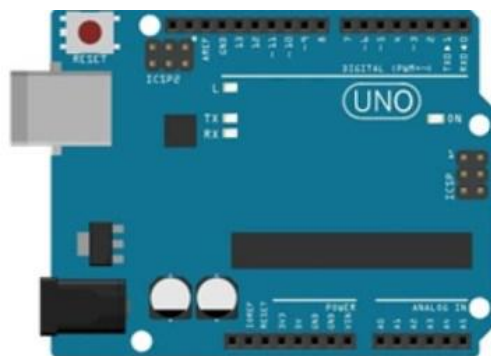


Arduino Compatible Board

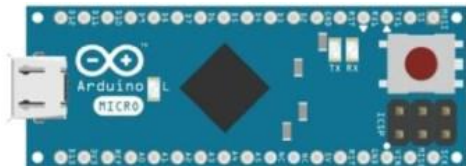
Arduino Compatible Board is a circuit board, which integrates micro controller, input, output interface and etc. Arduino Board can use the sensor to sense the environment and receive user's operation to control LED, motor rotation, etc. We just need to assembly circuit and write the code.

Currently, Arduino Board has several models, and the code between boards of different types is universal (some boards may not be completely compatible because of the differences in hardware). Popular boards include:

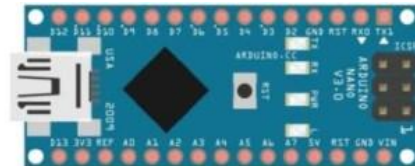
UNO R3 Board:



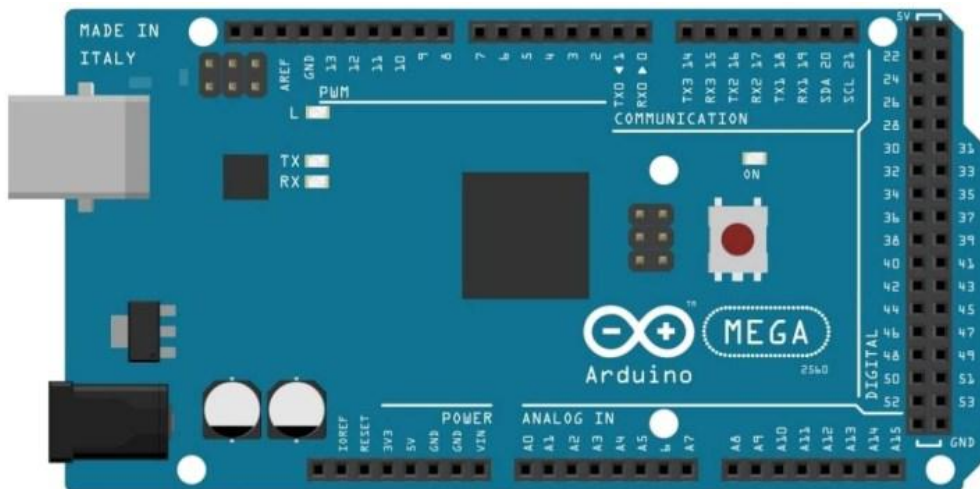
MICRO

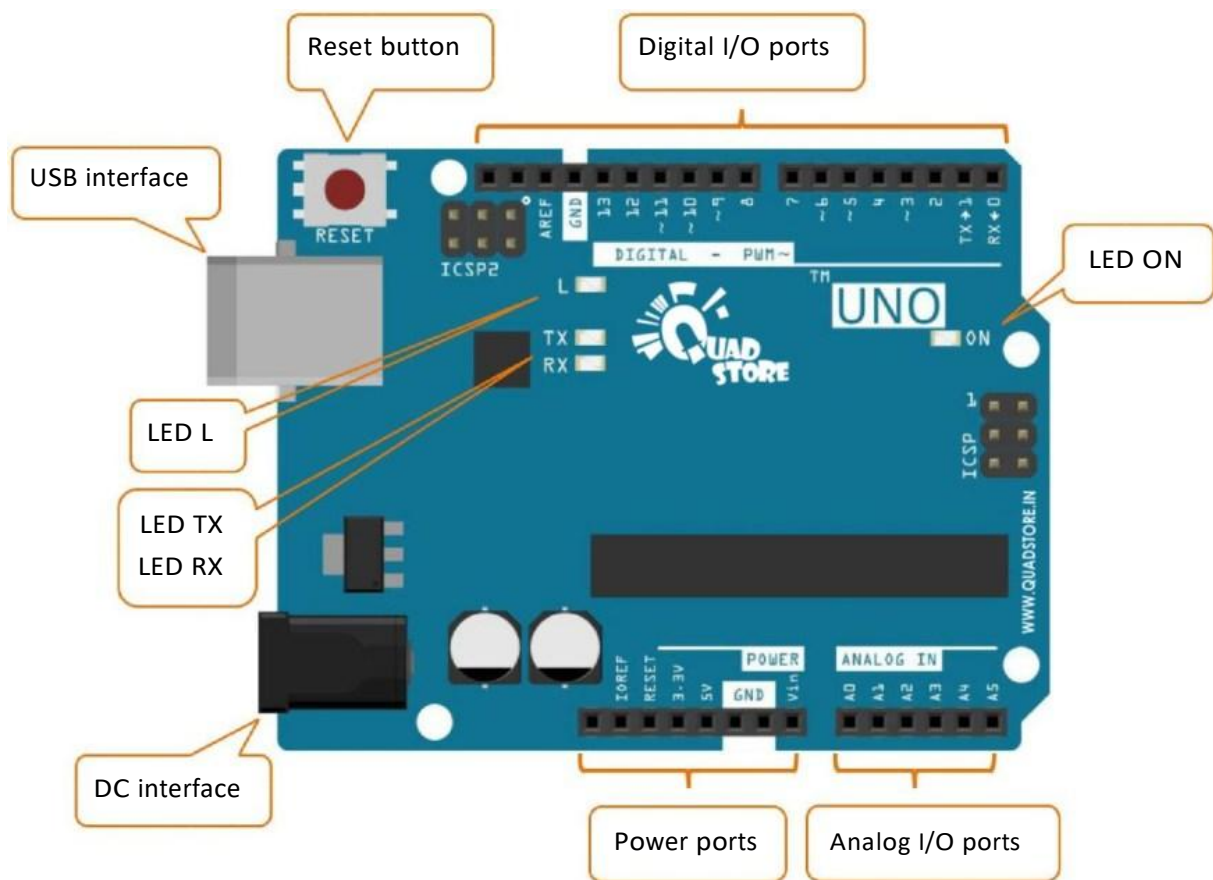


NANO



MEGA





Contents

- Getting Started with Arduino
- Installing Arduino IDE and using the Uno R3 board
- About Uno R3 compatible Arduino board
- Lesson-1 Blinking LED
- Lesson-2 RGB LED
- Lesson-3 Push Button Switch
- Lesson-4 Photoresistor or LDR
- Lesson-5 Buzzer
- Lesson-6 7-Segment Display
- Lesson-7 Potentiometer
- Lesson-8 Tilt Sensor
- Lesson-9 Servo Motor

Getting Started with Arduino

What is an Arduino?

Arduino is an open-source physical computing platform designed to make experimenting with electronics more fun and intuitive. Arduino has its own unique, simplified programming language, a vast support network, and thousands of potential uses, making it the perfect platform for both beginner and advanced DIY enthusiasts.

www.arduino.cc

A Computer for the Physical World:

The friendly blue board in your hand (or on your desk) is the Arduino. In some ways you could think of Arduino as the child of traditional desktop and laptop computers. At its roots, the Arduino is essentially a small portable computer. It is capable of taking inputs (such as the push of a button or a reading from a light sensor) and interpreting that information to control various outputs (like a blinking LED light or an electric motor).

That's where the term "physical computing" is born - an Arduino is capable of taking the world of electronics and relating it to the physical world in a real and tangible way. Trust us - this will all make more sense soon.

Arduino UNO SMD R3

The Arduino Uno is one of several development boards based on the ATmega328. We like it mainly because of its extensive support network and its versatility. It has 14 digital input/output pins (6 of which can be PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. Don't worry, you'll learn about all these later.

Installing Arduino IDE and Using Uno R3 board

STEP-1: Download the Arduino IDE (Integrated Development Environment)

Access the Internet:

In order to get your Arduino up and running, you'll need to download some software first from www.arduino.cc (it's free!). This software, known as the Arduino IDE, will allow you to program the Arduino to do exactly what you want. It's like a word processor for writing programs. With an internet-capable computer, open up your favorite browser and type in the following URL into the address bar:

www.arduino.cc/en/Main/Software

← → ↻ Secure | <https://www.arduino.cc/en/Guide/HomePage>

BUY SOFTWARE PRODUCTS LEARNING FORUM

Code online on the Arduino Web Editor

To use the online IDE simply follow [these instructions](#). Remember that boards work out-of-the-box on the [Web Editor](#), no need to install anything.

Install the Arduino Desktop IDE

To get step-by-step instructions select one of the following link accordingly to your operating system.

- [Windows](#)
- [Mac OS X](#)
- [Linux](#)
- [Portable IDE](#) (Windows and Linux)

Choose your board in the list here on the right to learn how to get started with it and how to use it on the Desktop IDE.

Windows Taskbar: Ask me anything, File Explorer, Edge, Mail, Photos, Calculator, etc.

← → ↻ Secure | <https://www.arduino.cc/en/Guide/Windows>

BUY SOFTWARE PRODUCTS LEARNING FORUM SUPPORT BLOG

GETTING STARTED > Windows ENCLIPER

Install the Arduino Software (IDE) on Windows PCs

This document explains how to install the Arduino Software (IDE) on Windows machines:

- [Download the Arduino Software \(IDE\)](#)
- [Proceed with board specific instructions](#)

Download the Arduino Software (IDE)

Get the latest version from the [download page](#). You can choose between the installer (.exe) and the Zip packages. We suggest you use the first one that installs directly everything you need to use the Arduino Software (IDE), including the drivers. With the Zip package you need to install the drivers manually. The Zip file is also useful if you want to create a portable installation.

When the download finishes, proceed with the installation and please allow the driver installation process when you get a warning from the operating system.

For different operating system platforms, the way of using Arduino IDE is different. Please refer to the following links:

Windows User: <http://www.arduino.cc/en/Guide/Windows>

Mac OS User: <http://www.arduino.cc/en/Guide/MacOSX>

Linux User: <http://playground.arduino.cc/Learning/Linux>

For more detailed information about Arduino IDE, please refer to the following link:

<http://www.arduino.cc/en/Guide/HomePage>

STEP-2: Connect your Arduino Uno to your Computer:

Use the USB cable provided in the kit to connect the Arduino to one of your computer's USB inputs.



STEP-3: Install Drivers

Depending on your computer's operating system, you will need to follow specific instructions. Please go to the URLs below for specific instructions on how to install the drivers onto your Arduino Uno.

Windows Installation Process:

Go to the web address below to access the instructions for installations on a Windows-based computer.

<http://arduino.cc/en/Guide/Windows>

Macintosh OS X Installation Process:

Macs do not require you to install drivers. Enter the following URL if you have questions. Otherwise proceed to next page.

<http://arduino.cc/en/Guide/MacOSX>

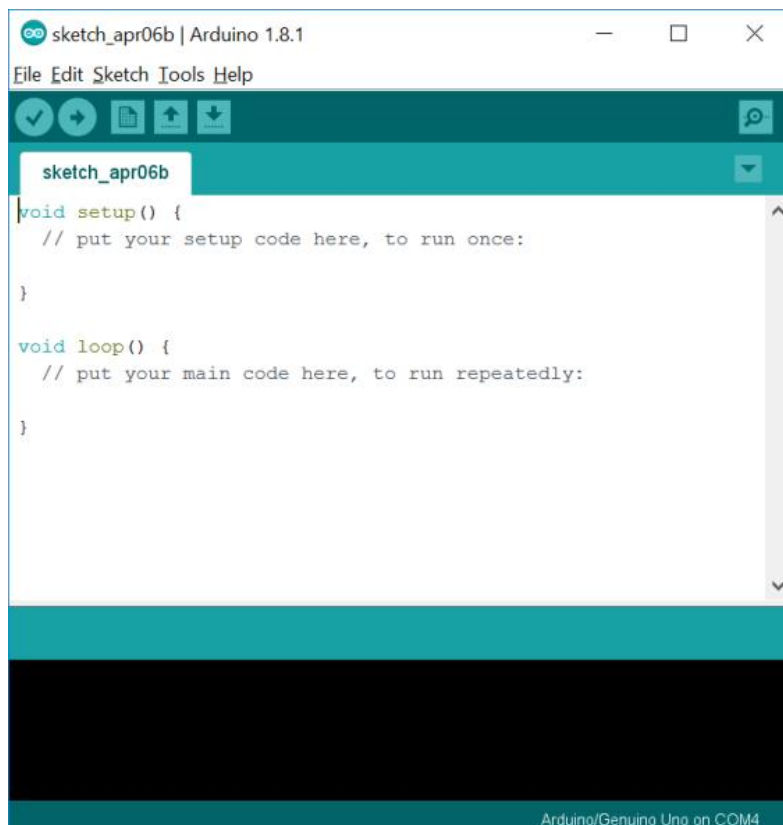
Linux:

32 bit / 64 bit, Installation Process Go to the web address below to access the instructions for installations on a Linux-based computer.

<http://www.arduino.cc/playground/Learning/Linux>

STEP-4: Open the Arduino IDE

Open the Arduino IDE software on your computer. Poke around and get to know the interface. We aren't going to code right away, this is just an introduction. The step is to set your IDE to identify your Arduino Uno.



GUI (Graphical User Interface)



Verify

Checks your code for errors compiling it.



Upload

Compiles your code and uploads it to the configured board. See [uploading](#) below for details.

Note: If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer"



New

Creates a new sketch.



Open

Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.

Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the File | Sketchbookmenu instead.



Save

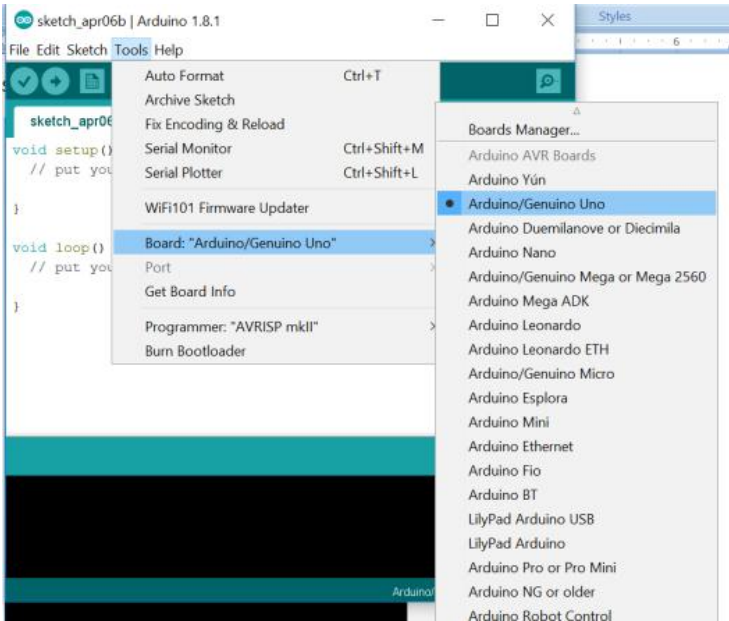
Saves your sketch.



Serial Monitor

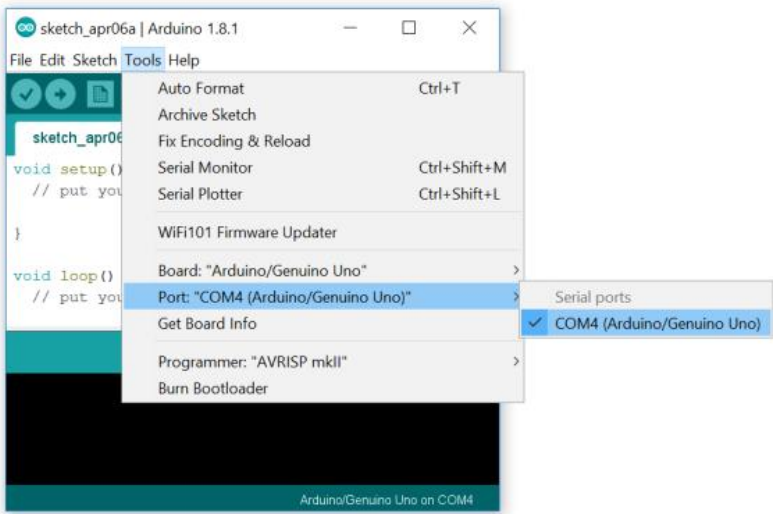
Opens the [serial monitor](#).

STEP-5: Select your board: Arduino Uno



STEP-6: Select your Serial Device

Windows: Select the serial device of the Arduino 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). To find out, you can disconnect your Arduino board and re-open the menu; the entry that disappears should be the Arduino board. Reconnect the board and select that serial port.

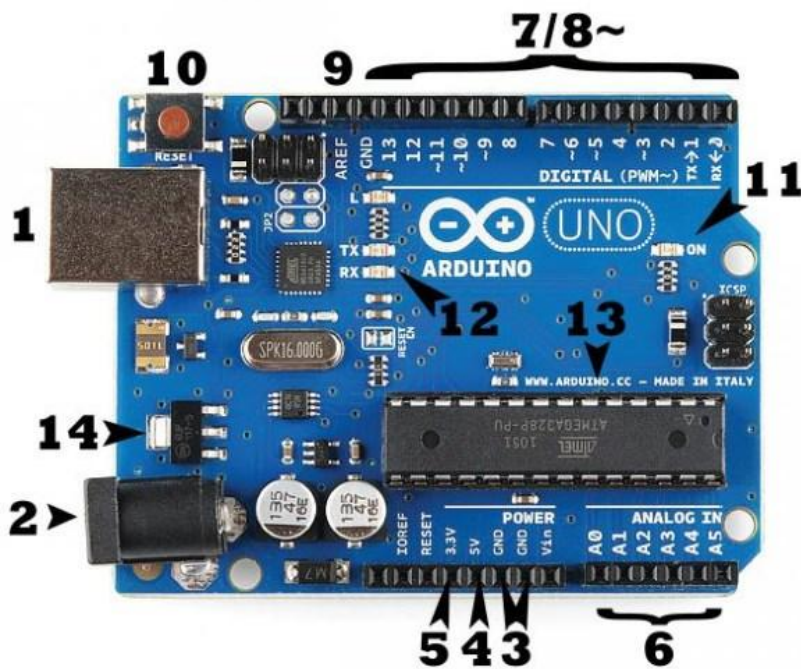


Mac OS: Select the serial device of the Arduino board from the Tools > Serial Port menu. On the Mac, this should be something with /dev/tty.usbmodem (for the Uno or Mega 2560) or /dev/tty.usbserial (for older boards) in it.

About Arduino Uno R3 board

What's on the board?

There are many varieties of Arduino boards that can be used for different purposes. Some boards look a bit different from the one below, but most Arduino have the majority of these components in common:



Power (USB / Barrel Jack)

Every Arduino board needs a way to be connected to a power source. The Arduino UNO can be powered from a USB cable coming from your computer or a wall power supply that is terminated in a barrel jack. In the picture above the USB connection is labeled **1** and the barrel jack is labeled **2**.

NOTE: Do NOT use a power supply greater than 20 Volts as you will overpower (and thereby destroy) your Arduino. The recommended voltage for most Arduino models is between 6 and 12 Volts.

Pins (5V, 3.3V, GND, Analog, Digital, PWM, AREF)

The pins on your Arduino are the places where you connect wires to construct a circuit (probably in conjunction with a breadboard and some wire). They usually have black plastic ‘headers’ that allow you to just plug a wire right into the board. The Arduino has several different kinds of pins, each of which is labeled on the board and used for different functions.

- **GND 3:** Short for ‘Ground’. There are several GND pins on the Arduino, any of which can be used to ground your circuit.
- **5V 4 & 3.3V 5:** As you might guess, the 5V pin supplies 5 volts of power, and the 3.3V pin supplies 3.3 volts of power. Most of the simple components used with the Arduino run happily off of 5 or 3.3 volts.
- **Analog 6:** The area of pins under the ‘Analog In’ label (A0 through A5 on the UNO) are Analog In pins. These pins can read the signal from an analog sensor (like a temperature sensor) and convert it into a digital value that we can read.
- **Digital 7:** Across from the analog pins are the digital pins (0 through 13 on the UNO). These pins can be used for both digital input (like telling if a button is pushed) and digital output (like powering an LED).
- **PWM 8:** You may have noticed the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11 on the UNO). These pins act as normal digital pins, but can also be used for something called Pulse-Width Modulation (PWM). We have a tutorial on PWM, but for

now, think of these pins as being able to simulate analog output (like fading an LED in and out).

- **AREF (9)**: Stands for Analog Reference. Most of the time you can leave this pin alone. It is sometimes used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

Reset Button

Just like the original Nintendo, the Arduino has a reset button (10). Pushing it will temporarily connect the reset pin to ground and restart any code that is loaded on the Arduino. This can be very useful if your code doesn't repeat, but you want to test it multiple times. Unlike the original Nintendo however, blowing on the Arduino doesn't usually fix any problems.

Power LED Indicator

Just beneath and to the right of the word "UNO" on your circuit board, there's a tiny LED next to the word 'ON' (11). This LED should light up whenever you plug your Arduino into a power source. If this light doesn't turn on, there's a good chance something is wrong. Time to re-check your circuit!

TX RX LEDs

TX is short for transmit, RX is short for receive. These markings appear quite a bit in electronics to indicate the pins responsible for serial communication. In our case, there are two places on the Arduino UNO where TX and RX appear – once by digital pins 0 and 1, and a second time next to the TX and RX indicator LEDs (12). These LEDs will give us some nice visual indications whenever our Arduino is receiving or transmitting data (like when we're loading a new program onto the board).

Main IC

The black thing with all the metal legs is an IC, or Integrated Circuit (13). Think of it as the brains of our Arduino. The main IC on the Arduino is slightly different from board type to board type, but is usually from the ATmega line of IC's from the ATMEL company. This can be important, as you may need to know the IC type (along with your board type) before loading up a new program from the Arduino software. This information can usually be found in writing on the top side of the IC. If you want to know more about the difference between various IC's, reading the datasheets is often a good idea.

Voltage Regulator

The voltage regulator (14) is not actually something you can (or should) interact with on the Arduino. But it is potentially useful to know that it is there and what it's for. The voltage regulator does exactly what it says – it controls the amount of voltage that is let into the Arduino board. Think of it as a kind of gatekeeper; it will turn away an extra voltage that might harm the circuit. Of course, it has its limits, so don't hook up your Arduino to anything greater than 20 volts.

Lesson 1 - Blinking LED

Overview:

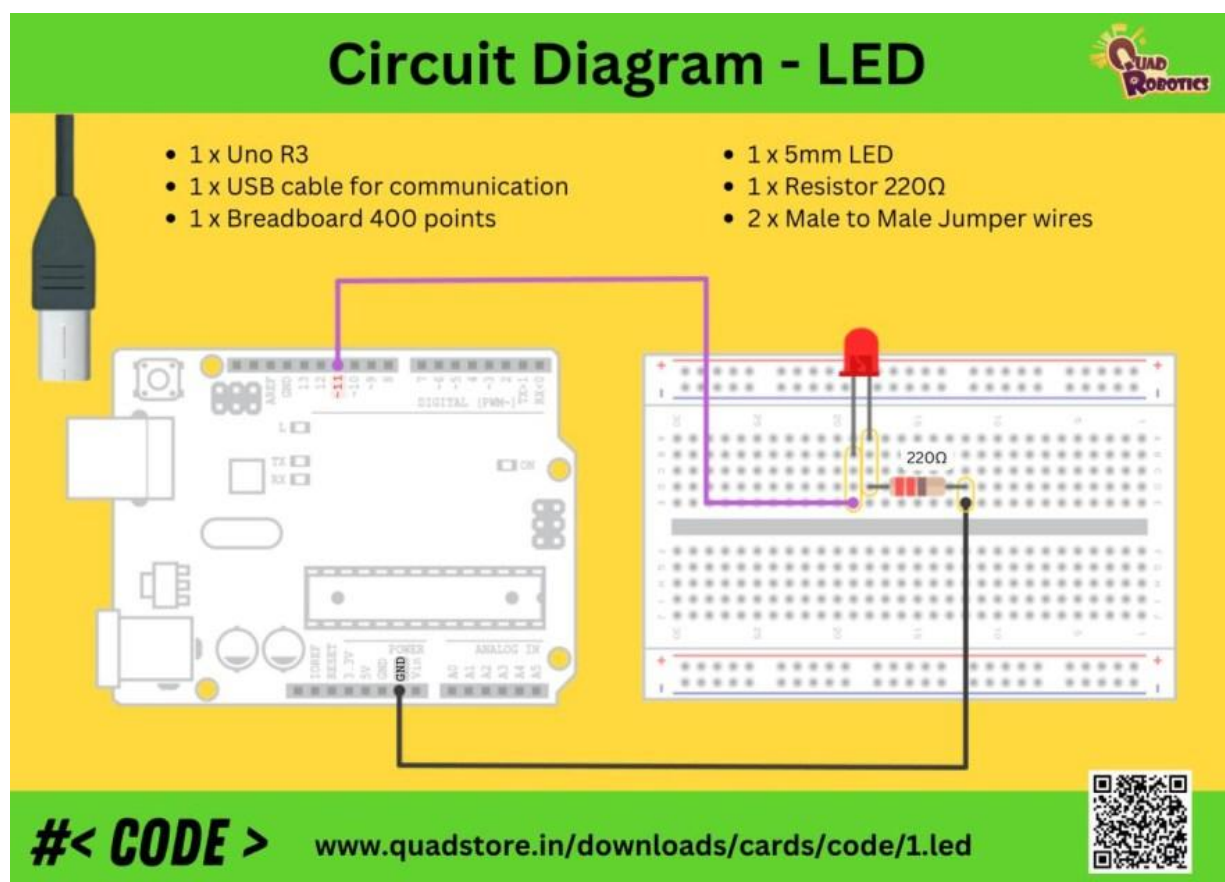
In project we will learn how to make an LED blink.

Components:

- 1 * UNO R3 board
- 1 * USB Cable
- 1 * 220Ω Resistor
- 1 * LED
- 1 * Breadboard
- 2 * Jumper Wires

Procedure:

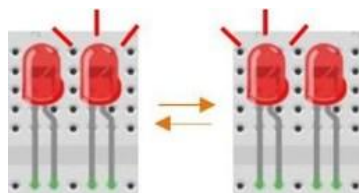
Step 1: Build the circuit using connections provided in the card that comes with the kit.



Step 2: Program: Open the program “1.led.ino” from the “CODE” Folder

Step 3: Compile the program and upload to Arduino UNO board

OUTPUT: You will see the LED blinking in regular intervals.



Lesson 2 – RGB LED

Overview

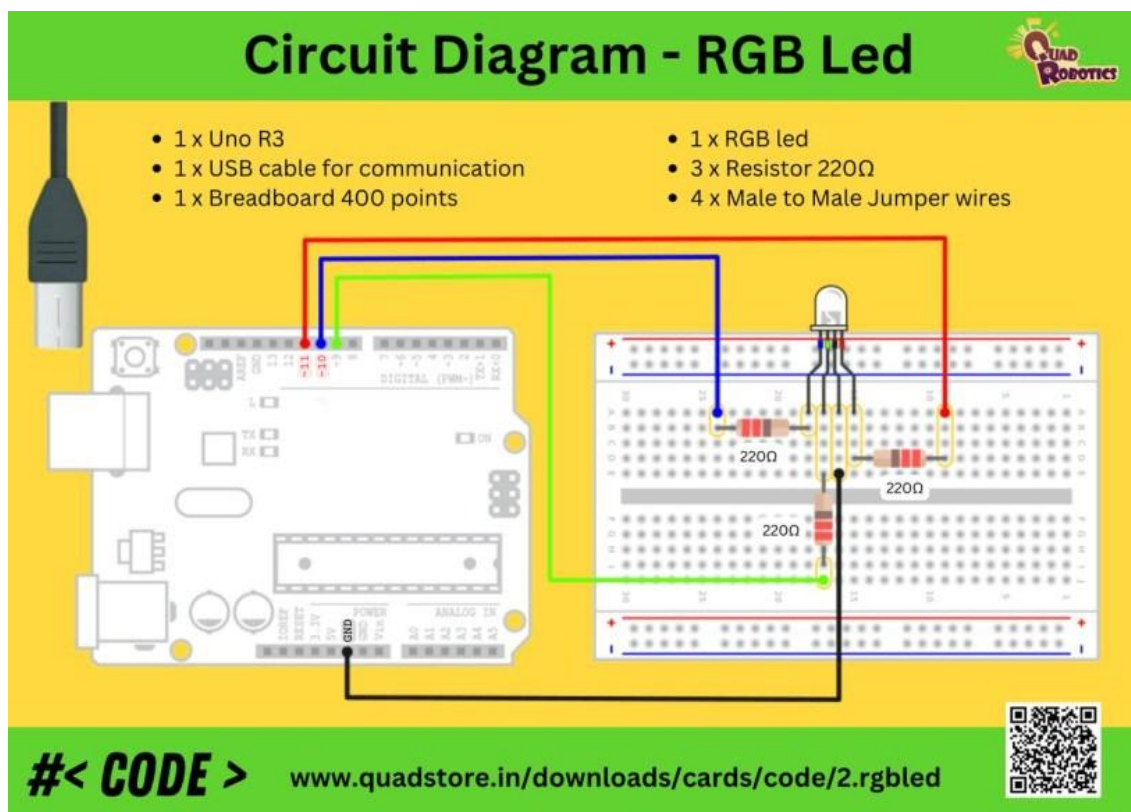
This project demonstrates how to control an RGB LED using an Arduino Uno R3, allowing you to create a variety of colors by adjusting the brightness of the red, green, and blue (RGB) components through pulse-width modulation (PWM).

Components

- 1 * Arduino UNO
- 1 * USB Cable
- 1 * RGB LED
- 3 * 220Ω Resistor
- 1 * Breadboard
- Several jumper wires

Procedure:

Step 1: Build the circuit using connections provided in the card that comes with the kit.



Step 2: Program: Open the program "2.rgbled.ino" from the "CODE" Folder

Step 3: Compile the program and upload to Arduino UNO board

OUTPUT: Observe the RGB LED changing colors according to the code logic.

Lesson 3 – Push Button Switch

Overview

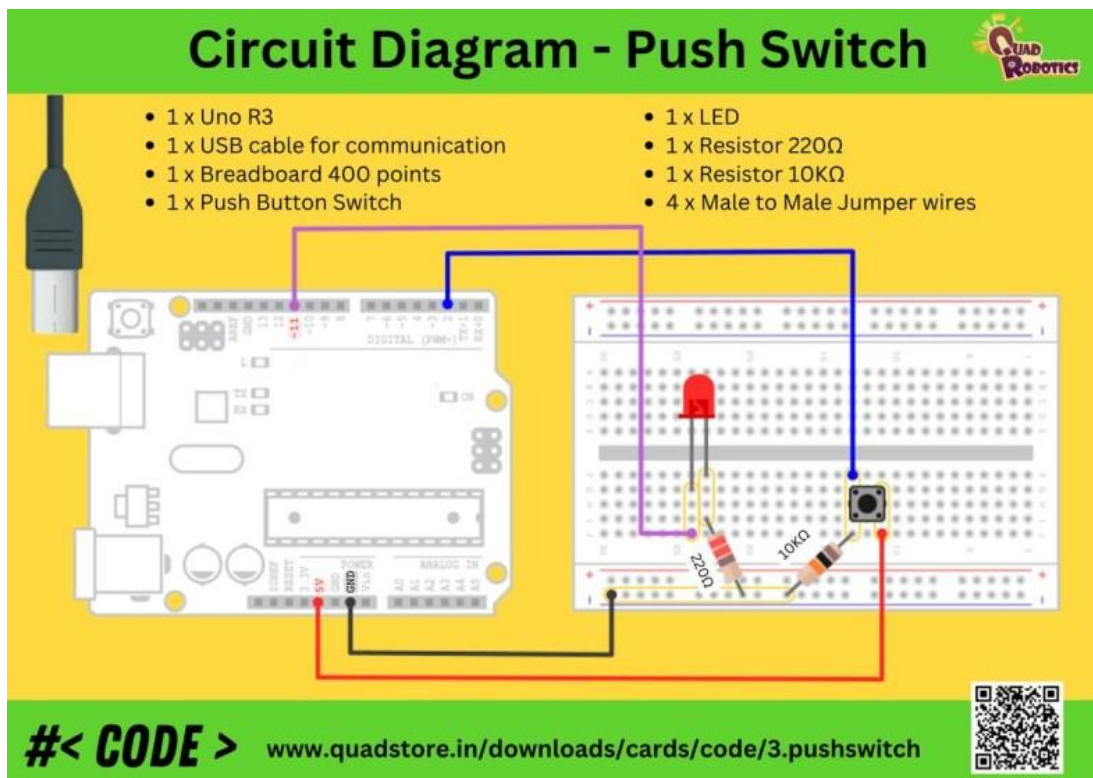
In this lesson, we will learn how to detect the state of a button, and then toggle the state of the LED based on the state of the button.

Components

- 1 * Arduino UNO
- 1 * USB Cable
- 1 * Push Button
- 1 * LED
- 1 * 10k Ω Resistor
- 1 * 220 Ω Resistor
- 1 * Breadboard
- Several jumper wires

Procedure:

Step 1: Build the circuit using connections provided in the card that comes with the kit.



Step 2: Program: Open the program “3.pushswitch.ino” from the “CODE” Folder

Step 3: Compile the program and upload to Arduino UNO board

OUTPUT: Now press the button, and you can see the state of the LED will be toggled between ON and OFF.

Lesson 4 – Photoresistor / Light Dependent Resistor (LDR)

Overview

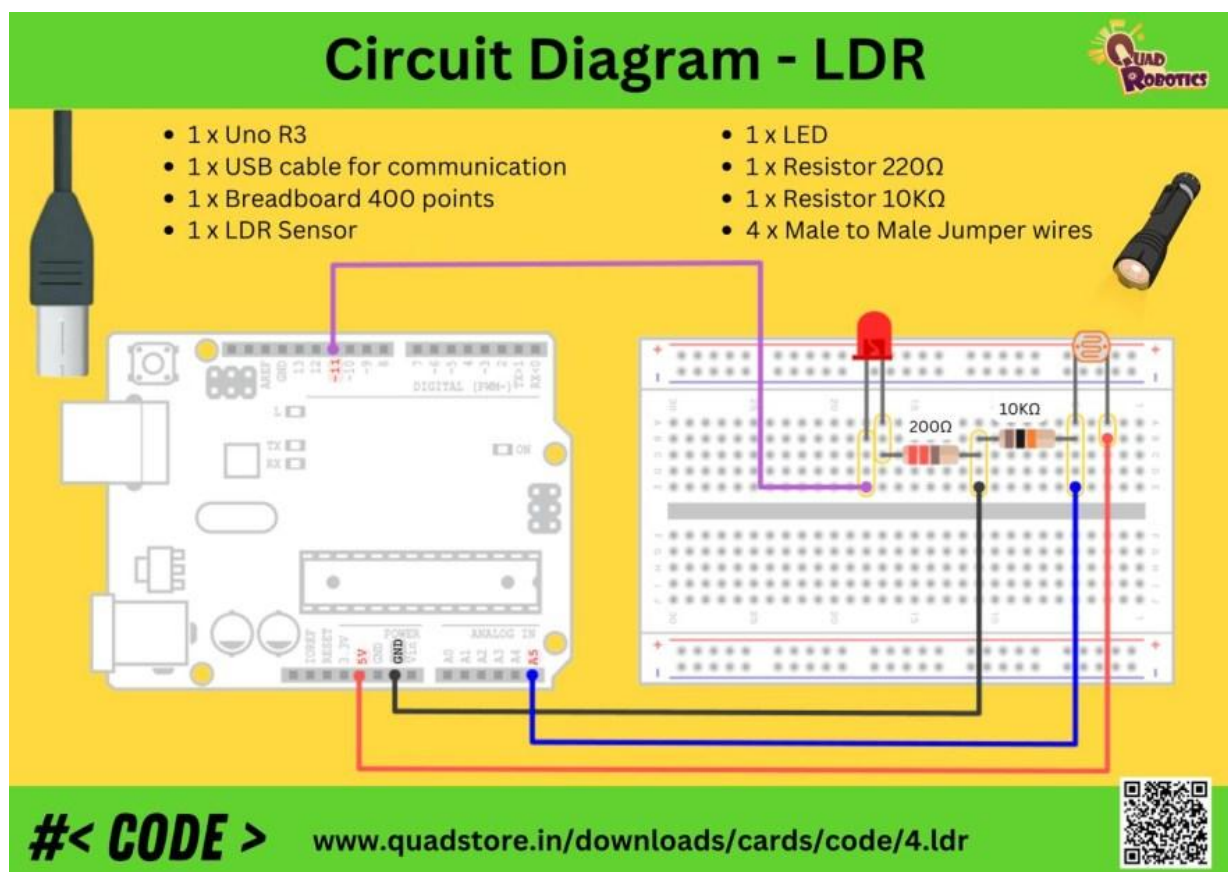
In this lesson, we will learn how to turn on the LED when the Light Dependent Resistor (LDR) detects light.

Components

- 1 * Arduino UNO
- 1 * USB Cable
- 1 * LDR Sensor
- 1 * LED
- 1 * 10k Ω Resistor
- 1 * 220 Ω Resistor
- 1 * Breadboard
- Several jumper wires

Procedure:

Step 1: Build the circuit using connections provided in the card that comes with the kit.



Step 2: Program: Open the program “4.ldr.ino” from the “CODE” Folder

Step 3: Compile the program and upload to Arduino UNO board

OUTPUT: Show some light in front of the LDR to make the LED turn on.

Lesson 5 – Buzzer

Overview

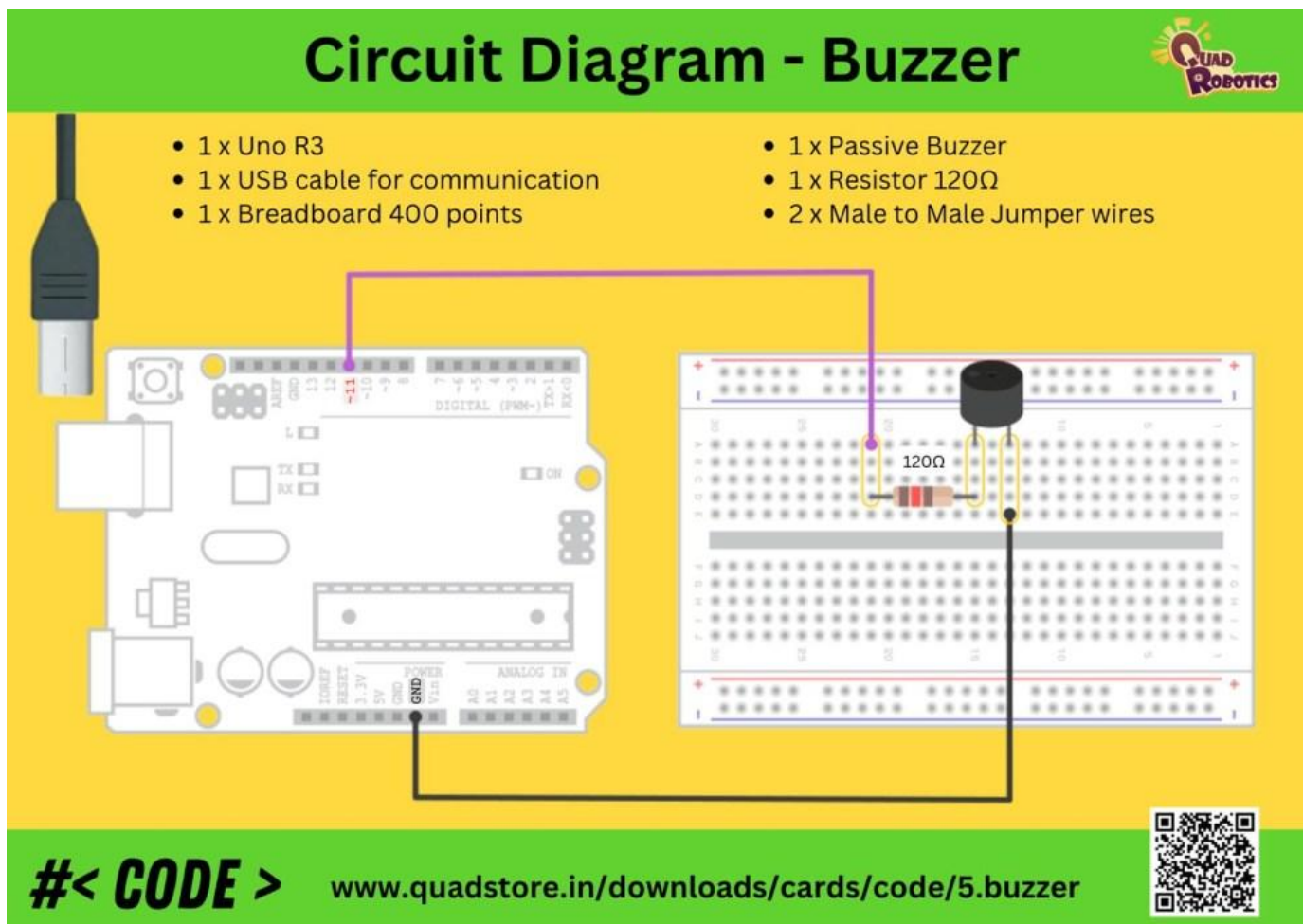
In this lesson, we will learn how to connect a buzzer to make a ring tone.

Components

- 1 * Arduino UNO
- 1 * USB Cable
- 1 * Buzzer
- 1 * 120k Ω Resistor
- 1 * Breadboard
- Several jumper wires

Procedure:

Step 1: Build the circuit using connections provided in the card that comes with the kit.



Step 2: Program: Open the program “5.buzzer.ino” from the “CODE” Folder

Step 3: Compile the program and upload to Arduino UNO board

OUTPUT: Buzzer should play some lovely ringtone.

Lesson 6: 7 Segment Display

Overview

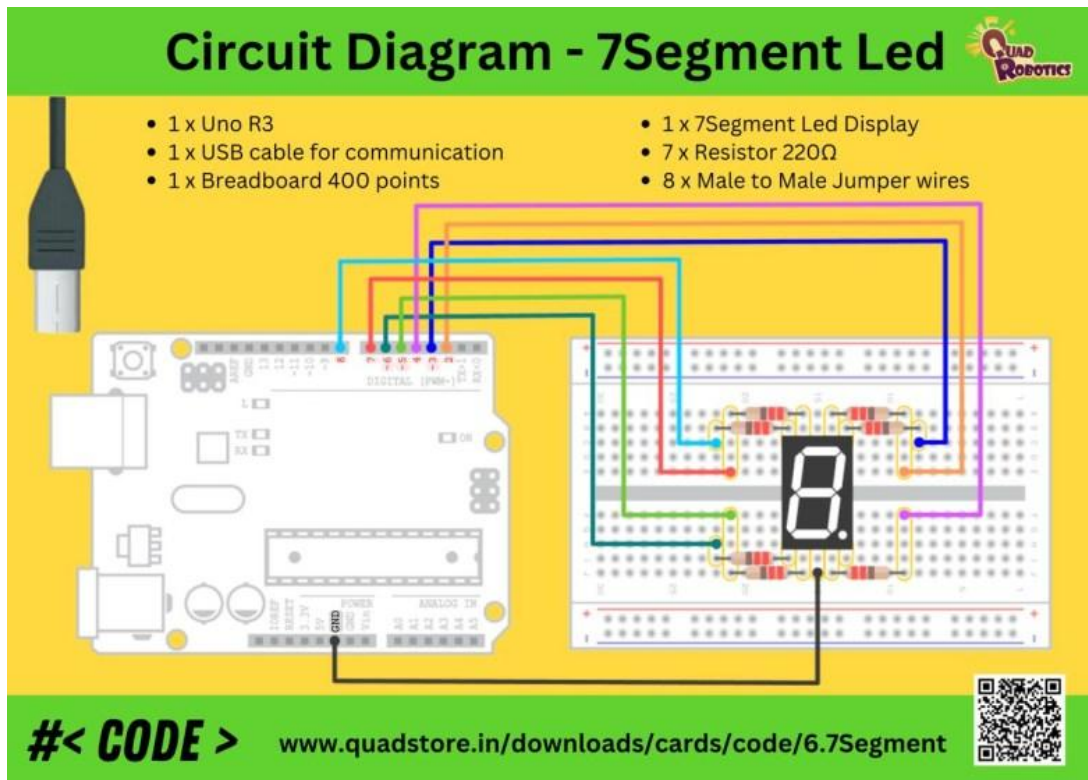
In this lesson, we will see how the 7 segment displays the counter from 9 to 0.

Components

- 1 * Arduino UNO
- 1 * USB Cable
- 1 * 7Segment Display
- 7 * 220k Ω Resistor
- 1 * Breadboard
- Several jumper wires

Procedure:

Step 1: Build the circuit using connections provided in the card that comes with the kit.



Step 2: Program: Open the program “6.7Segment.ino” from the “CODE” Folder

Step 3: Compile the program and upload to Arduino UNO board

OUTPUT: 7Segment display should display the counter from 9 to 0.

Lesson 7 – Potentiometer

Overview

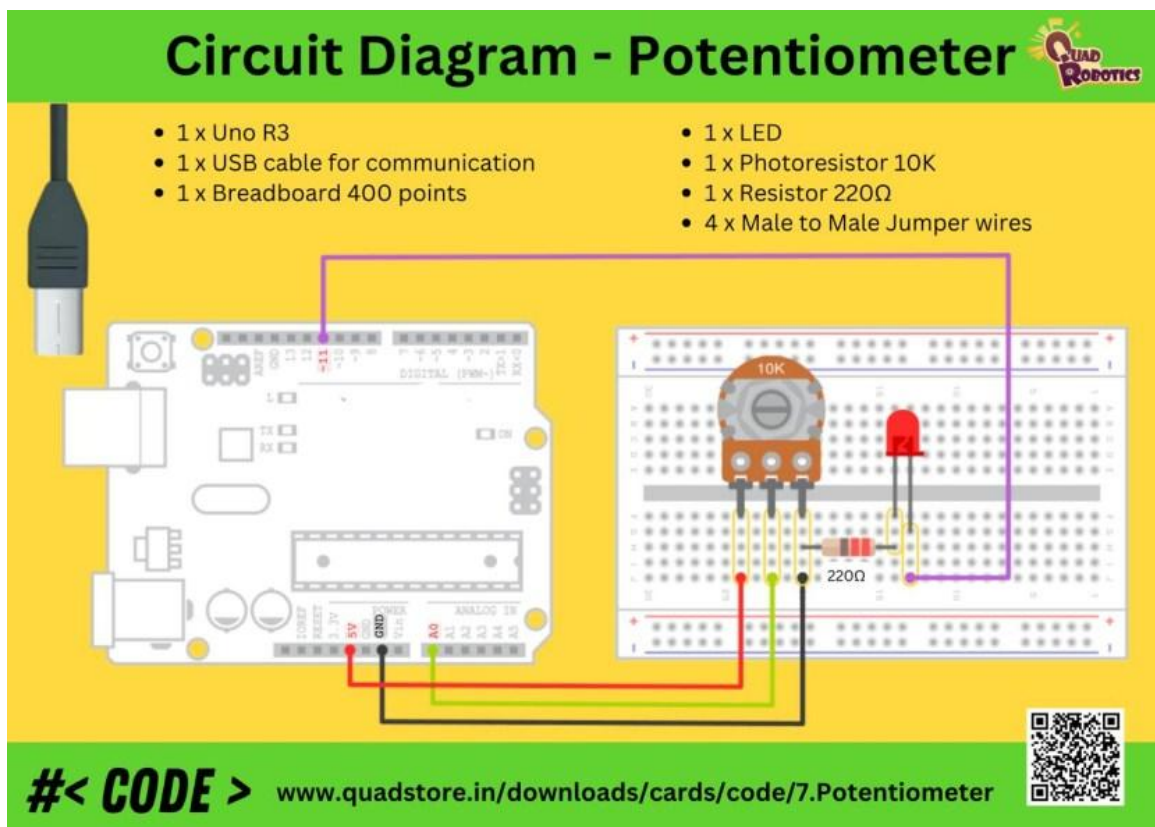
In this lesson, we will learn how to make the potentiometer increase, decrease or turn off the brightness of an LED.

Components

- 1 * Arduino UNO
- 1 * USB Cable
- 1 * LED
- 1 * 220k Ω Resistor
- 1 * Potentiometer
- 1 * Breadboard
- Several jumper wires

Procedure:

Step 1: Build the circuit using connections provided in the card that comes with the kit.



Step 2: Program: Open the program “7.Potentiometer.ino” from the “CODE” Folder

Step 3: Compile the program and upload to Arduino UNO board

OUTPUT: Turn the potentiometer to increase, decrease or turn off the brightness of an LED.

Lesson 8 – Tilt Sensor

Overview

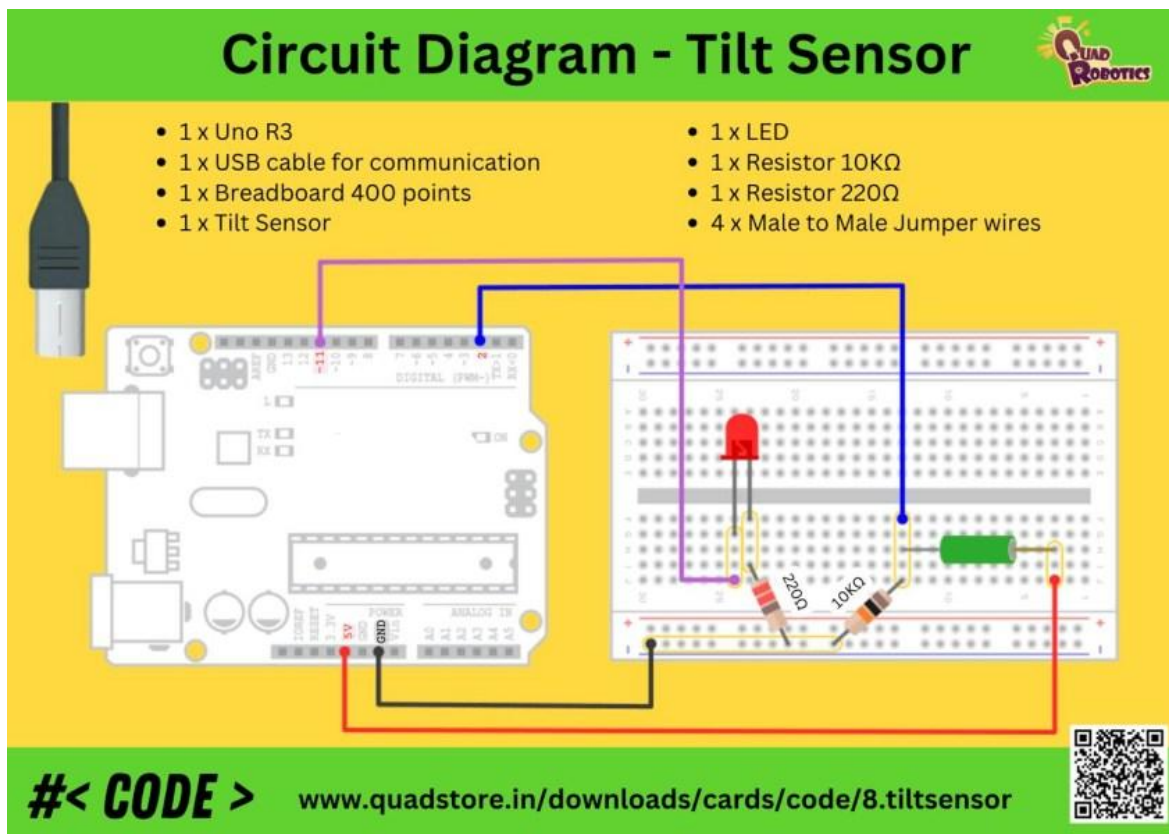
In this lesson, we will learn how to make use of the tilt sensor to turn on the LED when the sensor is tilted towards the gold terminal

Components

- 1 * Arduino UNO
- 1 * USB Cable
- 1 * LED
- 1 * 220k Ω Resistor
- 1 * Tilt Sensor
- 1 * Breadboard
- Several jumper wires

Procedure:

Step 1: Build the circuit using connections provided in the card that comes with the kit.



Step 2: Program: Open the program "8.tiltsensor.ino" from the "CODE" Folder

Step 3: Compile the program and upload to Arduino UNO board

OUTPUT: Tilt the sensor in opposite sides to turn on the LED on and off.

Lesson 9 – Servo Motor

Overview

In this lesson, we will make the servo take a 180° movement from one side to the other

Components

- 1 * Arduino UNO
- 1 * USB Cable
- 1 * Servo
- 1 * Breadboard
- Several jumper wires

Procedure:

Step 1: Build the circuit using connections provided in the card that comes with the kit.

Step 2: Program: Open the program “9.servo.ino” from the “CODE” Folder

Step 3: Compile the program and upload to Arduino UNO board

OUTPUT: Turn the potentiometer to increase, decrease or turn off the brightness of an LED.

