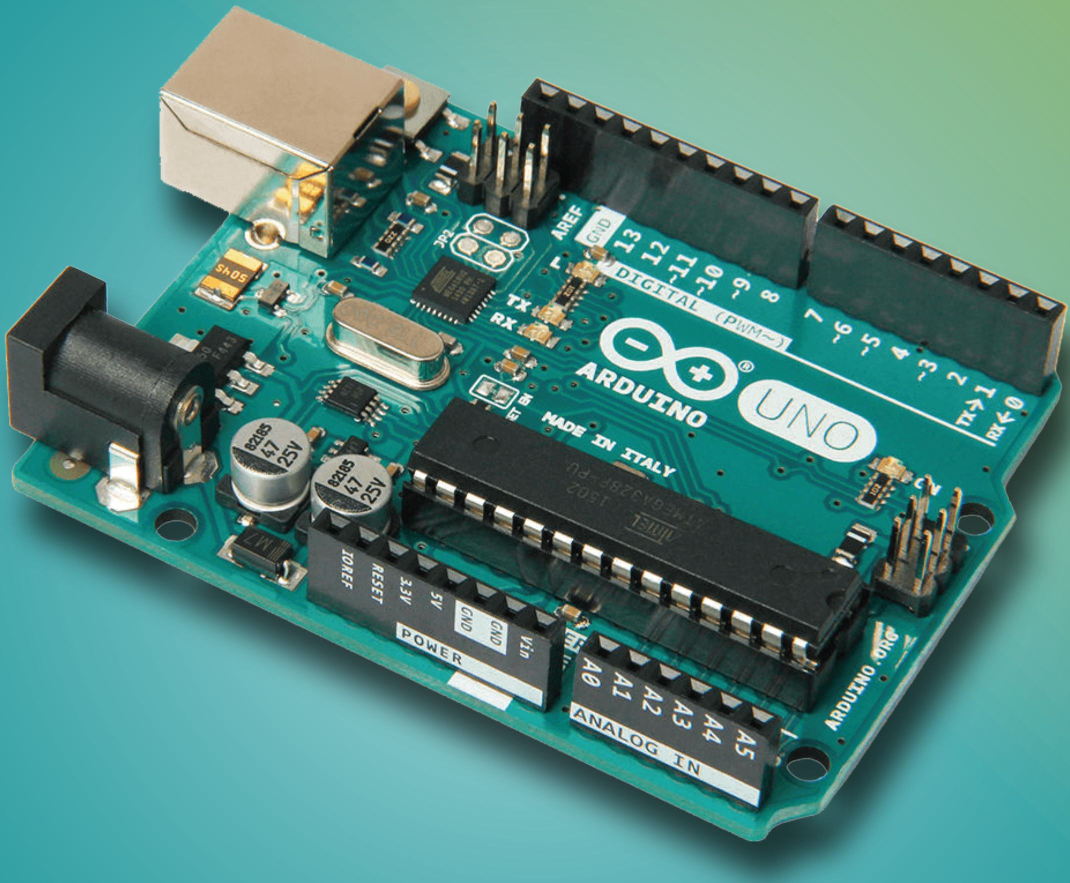
Arduino
Projects' Book

ARDUINO



Level 2

Rana Dajani

Arduino Projects' Book - Level 2

Published by **LKD Educational Resources 2023**

Amman - Jordan Tel: +962 6 5374141

Fax: +962 6 5516404 P.O.Box: 851346

Email: info@lkd.com.jo / info@ lkdebooks.com

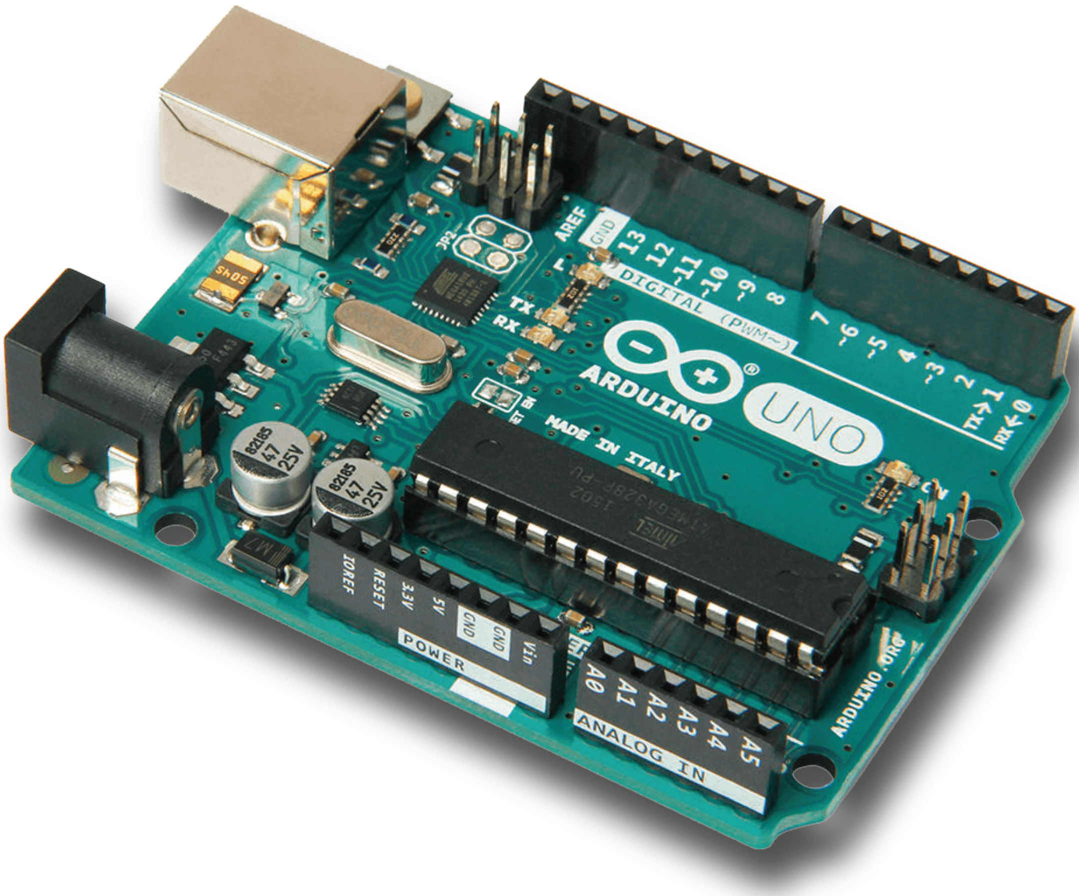
Websites: www.lkd.com.jo / www.lkdebooks.com

Author

Rana Dajani

ISBN: **2023/9/4937**

Arduino
Projects' Book



Level 2

Rana Dajani

Introduction

This book is the second level continuation of Arduino Projects' Book – Level 1 that provides an introduction to the basics of electronic circuits and programming using the Arduino C++ language.

In Arduino Projects' Book – Level 1, learners are introduced to the main circuit components such as, the Arduino microprocessor, bread boards, jump wires and resistors. As well as, a set of input and output components (LEDs, Piezo buzzers, Pushbuttons, Slide switches, Tilt sensors, PIR Motion sensors, Potentiometers and LCD screens).

Here is a summary for a quick review of the Arduino board and its commands:

The Arduino board has **12 digital pins (2-13) and 6 analog pins (A0-A5).**

The digital pins signal HIGH/1 or LOW/0 values through digitalWrite() or digitalRead() commands.

Some of the digital pins have a **PWM (~)** symbol that stands for pulse width modulation which allow you to send a certain value from the range (0-255) to a component using the analogWrite() command.

The analog pins allow you to recieve a range of voltage values passing through a component (0 – 1023) as input, using the analogRead() command.

Arduino systems inspire you to create devices that can interact with the world around you. By using an almost unlimited range of input and output devices, sensors, indicators, displays, motors, and more, you can program the exact interactions required to create a functional device.

The topics covered by the projects in this book are divided into two comprehensive themes: smart car and smart house, with ten lessons to complete for each theme.

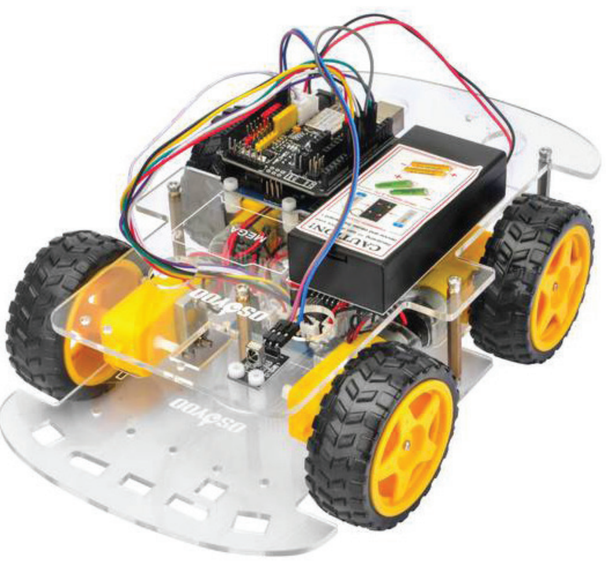
The table of contents has been color-coded according to various lesson types:

- An online virtual simulator can be used to test and work on the first five projects for each theme.

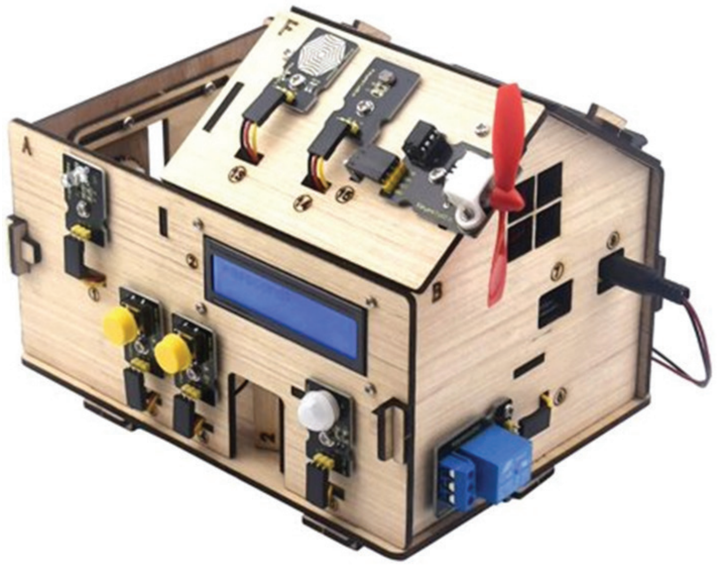
- You must physically handle the kit's components to complete the next four projects.

- The tenth and final projects are open-ended and requires you to use the skills you have learned from the previous lessons to create a completely developed project on the subject.

Smart Car

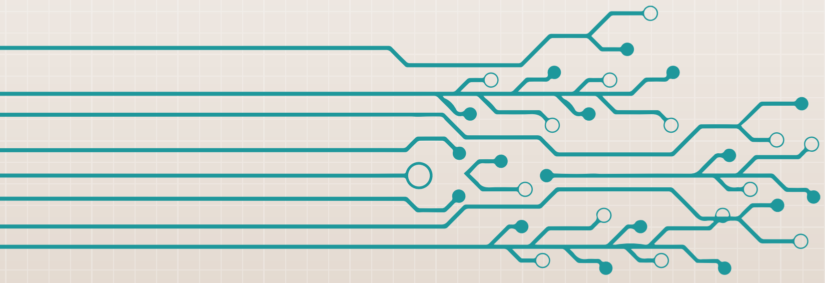


Smart House



Smart Car

Smart Car



Lesson

RGB LED

1

Review and Expand

Thislessonallowsyoutoreviewandwarmupyourpervious knowledge while controlling a new component, the multi-color LED (RGB LED).

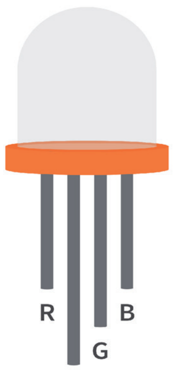
An RGB LED displays colours, by a combination of red, green and blue LEDs (light emitting diodes). Mixing these three colours in different levels of intensity can create almost any colour of light.

You have previously come across arrays in C++ to create a list of elements. C++ also allows multidimensional arrays.

The simplest form of the multidimensional array is the two-dimensional array, which can be thought of as a table of rows and columns.

Every element in an array is identified by its indices. In a 2D array the first index will indicate the row number and another index will indicate the column, in this format:

type arrayName [row][column];



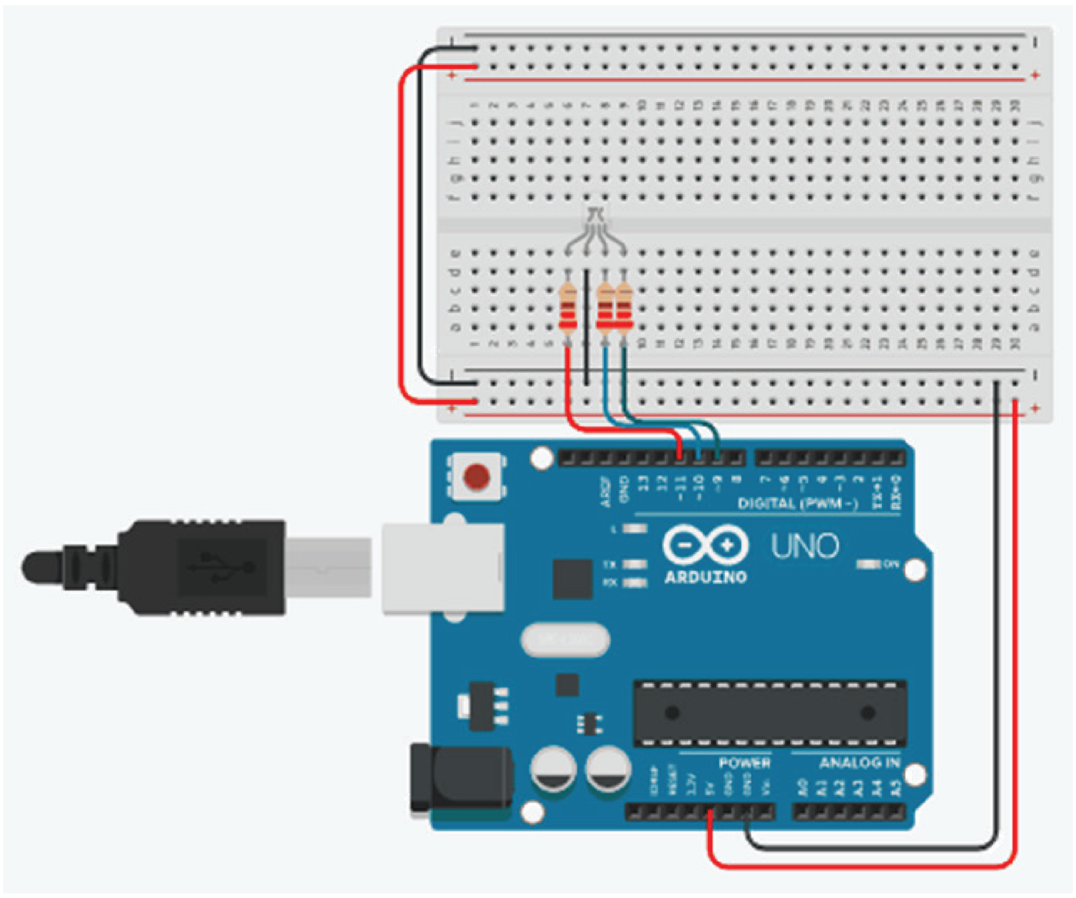
RGB LED
The RGB LED consists of three different LEDs, with a terminal for each and one common cathodes terminal

The type should be declared first, which will specify the type of data that will be saved into the array (e.g. int)

	Column 0	Column 1	Column 2	Column3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Build your circuit

Connect the cathode of the RGB LED which is the longer pin to the GND of Arduino and the other three pins to the PWM pins 9,10,11 of Arduino through the 220 ohm resistors. The resistors will prevent the excess amount of current to flow through the RGB LED.



The red, green and blue colours each use integer values from 0 to 255, which makes 256*256*256=16777216 possible colors.

The Arduino has an analogWrite() function which will help you obtain different colors for the RGB LED with the use of the PWM pins.

Build your program

The aim of this lesson is to program the RGB LEDs to turn on one colour of the rainbow after the other with 500 millisecond delay time and to create a rainbow pattern.

The pseudo code would be:

1. Declare variables that will hold the digital pin numbers to which the R, G, B, LED's terminals are connected
2. Declare a two-dimensional array variable that will hold the 7 colours of the rainbow

Inside the setup function:

3. Configure the digital pins that the LEDs are connected to as OUTPUTs.

Inside the loop function:

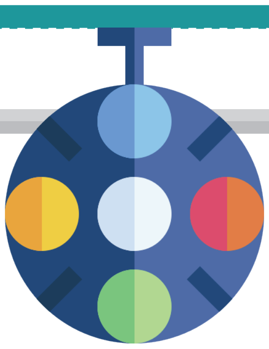
4. Use a for loop that will repeat for the number of rows in the table, using the loop's declared variable to reference the rows index, to:
    - analogWrite() to the red/green/blue pins their correct intensities respectively from the column indices of the table at every turn of every colour row
    - Add a delay time (500) between every colour

That translates to code like this:

```
int R=11;
int B=10;
int G=9;

int rainbow[7][3]={ {255,0,0}, //red
                    {255,70,200}, //orange
                    {148,0,211}, //yellow
                    {0,0,255}, //green
                    {0,191,255}, //blue
                    {20,255,100}, //navy
                    {255,255,0}, //violet
                  };

void setup(){
 pinMode(R, OUTPUT);
 pinMode(B, OUTPUT);
 pinMode(G, OUTPUT);
}

void loop(){
 for (int i =0; i<7; i++){
  analogWrite(R, rainbow[i][0]);
  analogWrite(B, rainbow[i][1]);
  analogWrite(G, rainbow[i][2]);
  delay(500);
 }
}
```

Take it further!

Challenge



Disco Lights

Use two RGB LEDs to display random colours to create a disco!

You can tweak it further to make it more interesting.

NOTE: recall the random() function

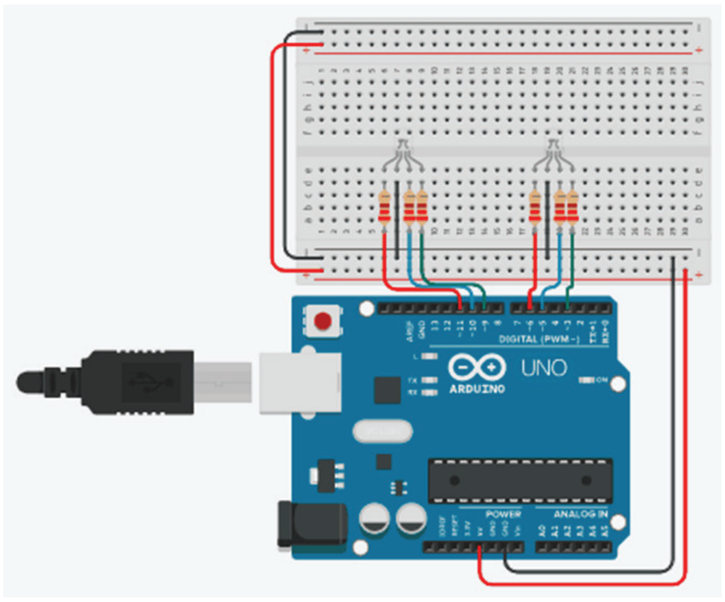
Hints:

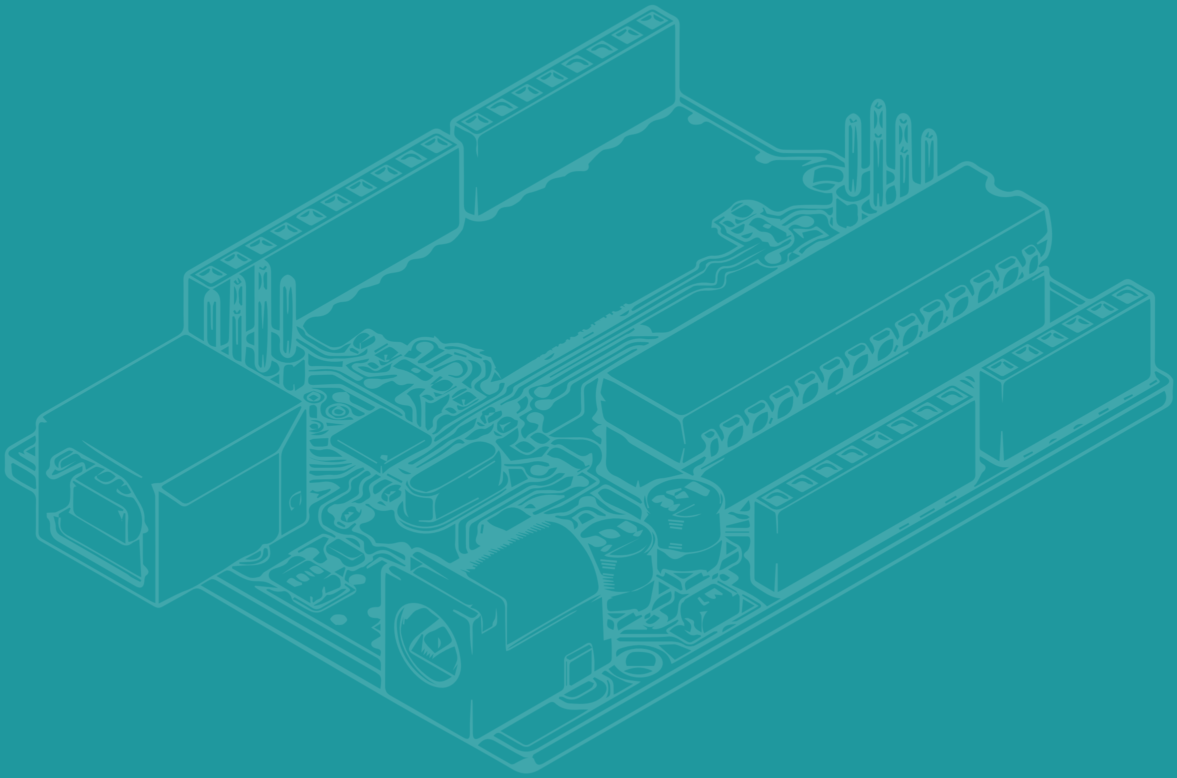
The random() function generates pseudo-random numbers.

- min: lower bound of the random value, inclusive (optional).
- max: upper bound of the random value, exclusive.

Syntax: random(min, max)

e.g. ___ random (2,9); // the random number generated is between 2 and 9, two is included in the range but nine is not.





ISBN: 2023/9/4937

ARDUINO