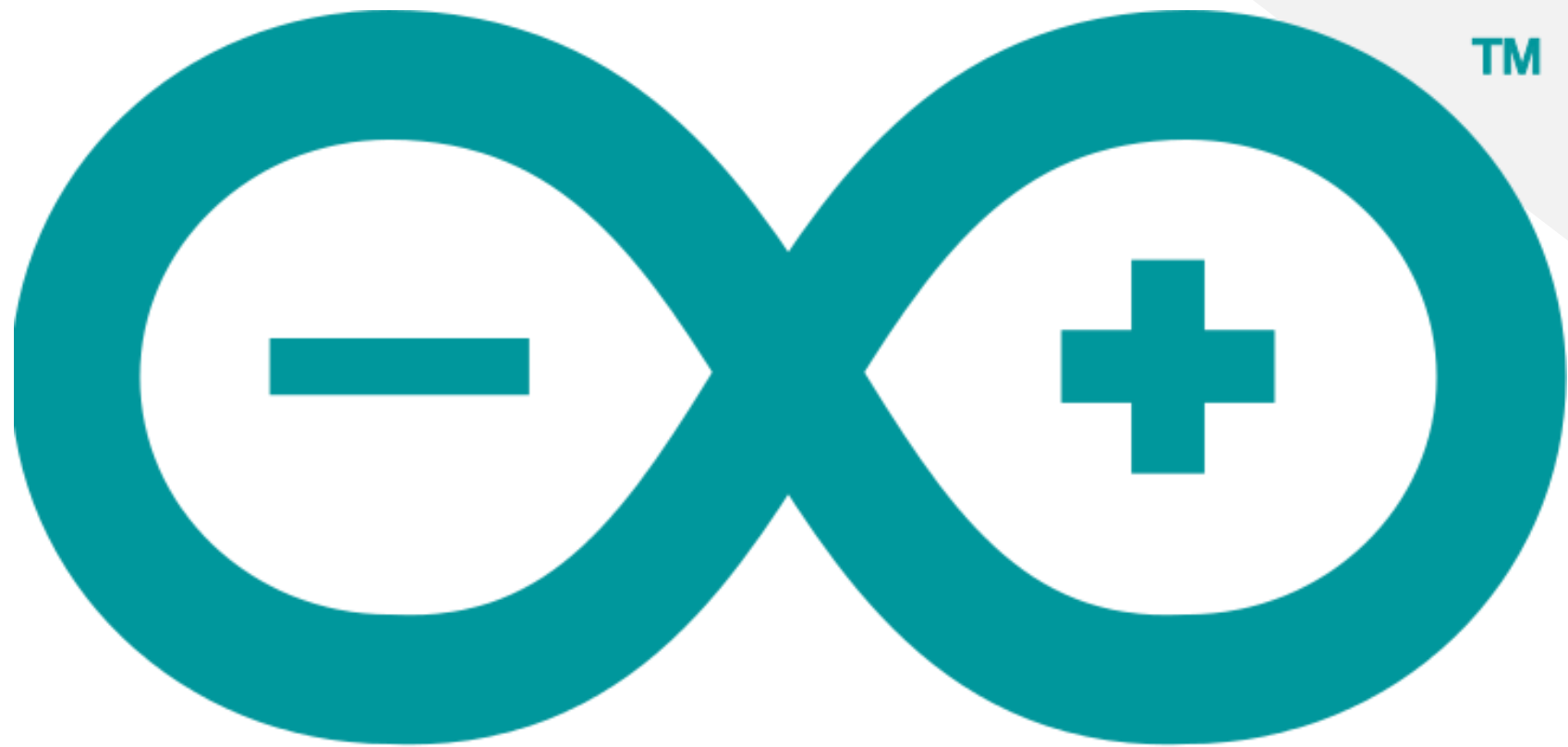


IoT

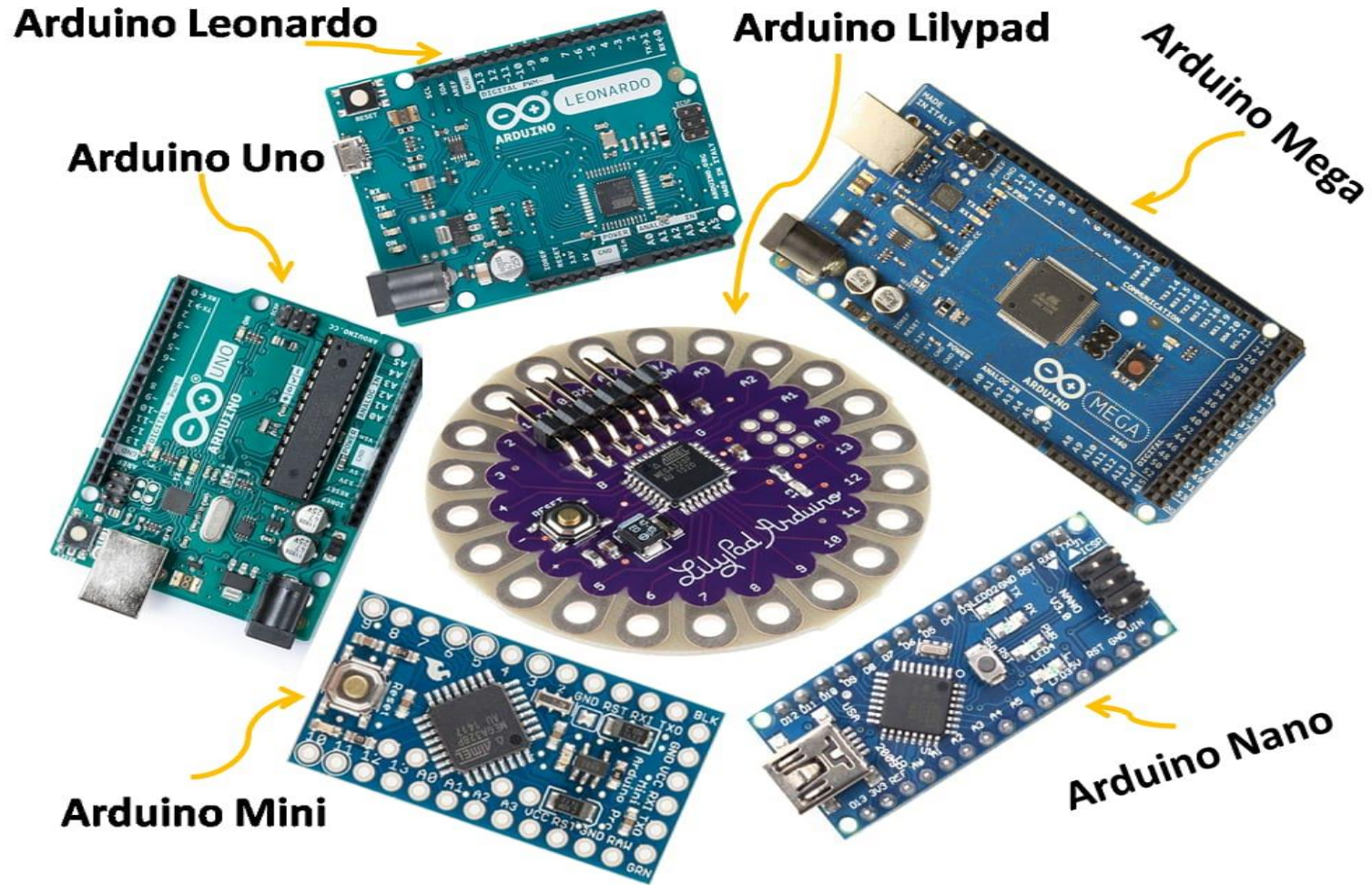
Practice 0

Ramyar A. Teimoor
University of Sulaimani
College of Science
Department of Computer
2025-2026

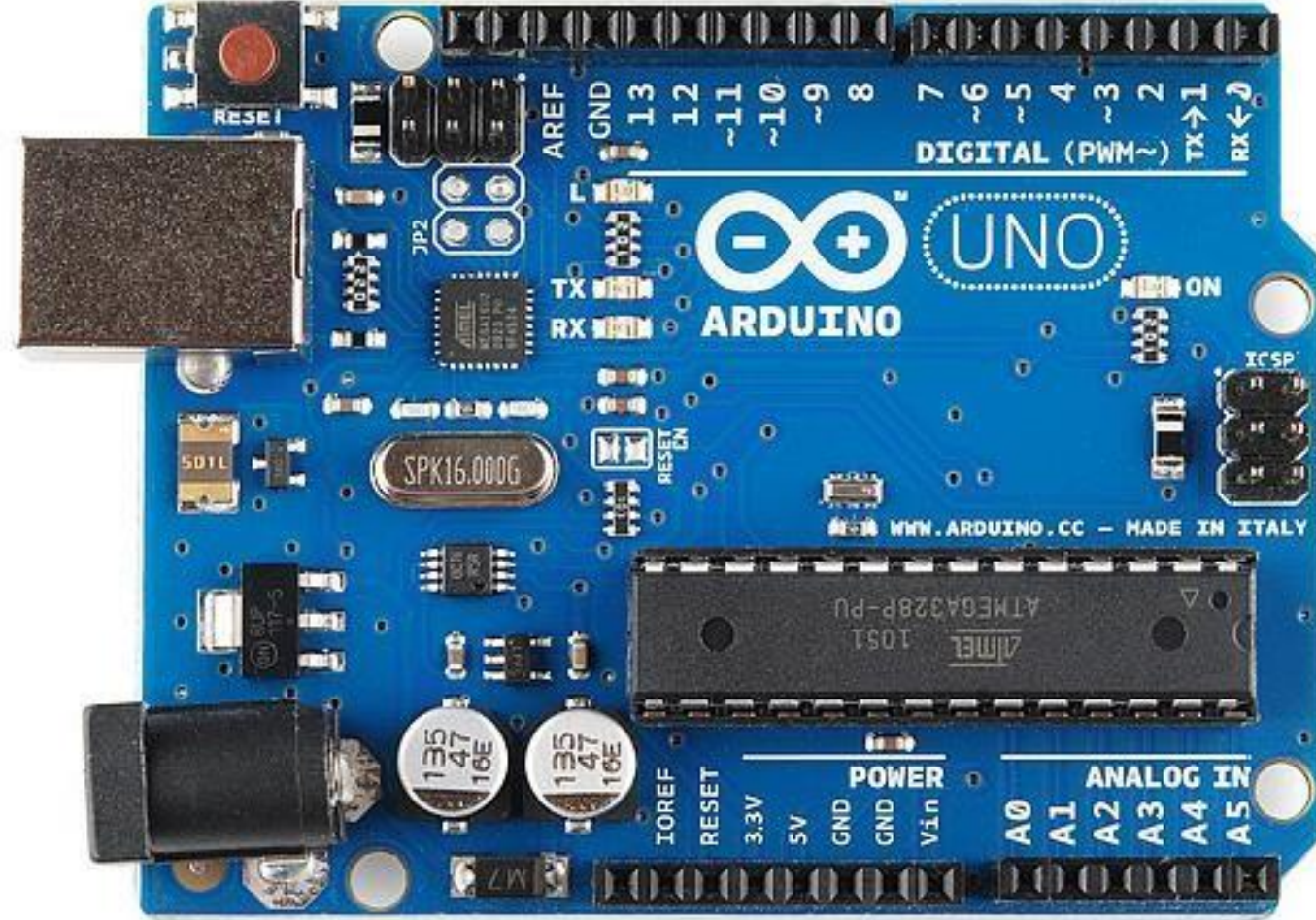


ARDUINO

Arduino Family



Arduino UNO



POWER
5V/3.3V/GND

**Analog
INPUTS**

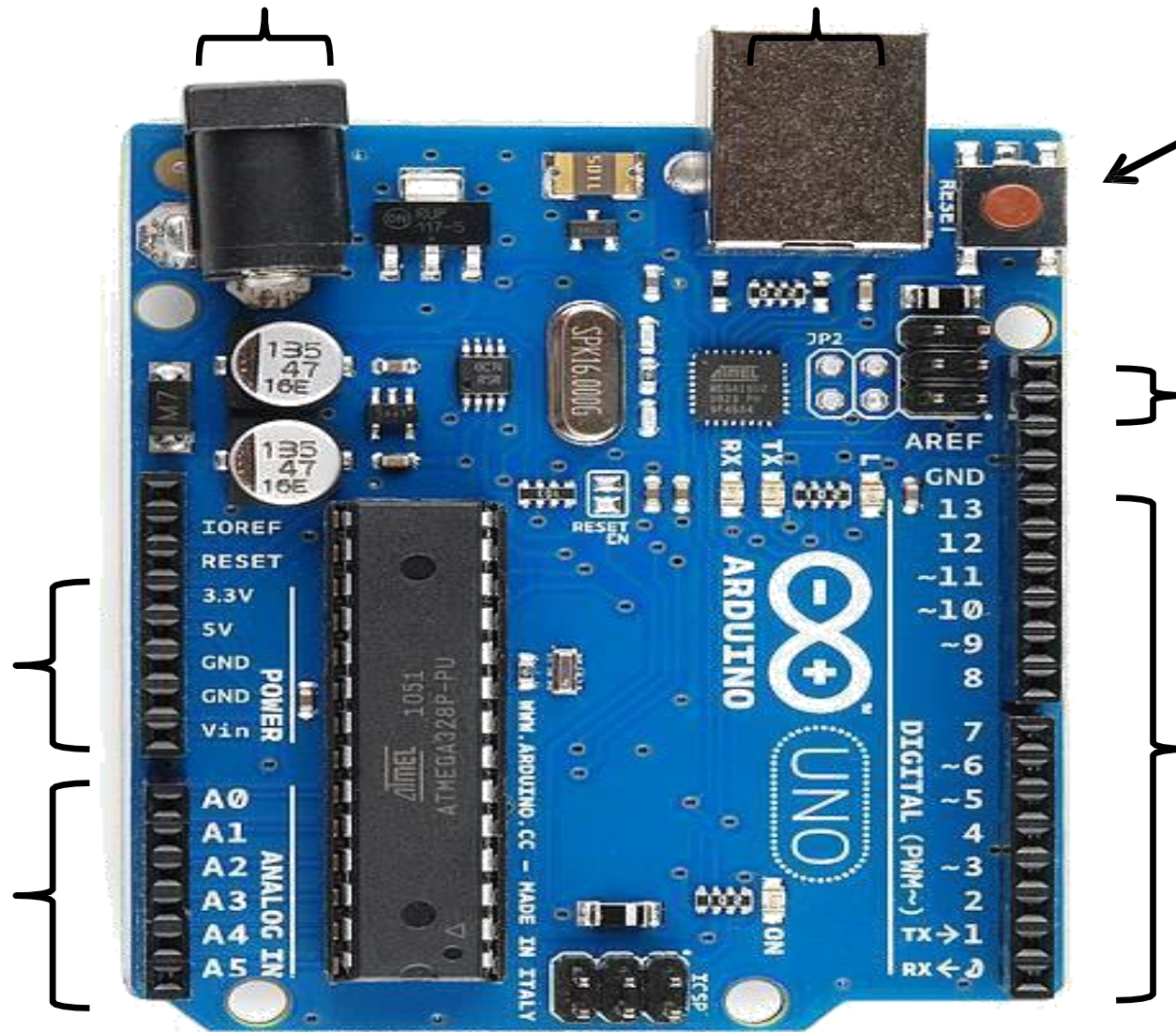
PWRIN

USB
(to Computer)

RESET

SCL\SDA
(I2C Bus)

Digital I\O
PWM(3,5,6,9,10,11)



ATmega328 Microcontroller to Arduino Pin Mapping



ATmega328 AVR
Microcontroller

General Specifications of ATmega328 Microcontroller

Device Name	ATmega328P
Flash memory	32 Kbytes
SRAM	2 Kbytes
EEPROM	1 Kbytes
Pin Count	28 Pins
General purpose I/O lines	23 I/O pins
Max. Operating Frequency	16 MHz
CPU	8-bit AVR
General purpose working registers	32
Number of ADC Inputs	6-channel 10-bit A/D converter
Timers	3
Interface Type	I2C, SPI, USART
Temp. Range (deg C)	-40 to 85 oC
Operating Voltage (Vcc)	1.8-5.5 volts.

ATga328 Microcontroller to Arduino Pin Mapping

ATMEGA328P-PU Chip to Arduino Pin Mapping

Arduino function										Arduino function
reset	(PCINT14/RESET) PC6	1		28	PC5 (ADC5/SCL/PCINT13)					analog input 5
digital pin 0 (RX)	(PCINT16/RXD) PD0	2		27	PC4 (ADC4/SDA/PCINT12)					analog input 4
digital pin 1 (TX)	(PCINT17/TXD) PD1	3		26	PC3 (ADC3/PCINT11)					analog input 3
digital pin 2	(PCINT18/INT0) PD2	4		25	PC2 (ADC2/PCINT10)					analog input 2
digital pin 3 (PWM)	(PCINT19/OC2B/INT1) PD3	5		24	PC1 (ADC1/PCINT9)					analog input 1
digital pin 4	(PCINT20/XCK/T0) PD4	6		23	PC0 (ADC0/PCINT8)					analog input 0
VCC	VCC	7		22	GND					GND
GND	GND	8		21	AREF					analog reference
crystal	(PCINT6/XTAL1/TOSC1) PB6	9		20	AVCC					VCC
crystal	(PCINT7/XTAL2/TOSC2) PB7	10		19	PB5 (SCK/PCINT5)					digital pin 13
digital pin 5 (PWM)	(PCINT21/OC0B/T1) PD5	11		18	PB4 (MISO/PCINT4)					digital pin 12
digital pin 6 (PWM)	(PCINT22/OC0A/AIN0) PD6	12		17	PB3 (MOSI/OC2A/PCINT3)					digital pin 11(PWM)
digital pin 7	(PCINT23/AIN1) PD7	13		16	PB2 (SS/OC1B/PCINT2)					digital pin 10 (PWM)
digital pin 8	(PCINT0/CLKO/ICP1) PB0	14		15	PB1 (OC1A/PCINT1)					digital pin 9 (PWM)

Digital Pins 11, 12 & 13 are used by the ICSP header for MISO, MOSI, SCK connections (Atmega168 pins 17, 18 & 19). Avoid low-impedance loads on these pins when using the ICSP header.

ATmega328P Pins

- ATMEGA328P has 28 pins which 23 of them are general purpose input/output line and they are divided to three PORTs according to their function. The ports are; PORTB which is 8 pin (PB0 to PB7), PORTD which is 8 pin as well (PD0 to PD7), and PORTC which is 7 pin (PC0 to PC6). The table below illustrates Atmega328p microcontroller pins function.

Atmega328p Microcontroller Pins Function.

Pin No.	Pin Function	Arduino Function	Pin No.	Pin Function	Arduino Function
1	PC6 (PCINT14/RESET)	Reset	15	PB1 (OC1A/PCINT1)	Digital pin 9 (PWM)
2	PD0 (PCINT16/RXD)	Digital pin 0 (RX)	16	PB2 (SS/OC1B/PCINT2)	Digital pin 10 (PWM)
3	PD1 (PCINT17/TXD)	Digital pin 1 (TX)	17	PB3 (MOSI/OC2A/PCINT2)	Digital pin 11 (PWM)
4	PD2 (PCINT18/INT0)	Digital pin 2	18	PB4 (MISO/PCINT4)	Digital pin 12
5	PD3 (PCINT19/OC2B/INT1)	Digital pin 3 (PWM)	19	PB5 (SCK/PCINT5)	Digital pin 13
6	PD4 (PCINT20/XCK/T0)	Digital pin 4	20	AVCC	VCC
7	VCC	VCC	21	AREF	Analog reference
8	GND	GND	22	GND	GND
9	PB6 (PCINT6/XTAL1/TOSC1)	Crystal	23	PC0 (ADC0/PCINT8)	Analog input 0
10	PB7 (PCINT7/XTAL2/TOSC2)	Crystal	24	PC1 (ADC1/PCINT9)	Analog input 1
11	PD5 (PCINT21/OC0B/T1)	Digital pin 5 (PWM)	25	PC2 (ADC2/PCINT10)	Analog input 2
12	PD6 (PCINT22/OC0A/AIN0)	Digital pin 6 (PWM)	26	PC3 (ADC3/PCINT11)	Analog input 3
13	PD7 (PCINT23/AIN1)	Digital pin 7	27	PC4 (ADC4/PCINT12)	Analog input 4
14	PB0 (PCINT0/CLKO/ICP1)	Digital pin 8	28	PC5 (ADC5/PCINT13)	Analog input 5

Digital Inputs (Sensors)

All digital inputs should be connected to digital pins (digital pins 0 – 13 of Arduino pin mapping) which can read digital data. The position of item within the microcontroller depends on declaring such item in the programming of a microcontroller. For example, switches can be connected to any pins of the microcontroller as need.

Digital Outputs (Actuators)

At the same time, digital pins (D0 – D13) can be used as digital input and digital output as well. The analog pins (A0 – A5) can be used as digital pins. Those depend on declaring those pins in the programming software.

Analog Inputs (Sensors)

All analog items should connect to analog pins (A0 – A5 which are pins 23 – 28 of Atmega328p). Then, the analog data can be converted to digital using ADC (Analog to Digital Converter) to be manipulated in the microcontroller. Analog inputs include: Potentiometer, and Thermistor.

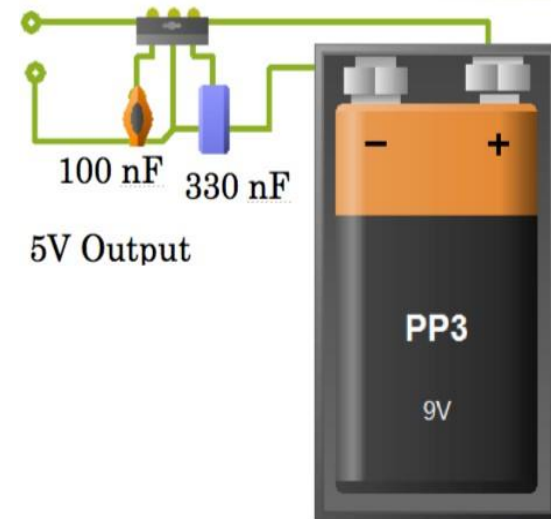
Analog Output (Actuators)

- The Arduino does not have a built-in digital- to-analog converter (DAC), but it can pulse- width modulate (PWM) a digital signal to achieve some of the functions of an analog output. The function used to output a PWM (pulse with modulation) signal is `analogWrite(pin, value)`. `pin` is the pin number used for the PWM output.

Arduino Power Supply

The power supply can be any voltage between 7 and 12 volts. So a small 9V battery will work just fine for portable applications.

Typically, while you are making your project, you will probably power it from USB for convenience.



Arduino Software

Arduino is an open-source environment which makes it easy to write code and upload it to the Arduino development board. It operates on Windows, Mac OS X, and Linux.

Arduino Software

- The Arduino language is a modified of the C++ programming language, but it uses built-in libraries to simplify complex coding jobs to make it easier for users to pick up
- Arduino Software written is called **sketches**. These sketches are written in the text editor, which are saved with the file extension .ino.

Arduino Integrated Development Environment

The open-source IDE (Integrated Development Environment) is split into three parts: the Toolbar across the top, the code or Sketch Window in the center, and the messages window in the bottom.

Arduino Integrated Development Environment

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



Arduino IDE 2.3.6

[Release notes](#)

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger. For more details, check the [Arduino IDE 2.0 documentation](#).

Windows Win 10 or newer (64-bit) [DOWNLOAD](#)

Nightly Builds

Download a preview of the incoming release with the most updated features and bugfixes.

The Arduino IDE 2.0 is open source and its source code is hosted on [GitHub](#).

Legacy IDE (1.8.19)

Download a legacy version of the Arduino IDE.

Windows Win 7 and newer [DOWNLOAD](#)

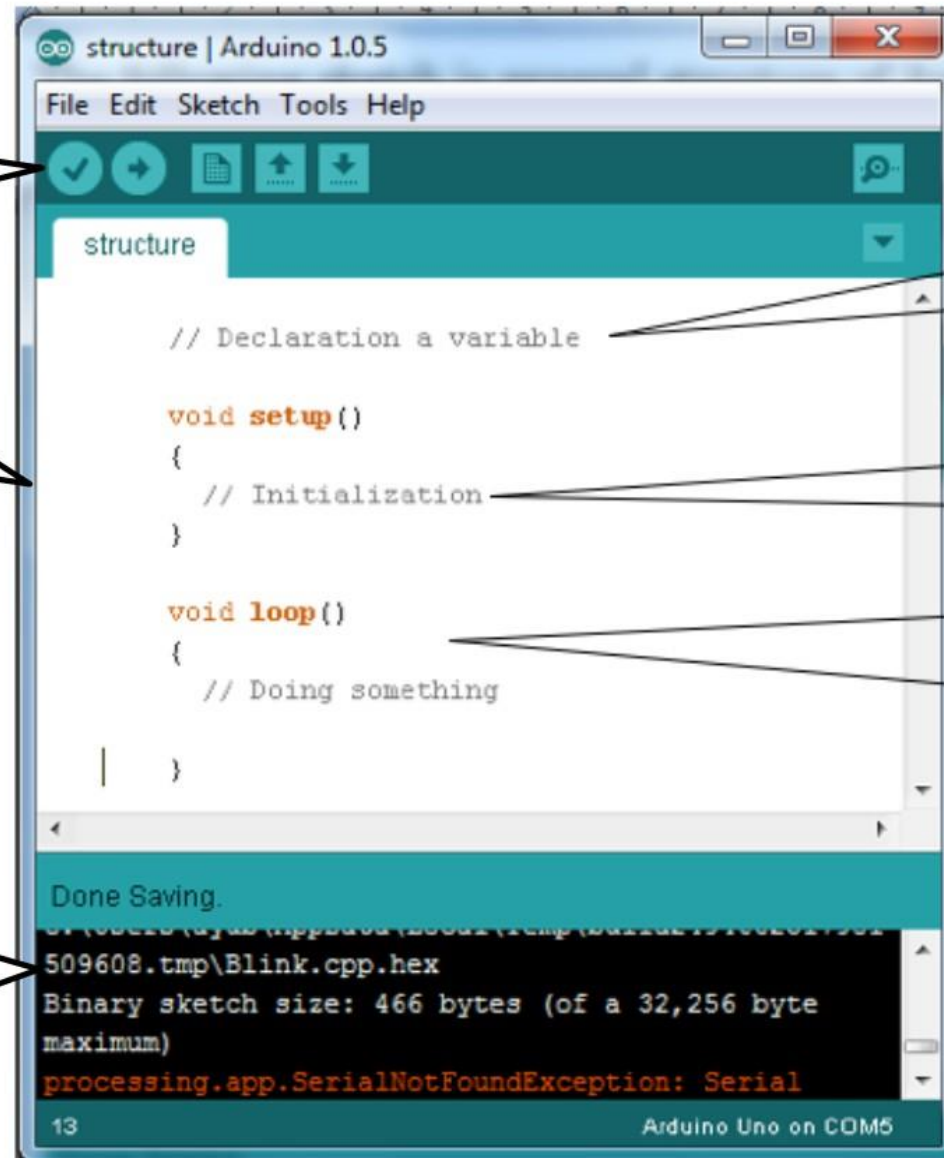


The Arduino IDE

Toolbar

Sketch

Messages window

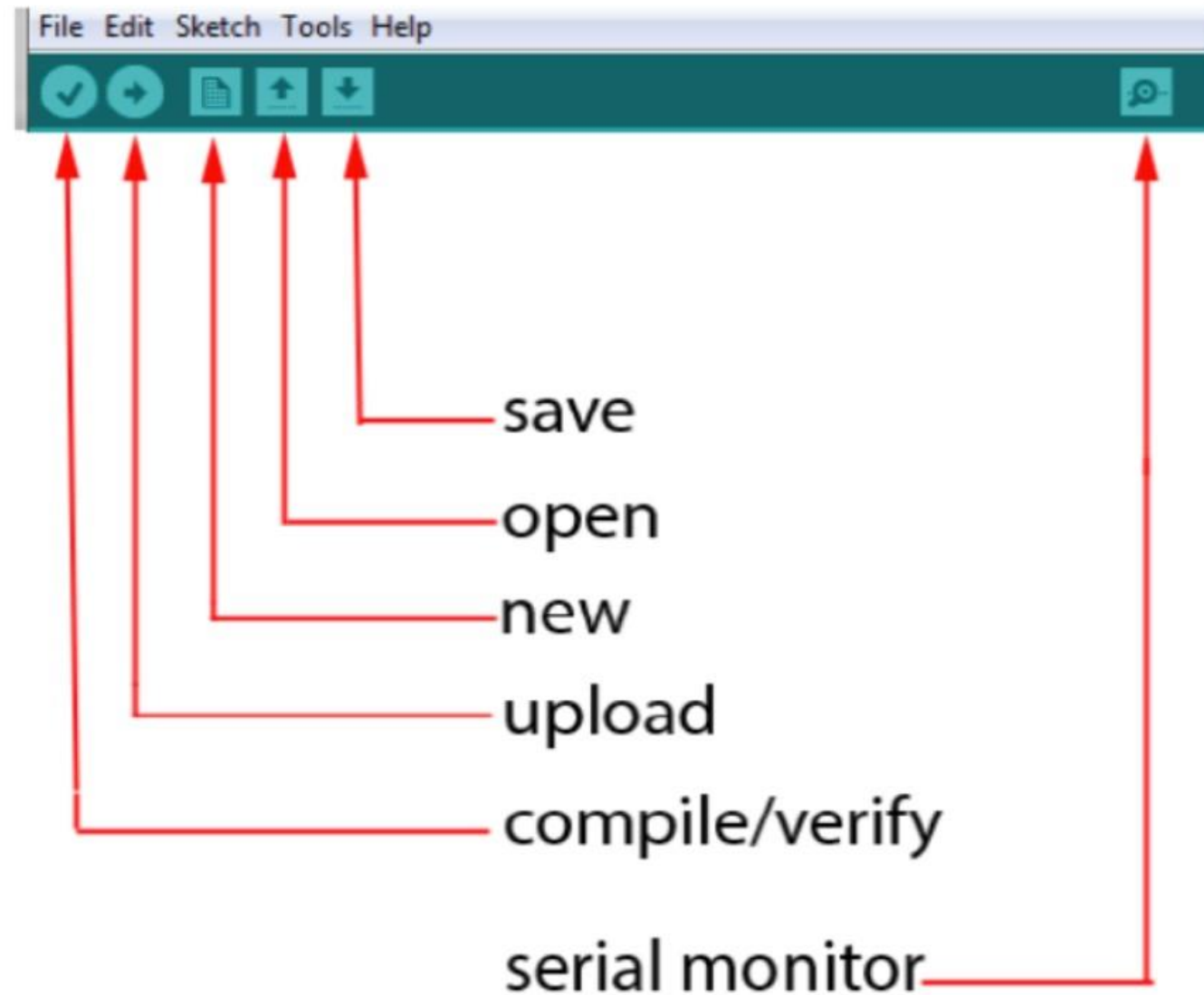


It is used to declare variables

Initialization of pins function

The loop routine runs over and over again forever:

The Arduino IDE



Arduino IDE

Verify: Checks your code for errors (Compile).

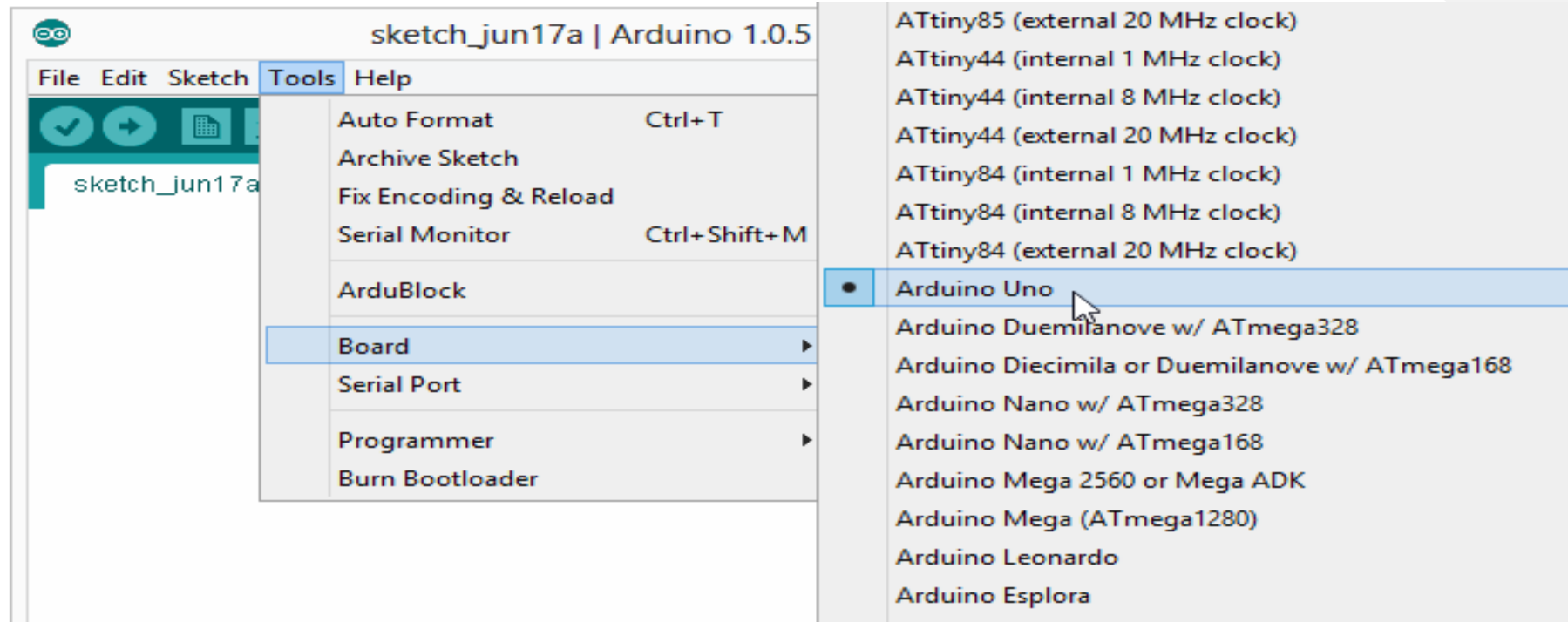
Upload: Compiles your code and uploads it to the Arduino board.

New: Creates a new sketch.

Selecting Board Type and Port

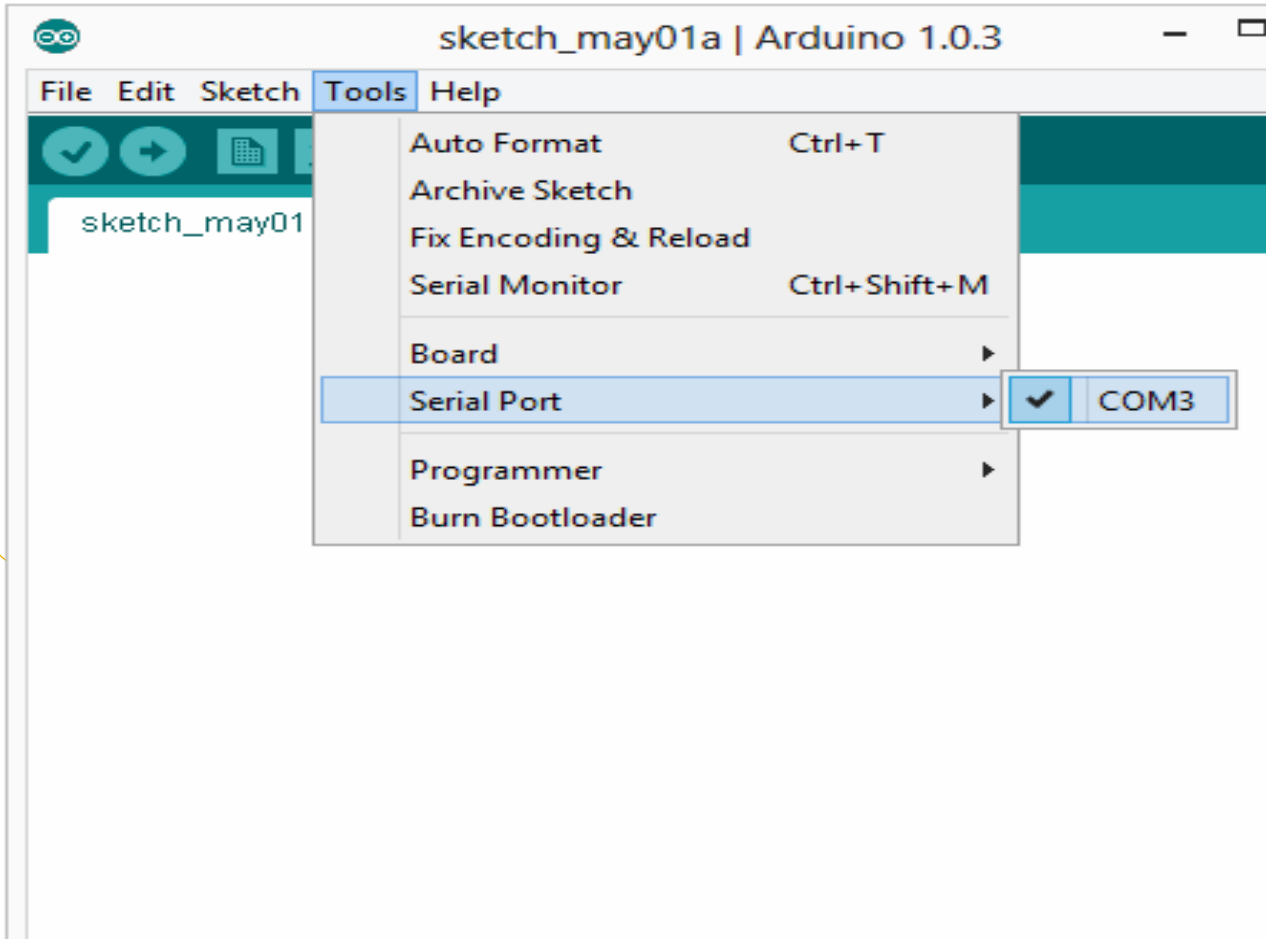
- You'll need to select the entry in the **Tools > Board** menu that corresponds to your Arduino or Genuino board.
- Select the serial device of the board from the
- **Tools > Serial Port** menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports).

Settings: Tools → Board



- Next, double-check that the proper board is selected under the Tools→Board menu.

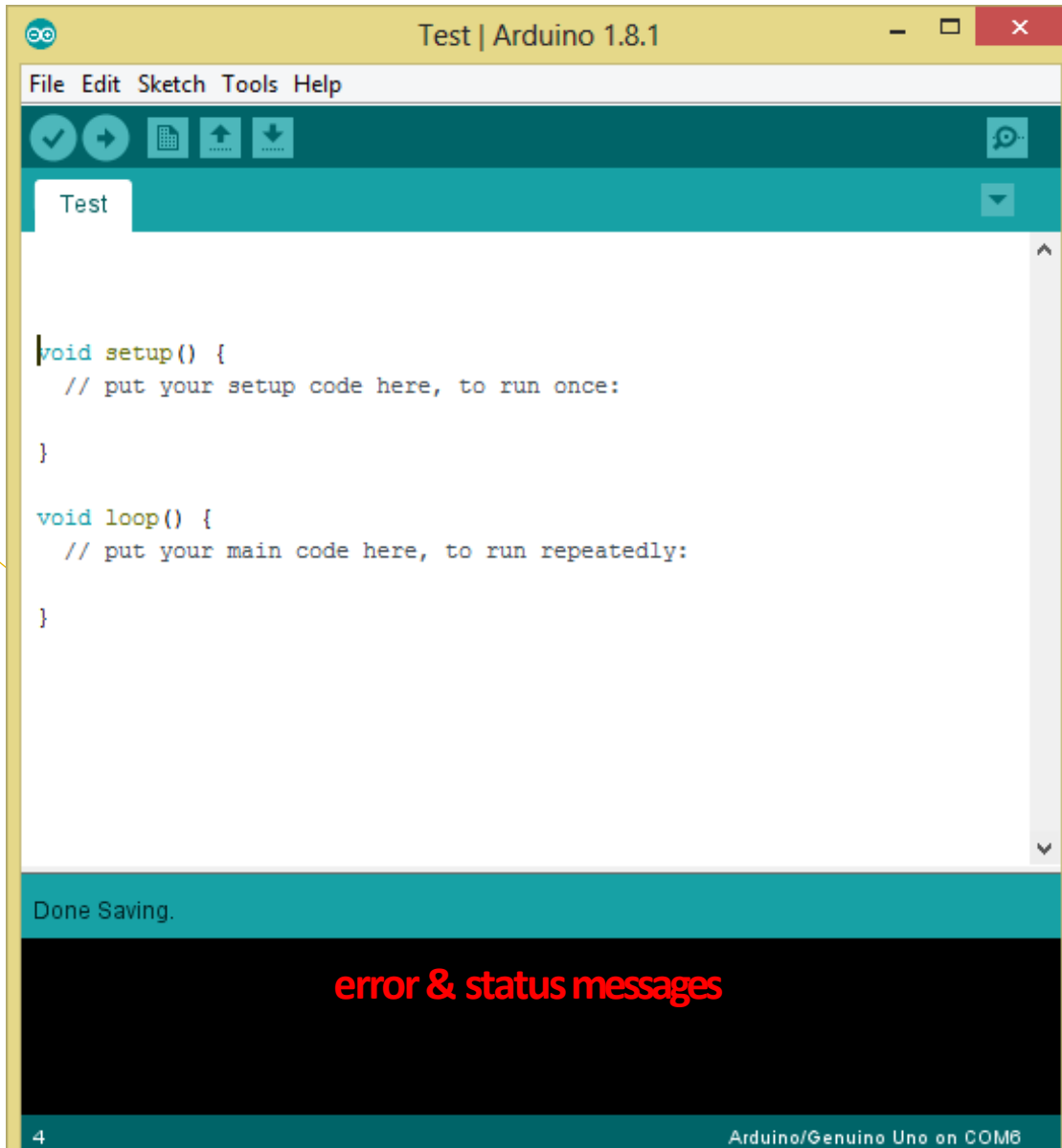
Settings: Tools → Serial Port



- Your computer communicates to the Arduino microcontroller via a serial port → through a USB-Serial adapter.
- Check to make sure that the drivers are properly installed.

Arduino

Integrated Development Environment (IDE)



Two required functions / methods / routines:

```
void setup()  
{  
    // runs once  
}
```

```
void loop()  
{  
    // repeats  
}
```

Arduino code

```
void setup(){
```

```
    // put your setup code here, to run once:
```

```
}
```

```
void loop(){
```

```
    // put your main code here, to run repeatedly:
```

```
}
```

Arduino code

- **setup** : It is called only when the Arduino is powered on or reset. It is used to initialize variables and pinmodes
- **loop** : The loop functions runs continuously till the device is powered off. The main logic of the code goes here. Similar to while (1) for micro-controller programming.

Pin Mode

- A pin on arduino can be set as input or output by using pinMode function.
- `pinMode(13, OUTPUT);` // sets pin 13 as output pin
- `pinMode(13, INPUT);` // sets pin 13 as input pin

Pin Mode

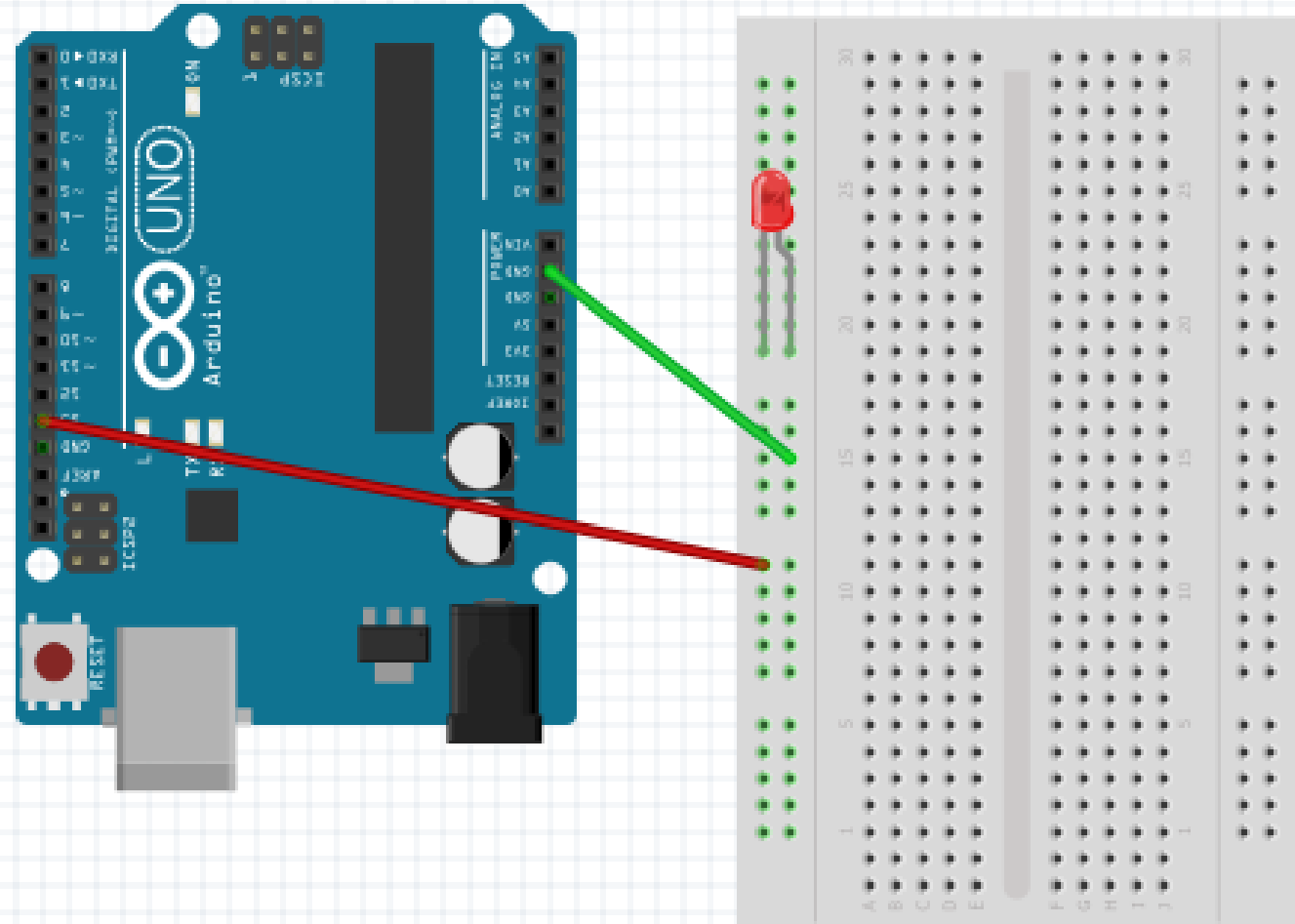
- **Int Outpin = 13;**
- **pinMode(Outpin , OUTPUT);**
- **Int Inpin = 12;**
- **pinMode(Inpin, INPUT);**

Reading/writing digital values

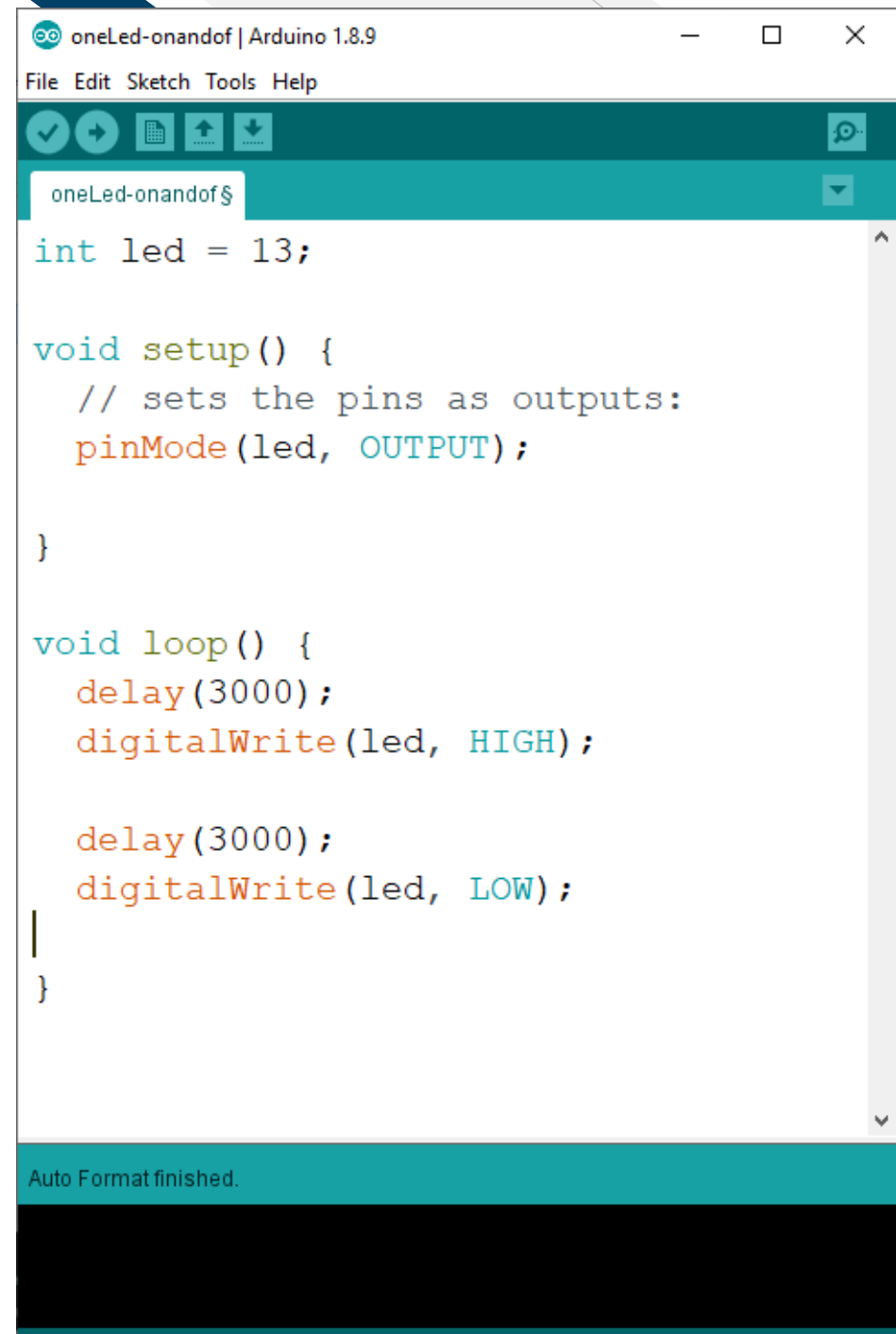
- `digitalWrite(13, LOW);` // Makes the output voltage on pin 13, 0V
- `delay(5000);` // Makes the sleep
- `digitalWrite(13, HIGH);` // Makes the output voltage on pin 13, 5V

LED Example

- Use :
- Pin 13
- GND



Send power to pin **13** after **3** seconds switch off power from it and this process continuous ...(on and off)



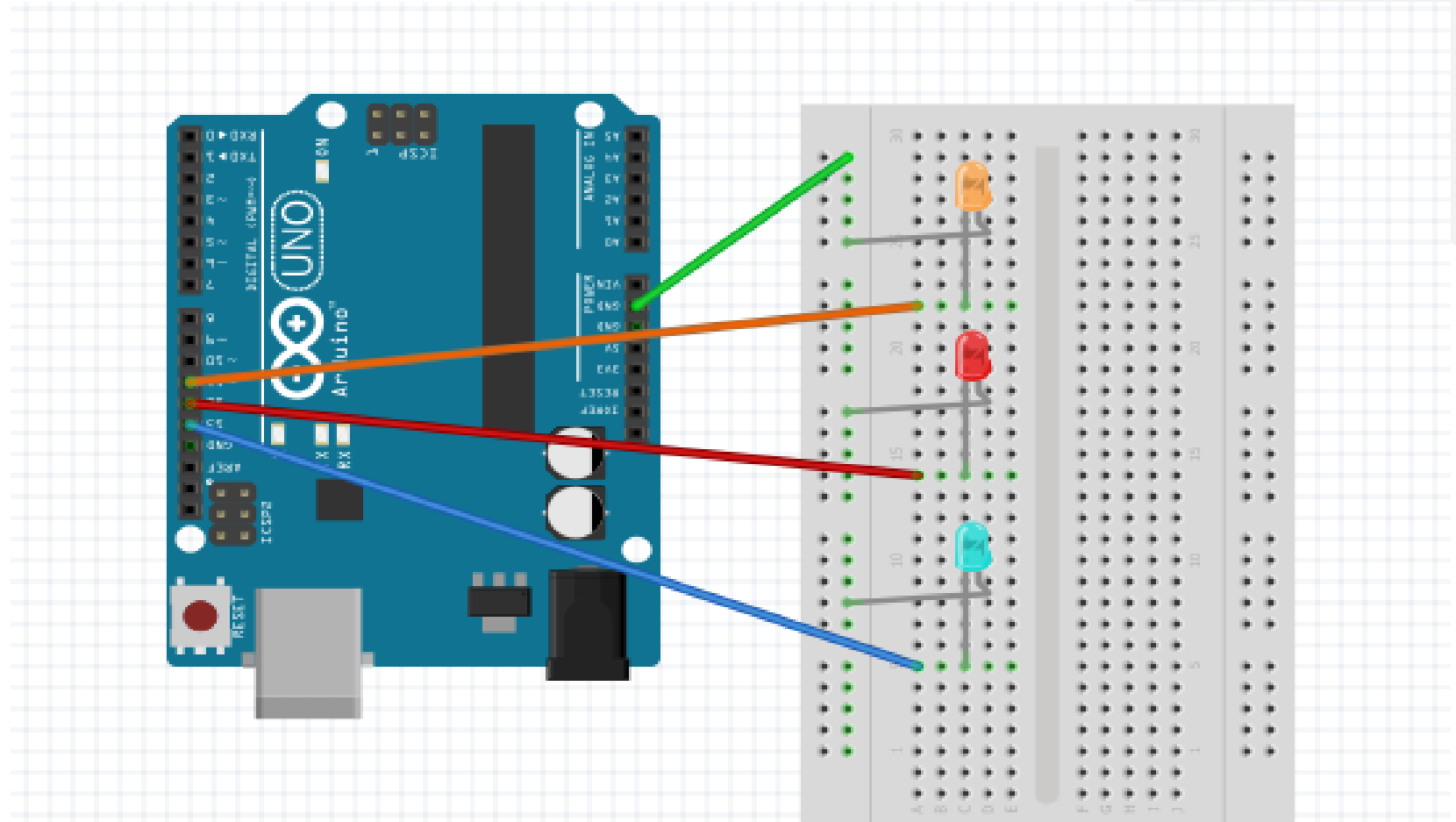
The screenshot shows the Arduino IDE interface with a sketch titled "oneLed-onandof | Arduino 1.8.9". The code is as follows:

```
oneLed-onandof$  
  
int led = 13;  
  
void setup() {  
  // sets the pins as outputs:  
  pinMode(led, OUTPUT);  
}  
  
void loop() {  
  delay(3000);  
  digitalWrite(led, HIGH);  
  
  delay(3000);  
  digitalWrite(led, LOW);  
}
```

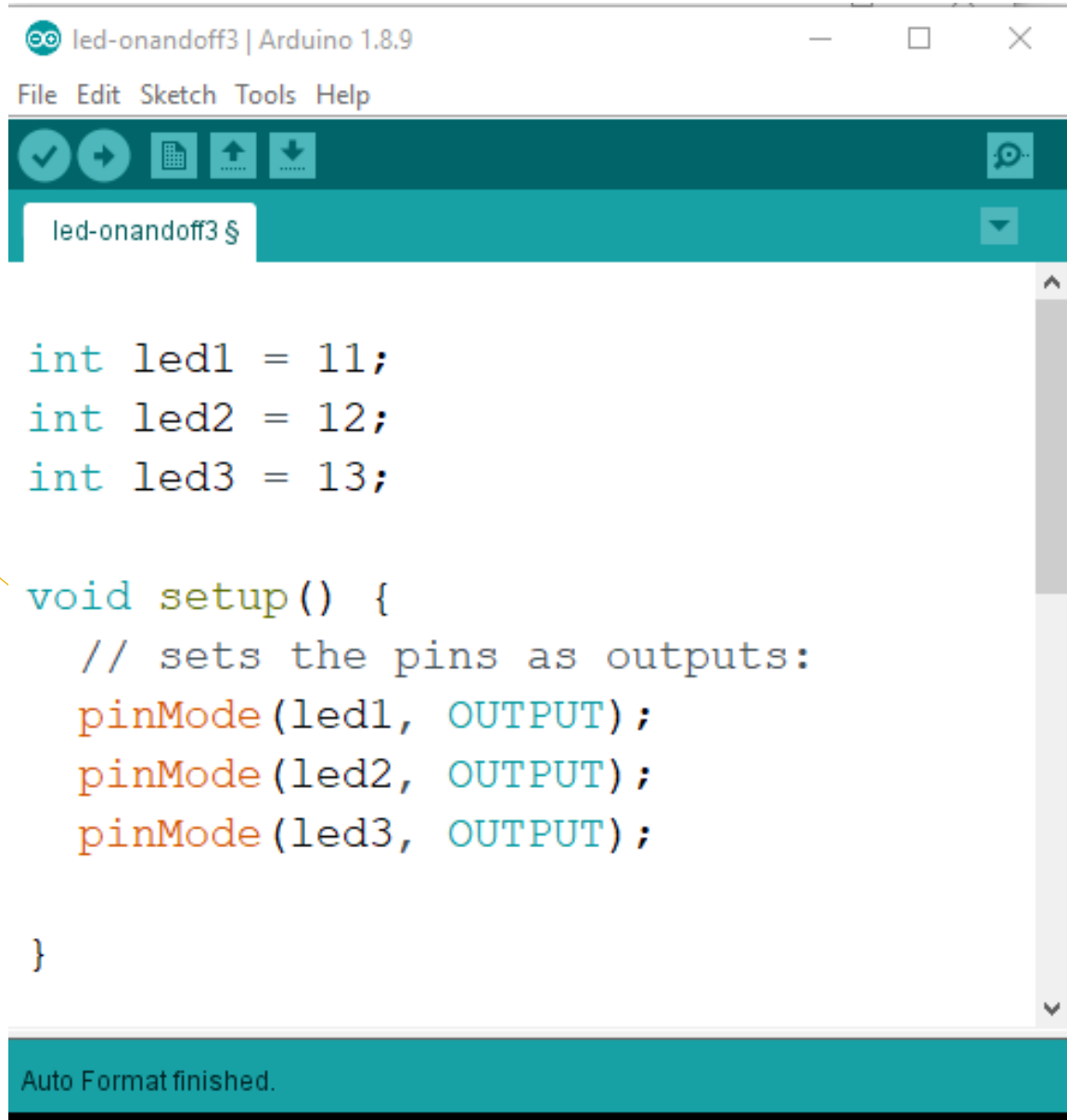
At the bottom of the IDE, a status bar indicates "Auto Format finished."

3 LED Example : Traffic

- Use :
- Pin 11
- Pin 12
- Pin 13
- GND



Part 1



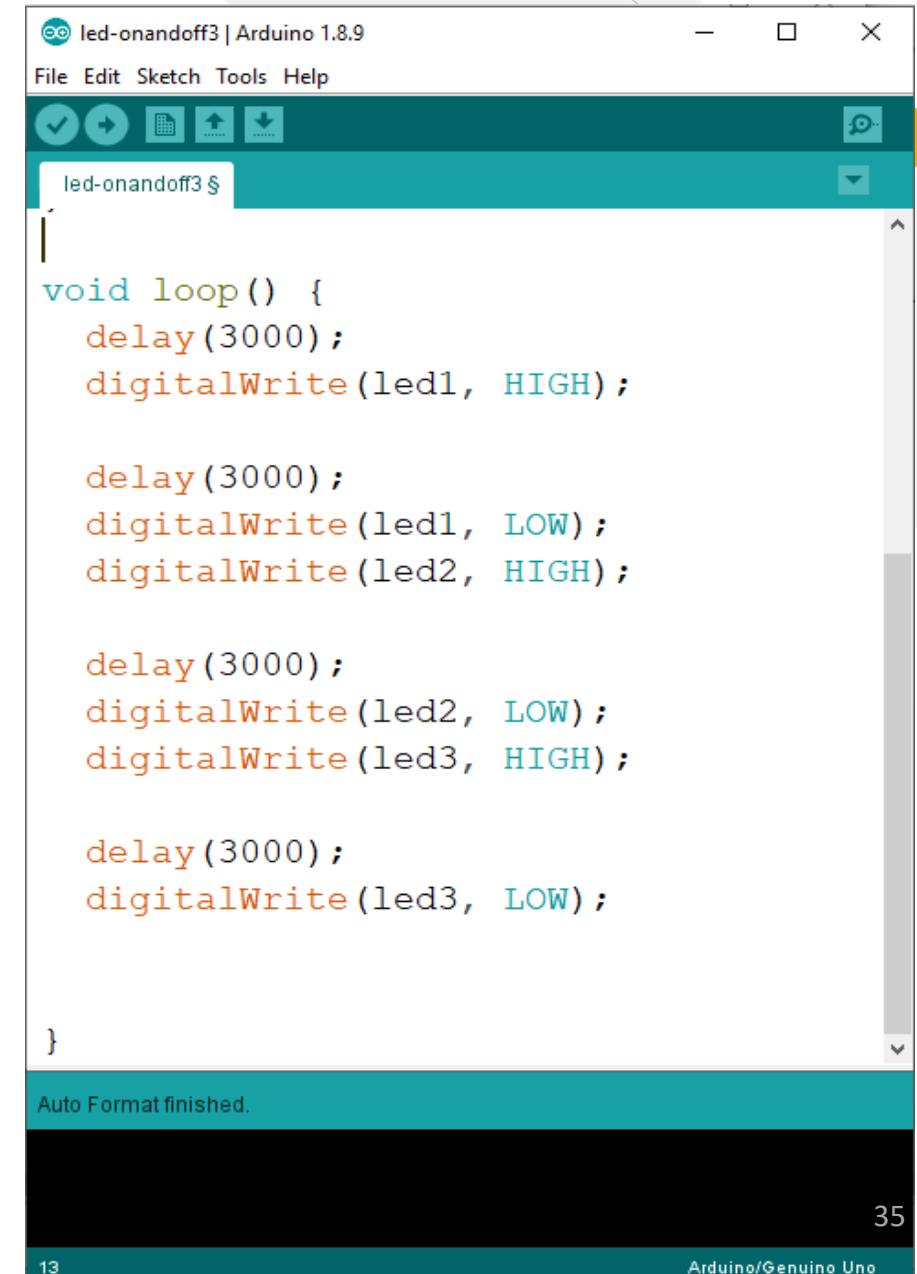
The screenshot shows the Arduino IDE window titled "led-onandoff3 | Arduino 1.8.9". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar contains icons for checking, running, serial monitor, upload, and download. The sketch name "led-onandoff3" is in the top bar. The code in the editor is as follows:

```
int led1 = 11;
int led2 = 12;
int led3 = 13;

void setup() {
  // sets the pins as outputs:
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(led3, OUTPUT);
}
```

A status bar at the bottom indicates "Auto Format finished."

Part 2



The screenshot shows the Arduino IDE window titled "led-onandoff3 | Arduino 1.8.9". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar contains icons for checking, running, serial monitor, upload, and download. The sketch name "led-onandoff3" is in the top bar. The code in the editor is as follows:

```
void loop() {
  delay(3000);
  digitalWrite(led1, HIGH);

  delay(3000);
  digitalWrite(led1, LOW);
  digitalWrite(led2, HIGH);

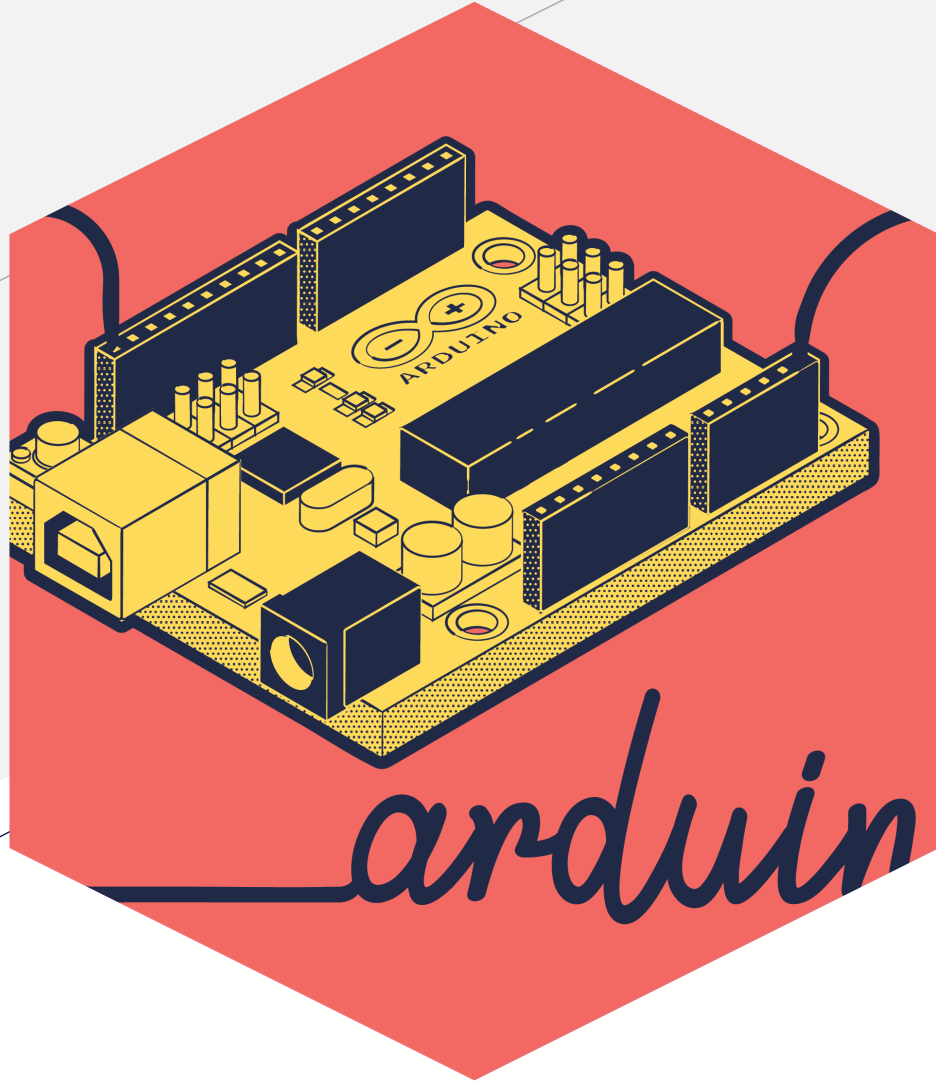
  delay(3000);
  digitalWrite(led2, LOW);
  digitalWrite(led3, HIGH);

  delay(3000);
  digitalWrite(led3, LOW);
}
```

A status bar at the bottom indicates "Auto Format finished."

Reference

- Embedded Controllers Using C and Arduino by James M. Fiore ,Version 2.1.10, 07 May 2021.
- Tianhong Pan . Yi Zhu ,Designing Embedded Systems with Arduino A Fundamental Technology for Makers © Springer Nature Singapore Pte Ltd. 2018.
- D.P. Kothari, K.V. Shiram, Sundaram. Embedded Systems. New age international, 2012 .
- <https://builtin.com/robotics>
- <https://robots.ieee.org/learn/types-of-robots/>
- <https://www.arduino.cc/en/guide/introduction>



Thank You.