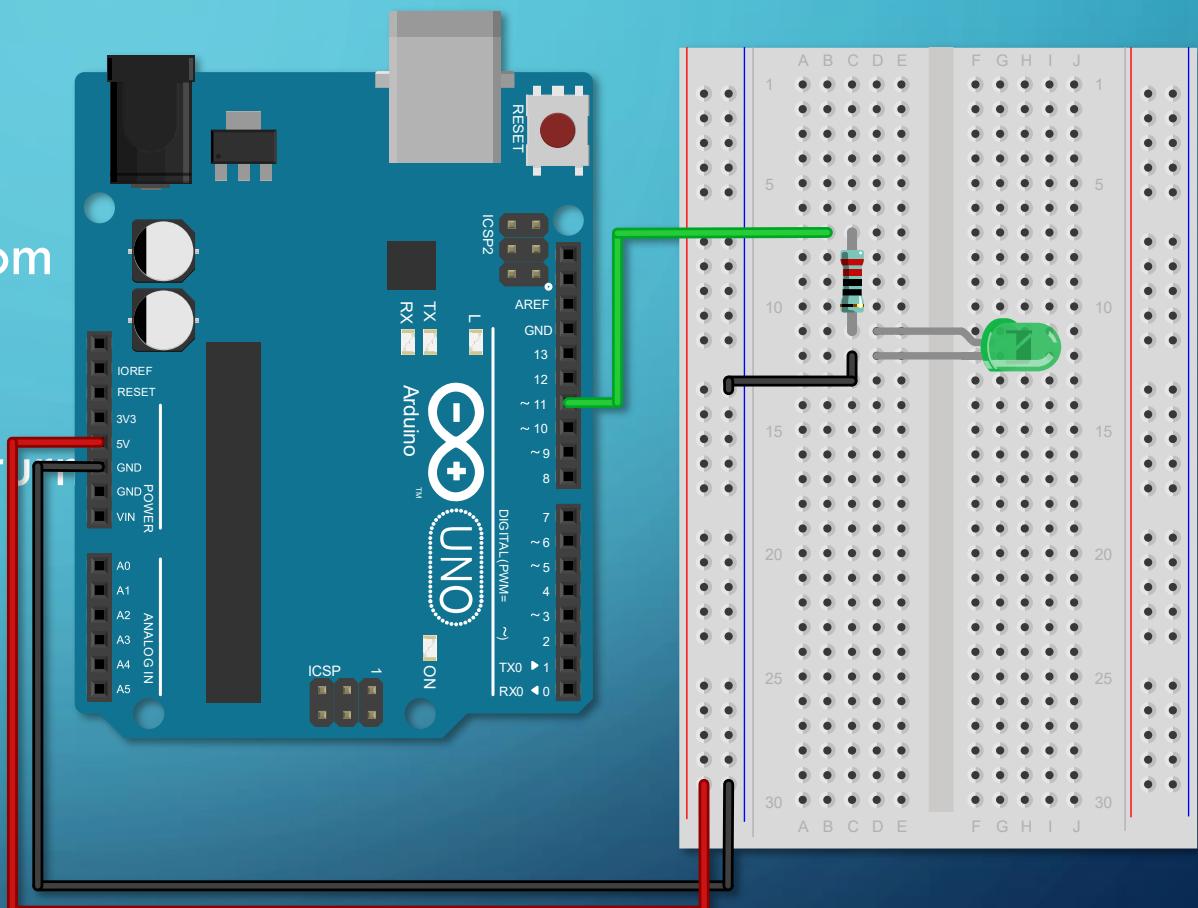
EXAMPLE 3: SERIAL OUT

- Arduino will write a message to the computer
- Serial Monitor can be found in "Tools>Serial Monitor"



EXAMPLE 4: SERIAL IN

- Arduino will receive a message from the computer
- In Serial Monitor we'll be able to turn on/off an LED



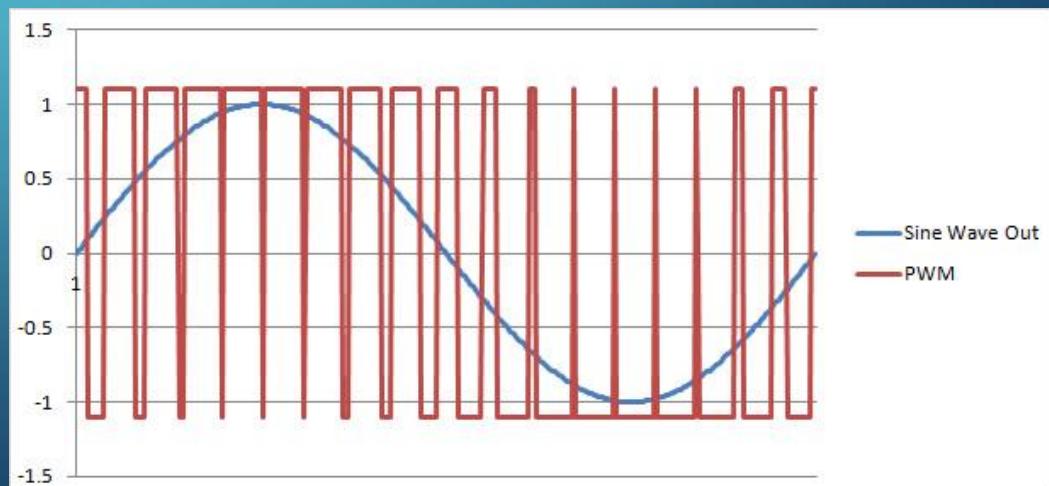
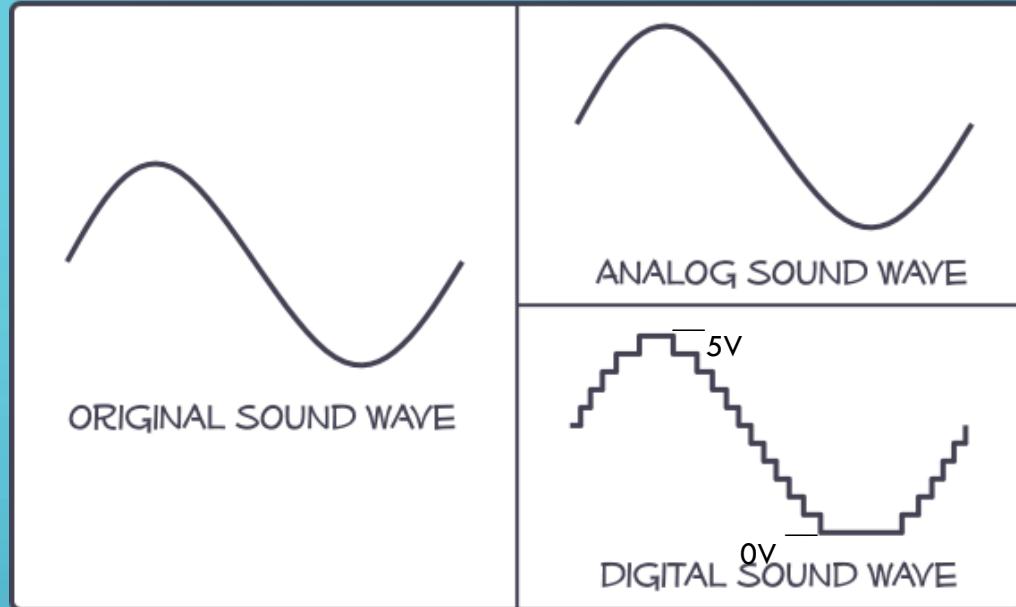
ABOUT ANALOG COMPONENTS

- Analog outputs and inputs are not just True or False
 - Values "in-between"
- Analogue outputs are for example:
 - LED (Variable brightness)
 - Motor (Variable speed)
 - Servomotor
- Analogue inputs er f.eks.:
 - Potentiometer
 - Joystick
 - Light Dependent Resistor



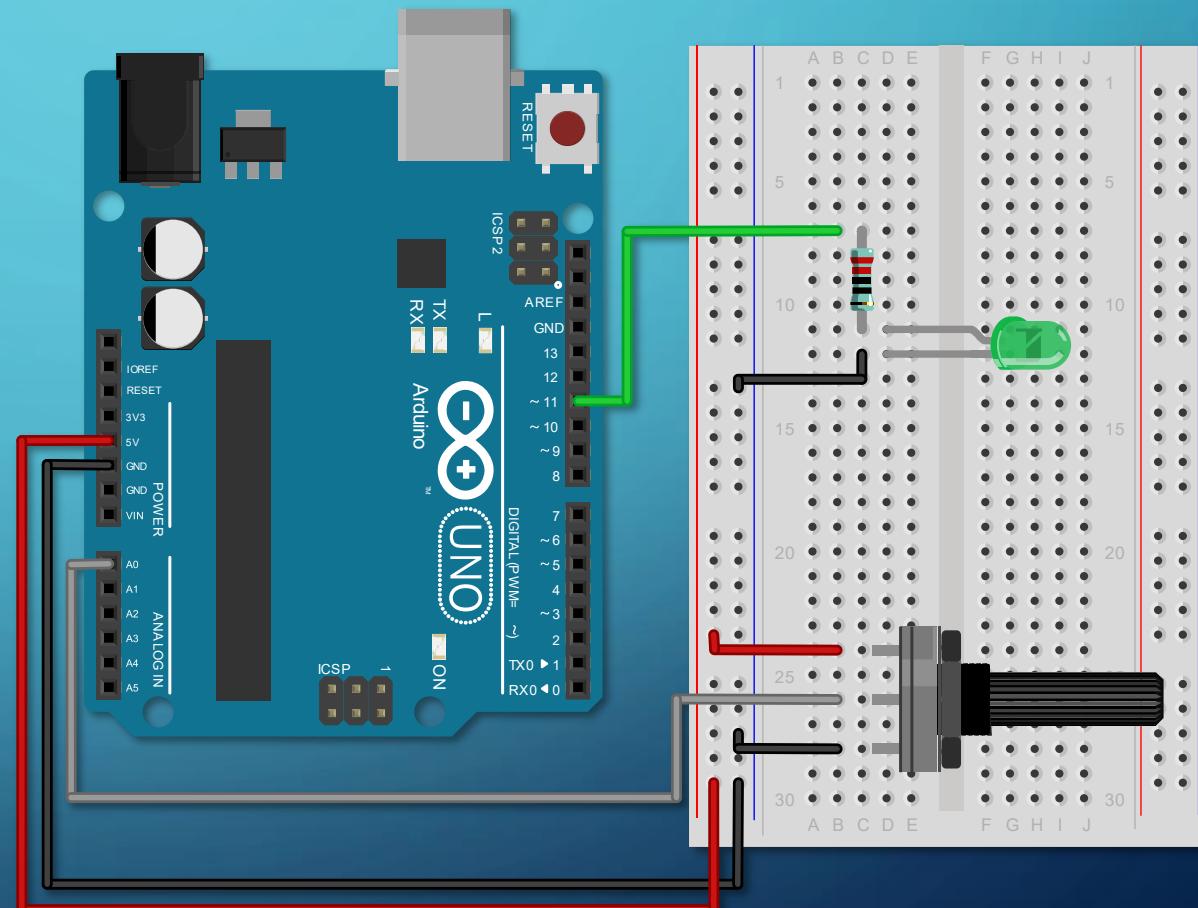
ANALOG ON ARDUINO

- Arduino can also measure analog data
 - Done on the analog input pins
- It can produce a "semi" analog signal
 - Done via PWM (Pulse Width Modulation)



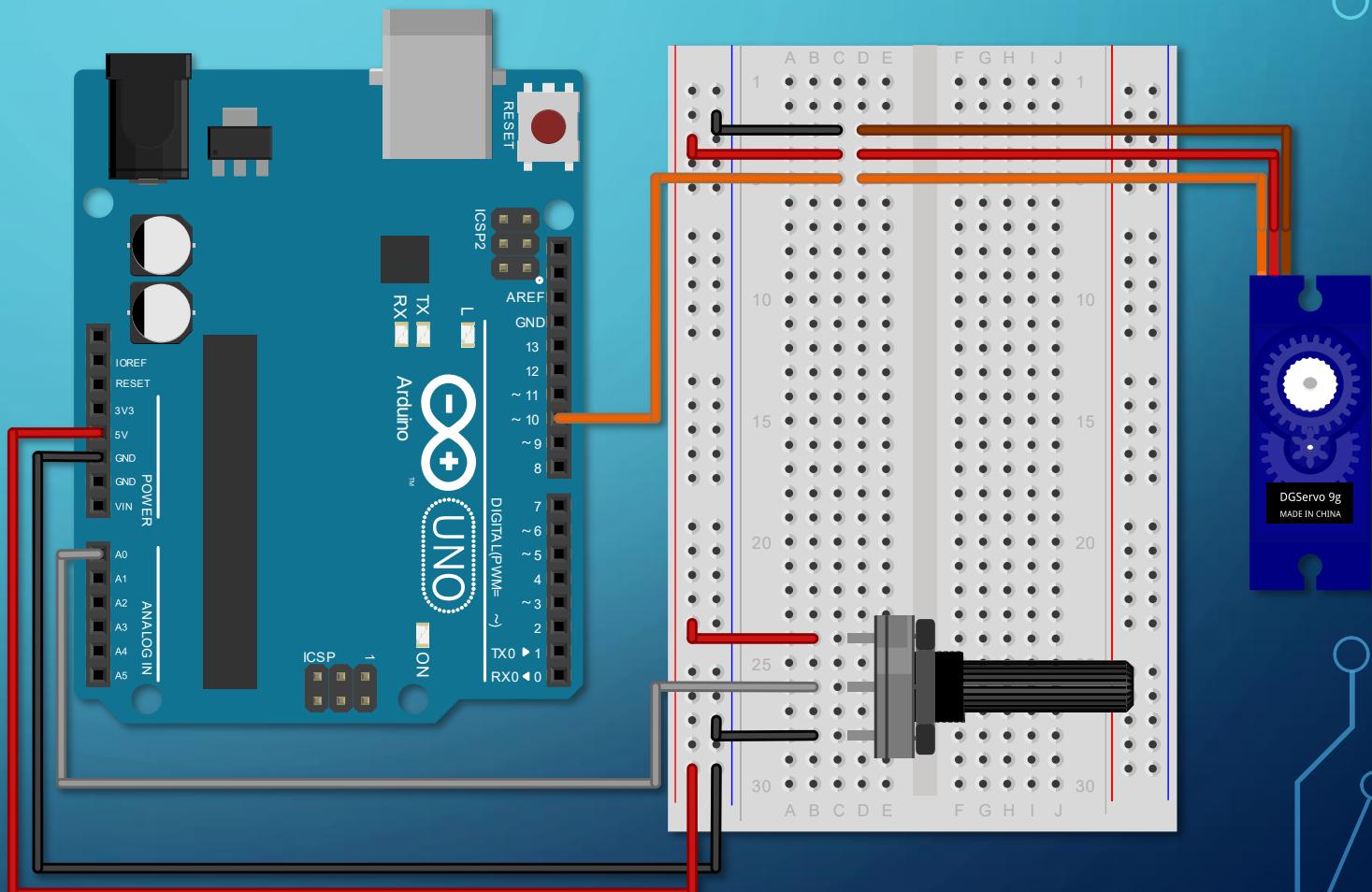
EXAMPLE 5: ANALOG LED

- We'll read the value of a potentiometer
- Use this data to dim an LED
- Status is written over Serial



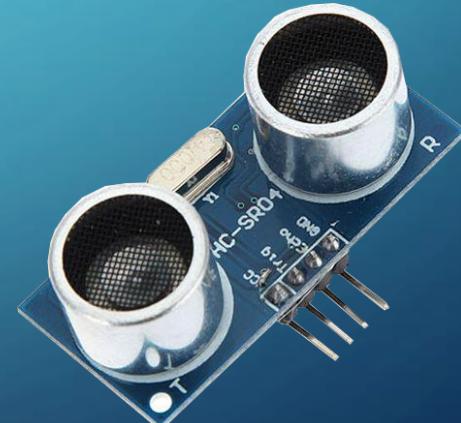
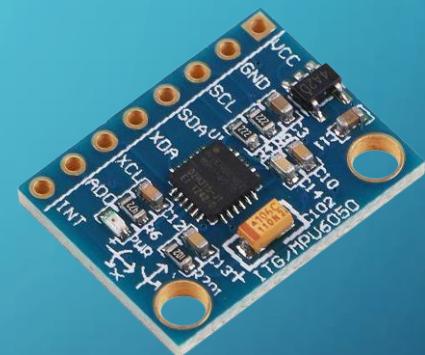
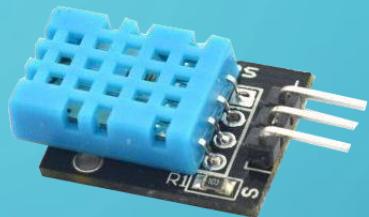
EXAMPLE 6: SERVO

- We'll include a "library" to control the Servo motor
- Controlled from potentiometer
- Status written over Serial



OTHER COMPONENTS - SENSORS

- Temperature-/humidity sensor
- Accelerometer/gyroscope
- Ultrasound distance sensor
- Often communicate through protocols protokoller
 - Handled easily through libraries



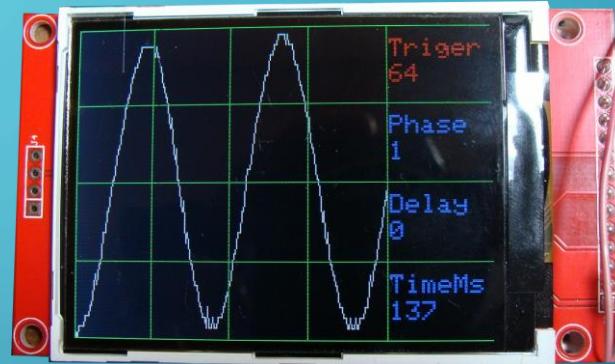
OTHER COMPONENTS - ACTUATORS

- Stepper Motors
- Speakers
- Linear actuators
- Pistons
- Most use no, or simple, protocols
 - Steppers and speakers have libraries



OTHER COMPONENTS - DISPLAYS

- Character displays (LCD)
- OLED
- TFT LCD
- Many different libraries each
 - Depends on use-case





An Arduino Uno microcontroller board is shown from a top-down perspective, tilted slightly. A small ESP32 module is mounted on top of the Arduino. The board features a blue PCB with various electronic components, including a central microcontroller chip, capacitors, and resistors. The Arduino logo and part numbers like 'ATMEGA328P' and '16MHz' are visible. A large, semi-transparent black rectangular box is centered over the ESP32 module. Inside this box, the text 'SETUP OF ESP32' is displayed in a bold, white, sans-serif font. Two thin, light-blue lines extend from the bottom-left and bottom-right corners of the text box, pointing towards the physical pins of the ESP32 module on the Arduino board.

SETUP OF ESP32

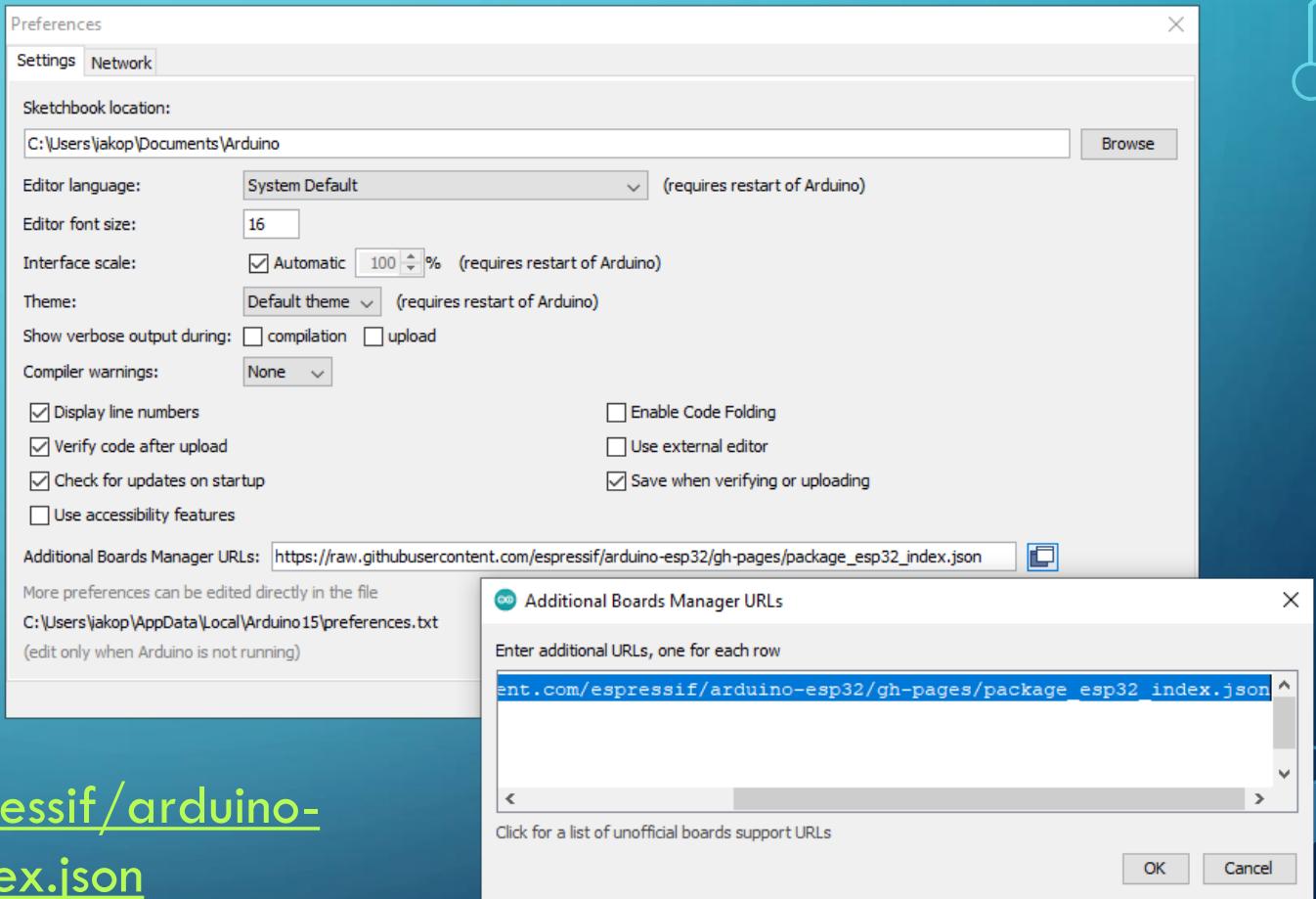
ESP32

- WiFi SoC
 - "System on Chip"
 - Read: MCU with built-in WiFi
- Heir to ESP8266
- Can be programmed through Arduino IDE
- Requires a little setup



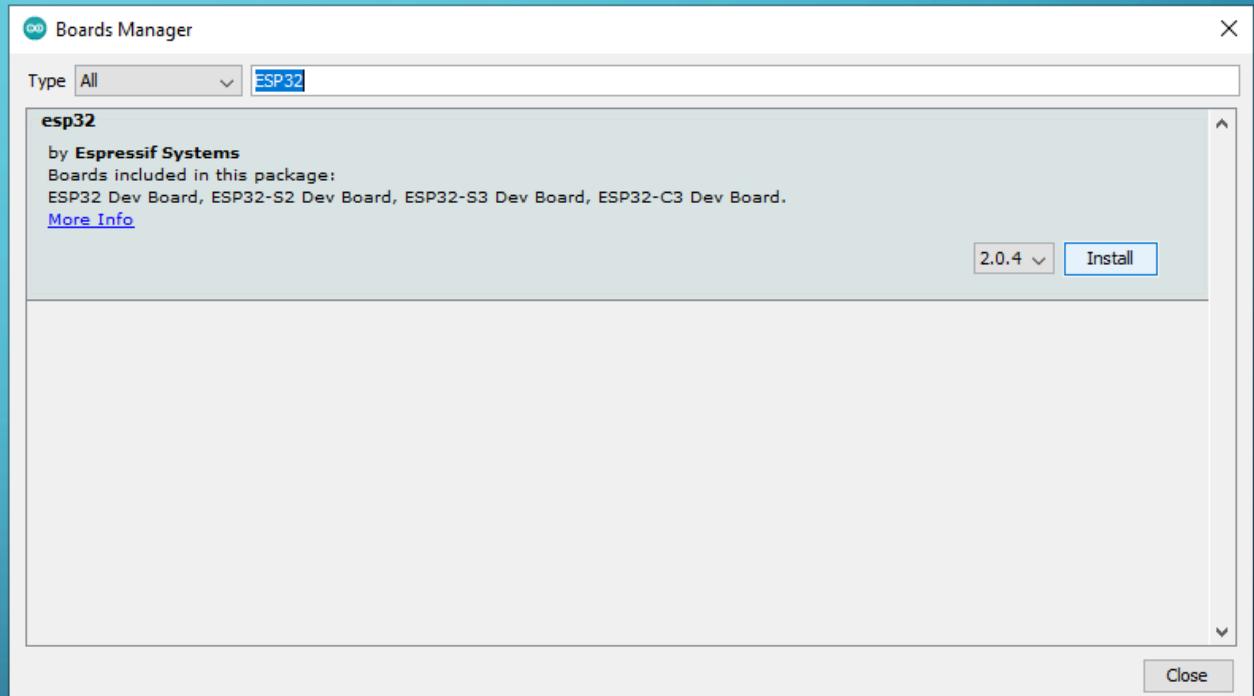
BOARD DEFINITION

- In Arduino IDE:
 - "File > Preferences"
- Additional board manager URLs:
 - Insert:
https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json



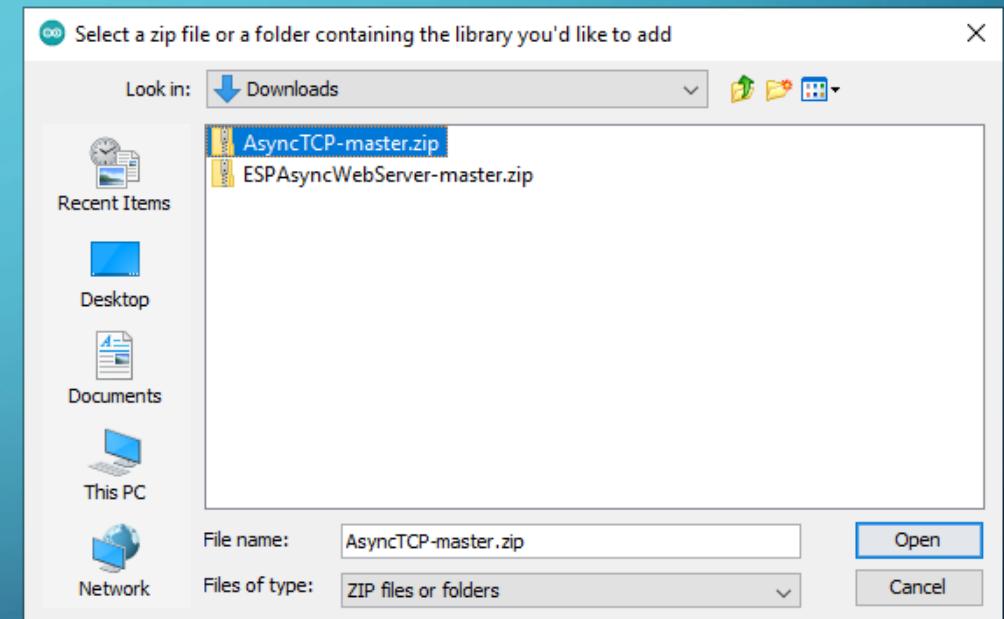
BOARD DEFINITION

- In "Tools > Board > Board Manager"
- Search for ESP32 and press Install
 - Takes a while



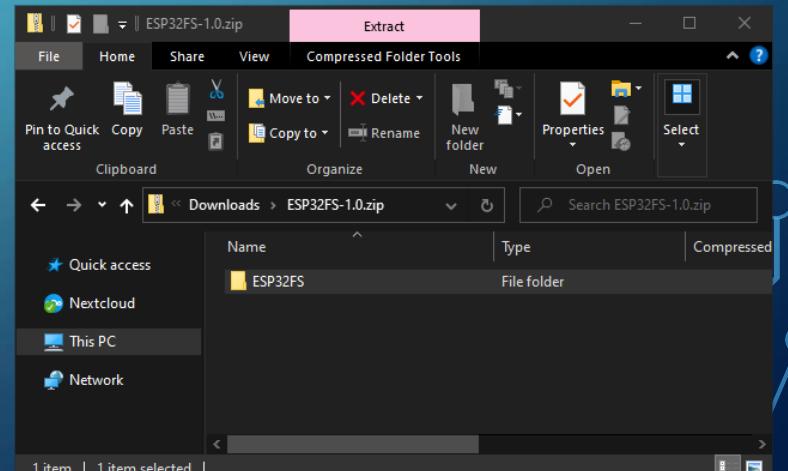
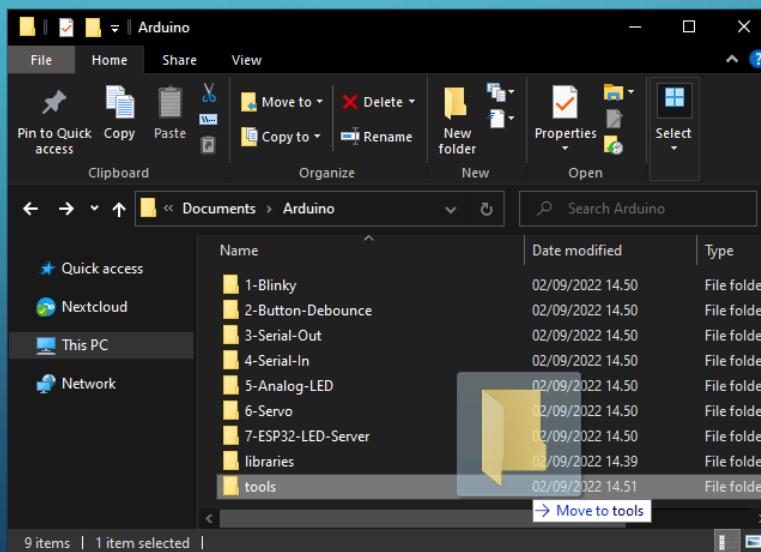
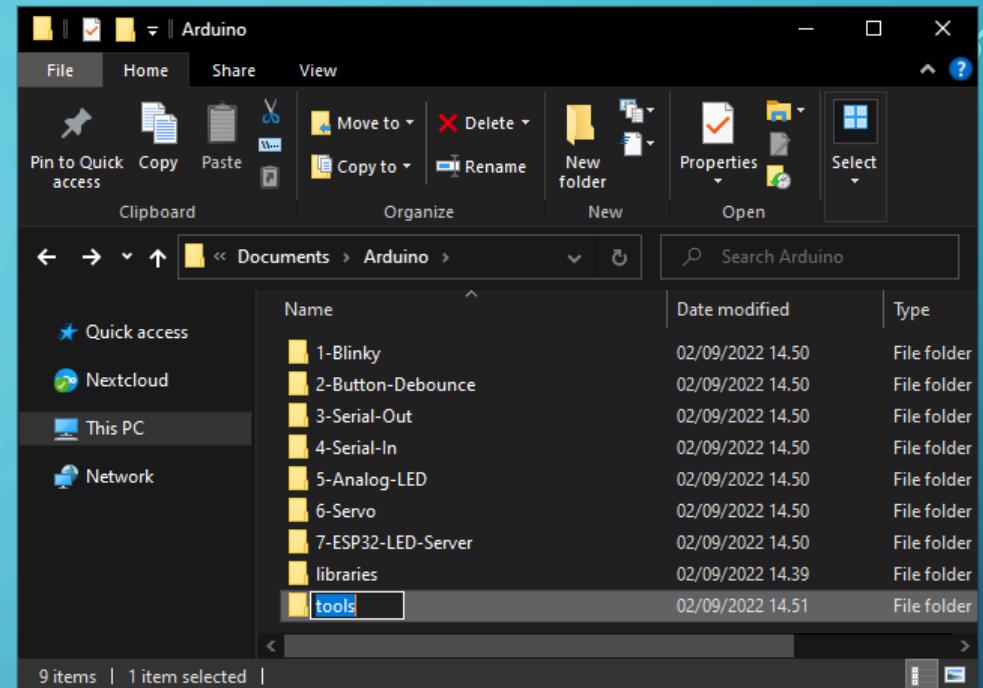
EXTERNAL LIBRARIES

- Download the following two links:
- <https://github.com/me-no-dev/ESPAsyncWebServer/archive/refs/heads/master.zip>
- <https://github.com/me-no-dev/AsyncTCP/archive/refs/heads/master.zip>
- In Arduino IDE "Sketch > Include Library > Add .ZIP Library..."
 - For every library, select and install it



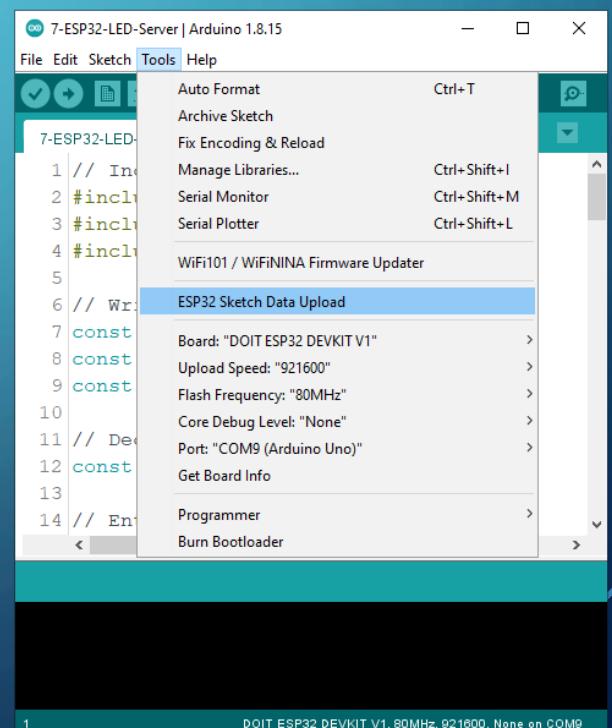
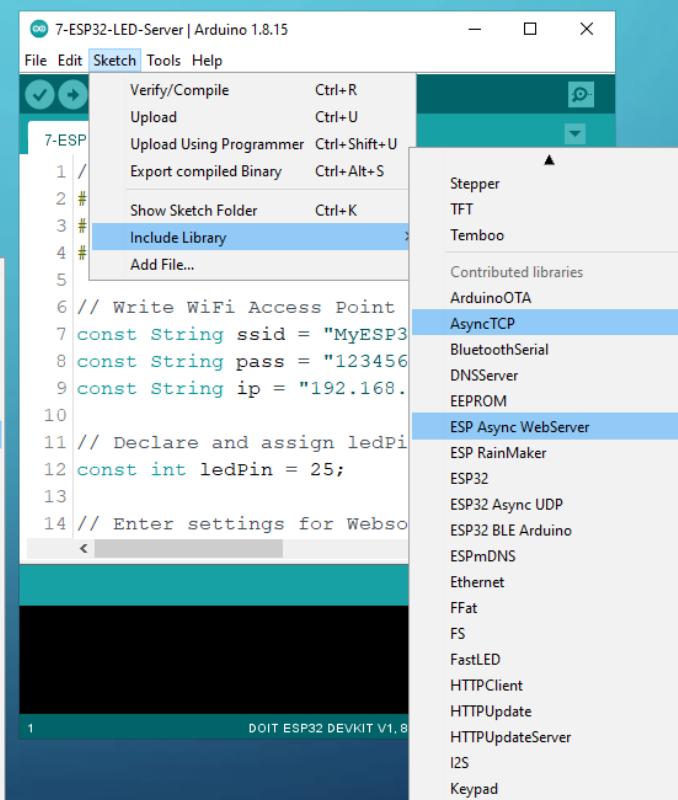
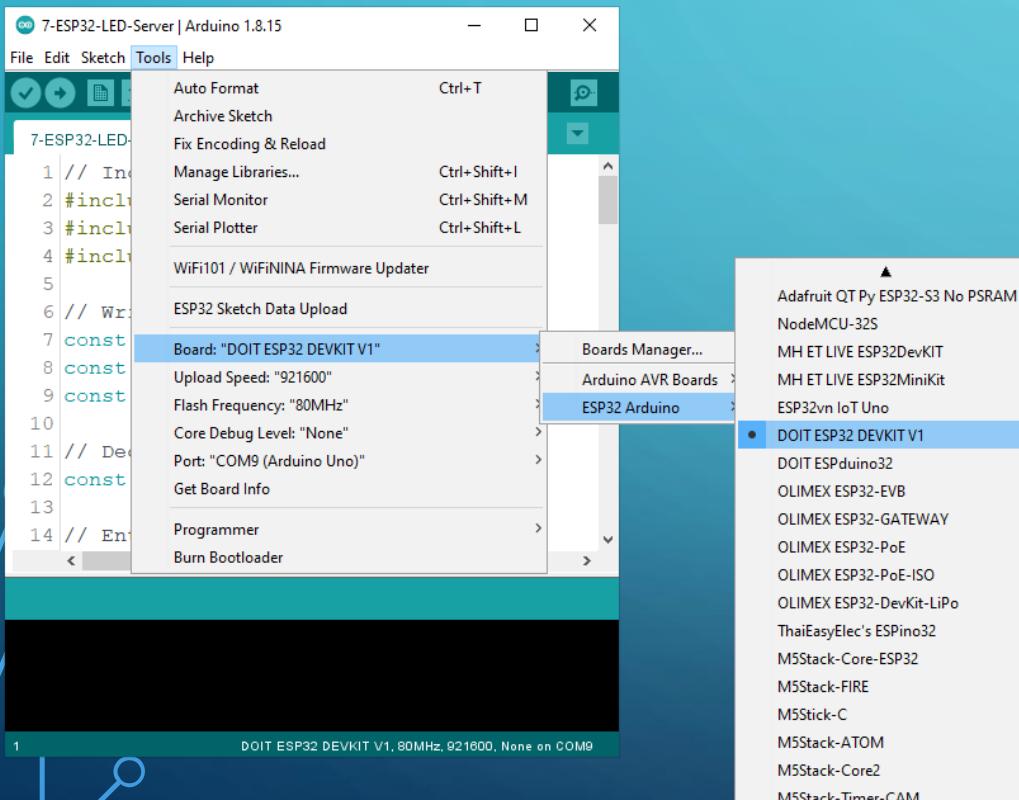
EXTERNAL TOOLS

- Download this tool plugin:
<https://github.com/me-no-dev/arduino-esp32fs-plugin/releases/download/1.0/ESP32FS-1.0.zip>
- In your PC's local "Documents/Arduino" folder create a "tools" folder
- Unpack ESP32FS into the tools folder
- Restart Arduino IDE



EXTERNAL TOOLS AND LIBRARIES

- We should now be able to pick the following in Arduino IDE





An Arduino Uno R3 microcontroller board is shown from a top-down perspective, tilted slightly. A silver ESP32 module is mounted on top of the board. The board features a blue PCB with various components like resistors, capacitors, and integrated circuits. The word "ARDUINO" is printed on the board. A central black rectangle contains the text "IOT ON ESP32".

IOT ON ESP32

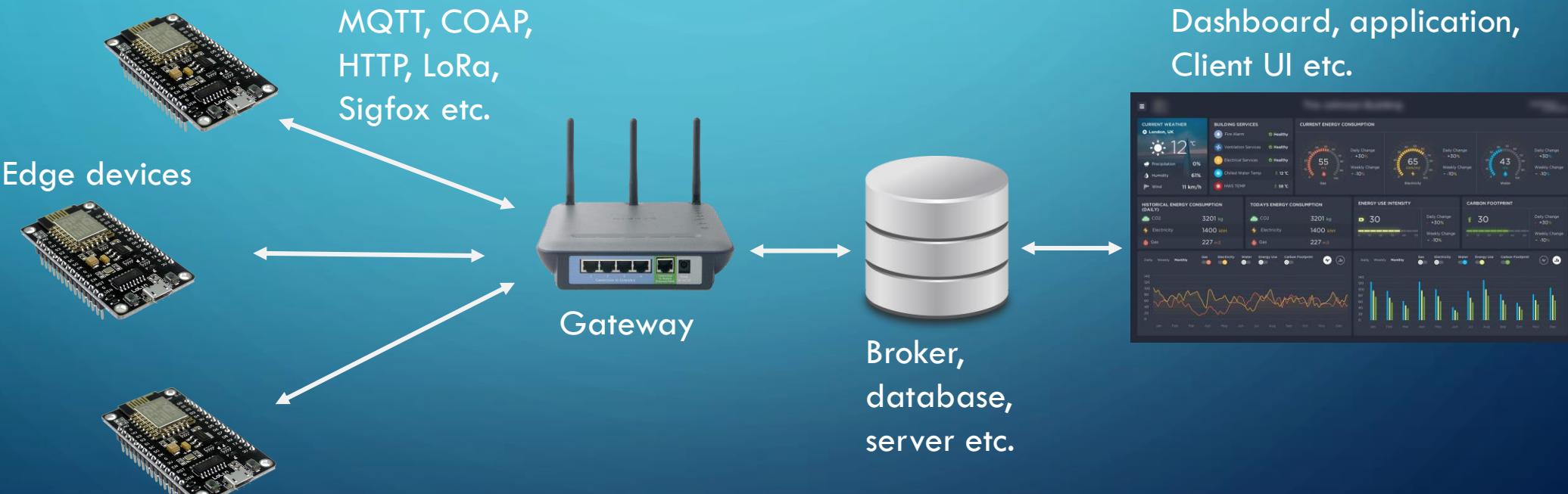
IOT IN GENERAL

- IoT (Internet of Things covers devices "available" over a network)
- Home network:
 - Home assistants
 - Philips Hue
- In the field:
 - WasteHero
 - TrapMe
 - MinkPolice



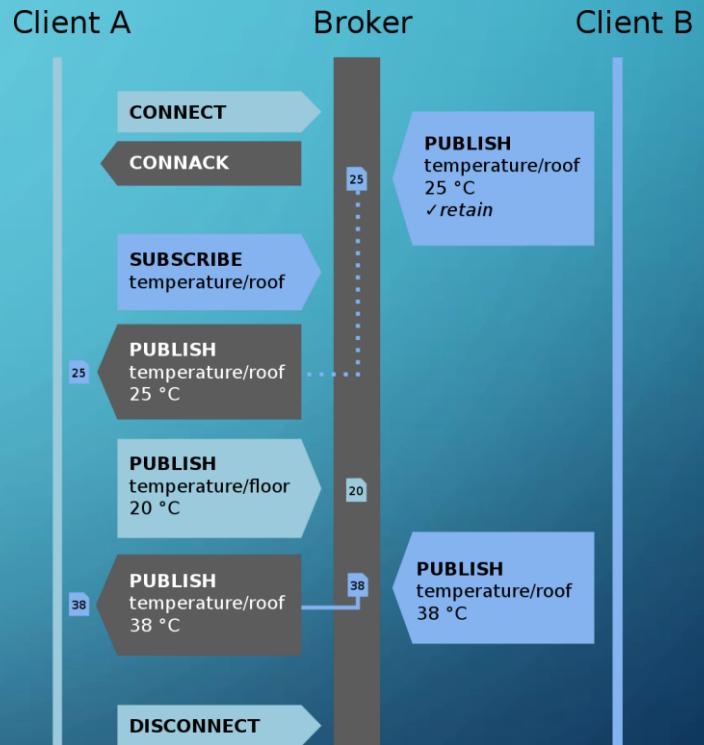
IOT IN GENERAL

- IoT deployment architectures usually look like:



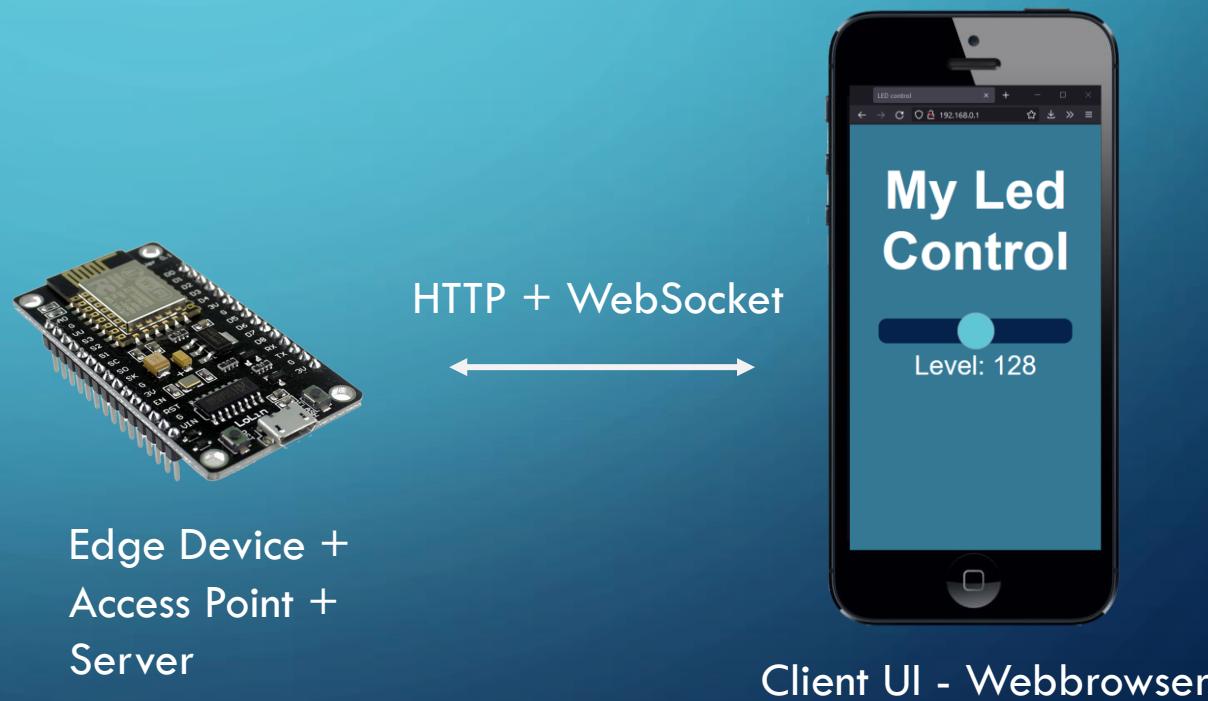
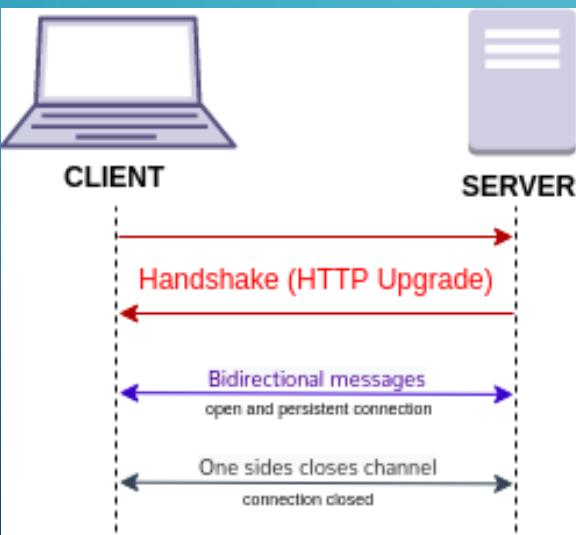
IOT IN GENERAL

- One of the most popular protocols for IoT is MQTT
 - Simple and performant
- Requires clients and broker
 - Example from Wikipedia
 - Two thermostats on a network talk to the same broker
 - Exchange data asynchronously, the broker passes relevant data on



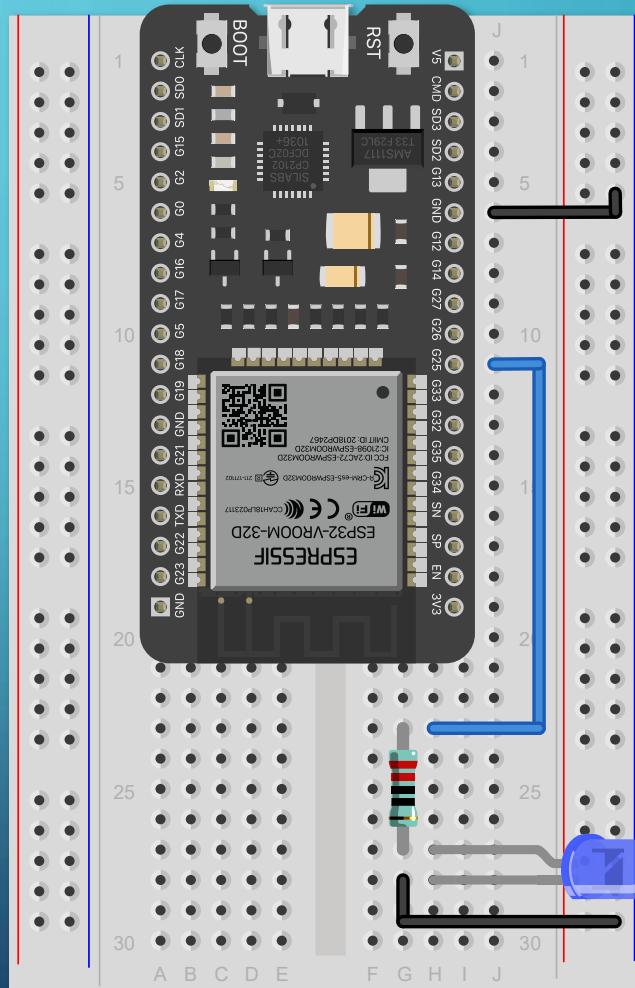
ESP32 WEB SERVER

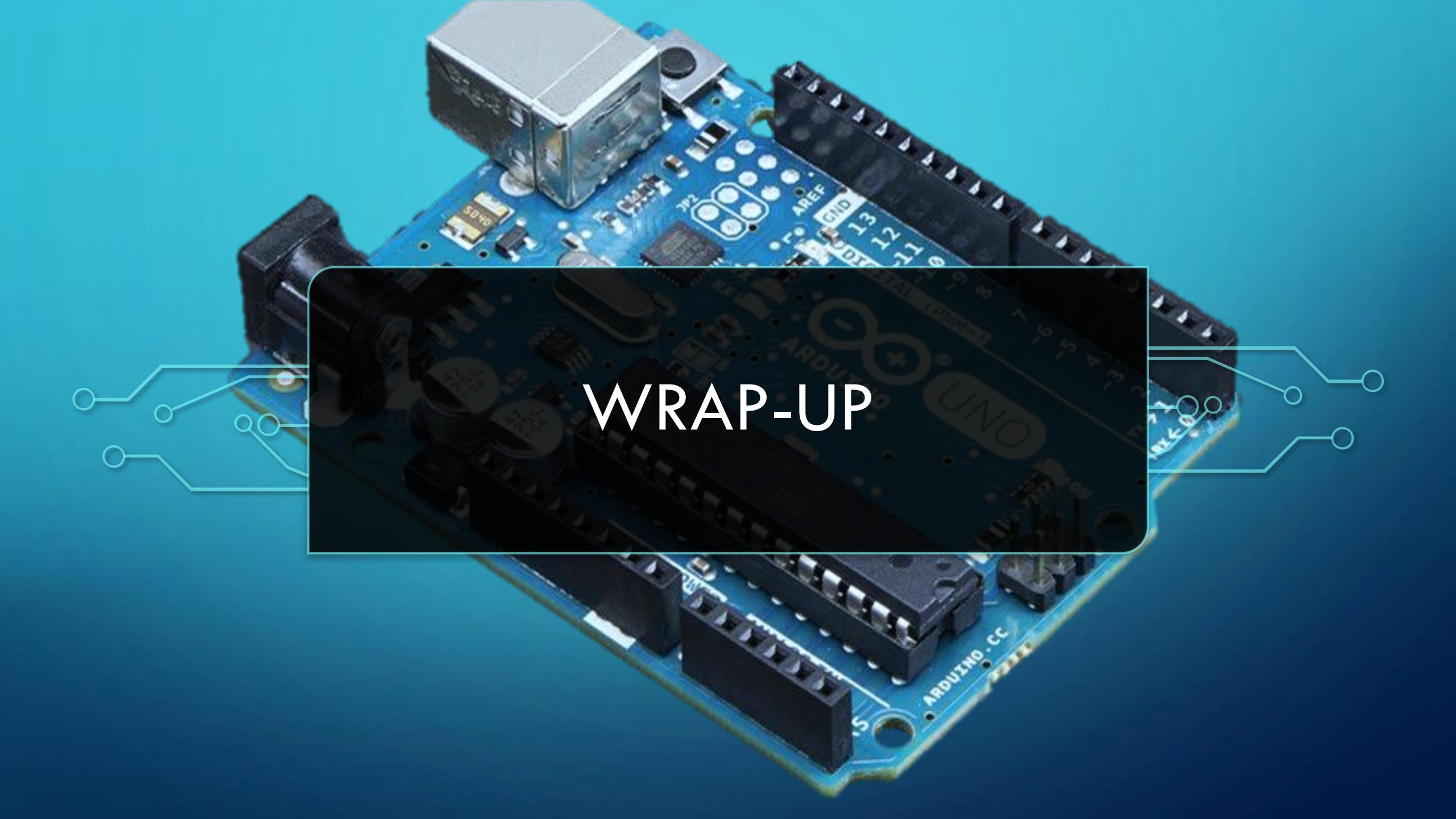
- To avoid setting up an entire network, we'll connect directly to the ESP32
- Websocket is simply direct messages "a-la Serial"



EXAMPLE 7: ESP32 LED SERVER

- We simply attach an LED to ESP32
- Most of the time will be used to code, decorate, and setup the webpage





WRAP-UP

OPSAMPLING

- Now you know the basics of:
 - Digital I/O
 - Serial UART
 - Analog I/O
 - Introduced to miscellaneous components
- Have set up ESP32 for Arduino development
- Can build a simple web service
 - ESP32 as server and access point
 - HTML/Javascript for communication
- Introduced to general IoT
 - Architecture
 - MQTT

EVALUATION

- Will happily accept direct feedback
 - What did you like, what not?
 - Could something be formulated better?
 - How did the material fit the time available?
- Most of all, THANK YOU for participating!

RESOURCES

- Some online resources you can use:
 - Slides, examples, etc.
 - <https://github.com/iakop/ArduinoCrashcourseloT>
 - Arduino IDE Download
 - <https://www.arduino.cc/en/software>
 - Arduino Language Reference
 - <https://www.arduino.cc/reference/en/>