MPATROL 1

TUTORIAL DOCUMENT

Table of Contents

SAFETY INSTRUCTIONS 3

CHAPTER 1 – Introduction to MPatrol 1 4

CHAPTER 2 - Introduction to Computer Programming 6

CHAPTER 3 - Basic Electronics 14

CHAPTER 4 – Arduino Development Board 25

CHAPTER 5 – mBlock Software 29

CHAPTER 6 – Example Projects 52

Project 1 - Turn On the Red LED 53

Project 2 - Blink the Red LED 54

Project 3 - Blink two LEDs 55

Project 4 - Generate a Buzz using the Buzzer 56

Project 5 -Emergency Vehicle Siren 57

Project 6- On and Off a Bulb using a Switch 58

Project 7 - On and Off a Buzzer using a Switch 59

Project 8 - On and Off a Motor using a Switch 60

Project 9– Obstacle Detector using IR Sensor 61

Project 10– Turn on a LED using a Touch Switch 62

Project 11– Turn on LEDs using a Sound Switch 63

Project 12– Home Security System 64

Project 13– Electronic Organ 65

Project 14– Alarm Timer 67

Project 15– Automatic Night Bulb 68

Project 16– Light Dimmer 70

Project 17– Motor Speed Controller 72

Project 18– Electronics Distance Meter 74

Project 19– Temperature & Humidity Meter 77

Project 20– Water Level Meter 80

Project 21– Moisture Level Meter 82

Project 22– Communicating with a Computer 84

Project 23– Device Controlling Using Computer Commands 90

Project 24– Working with Mathematical Blocks 93

Project 25– Working with String Manipulation Blocks 96

SAFETY INSTRUCTIONS

To avoid any risks of personal injury, please do not close your hands to motor, while you are working with the motor

CHAPTER 1 – Introduction to MPatrol 1

The subject "STEM" is now getting very popular around the world. It is becoming a compulsory subject in most of the schools from Grade 3. What is "STEM" mean? It is learning about **S**cience, **T**echnology, **E**ngineering, and **M**athematics. Millions of students worldwide now learn "STEM", starting from the age of 8 – 9 years old, and get ready to step on the future high technology world from their very young age.The subject "STEM" will be the source of future engineers and scientists who is going to be shaping the world in the future.

Computer Programming (Coding), Electronics and Robotics are essential subject areas of "STEM" education. Because of that, schools worldwide now started to learn these subjects to their students, from the young age of Grade 3. MPatrol 1 Learning Kit is the first step for students interested in Computer Programming, Electronics, and Robotics. In MPatrol 1, students can write computer programs to talk with electronic components such as sensors, lights, motors and buzzers. So, they can learn computer programming while learning electronics and robotics.MPatrol 1 Learning Kit has designed based on the Arduino Uno board. Arduino Uno is the most famous coding and electronics learning tool among students around the world. So, students can master the Arduino platform also while experimenting with MPatrol 1.

MPatrol 1 Learning Kit consists of so many sensors such as temperature and humidity sensor, Infrared sensor, water sensor, light sensor, moisture sensor, sound sensor, motion sensor, and touch sensor. It also has an ultrasonic module, LCD module, motor, buzzer, 2 LEDs, push-button switch, and a variable resistor. Students can write computer programs to communicate with these electronic modules, such as reading data from sensors and display them on the LCD. So can learn both coding and electronics at the same time.

MPatrol 1 can program using the simple programming language called Scratch. Scratch is the starting programming language in most schools around the world. If you are familiar with the advanced programming language called "C," you can program the MPatrol 1 using the "C" programming language as well. You can use mBlock software for programming the MPatrol 1, using Scratch or "C" programming language. mBlock is an open-source software that you can download free and available for Windows and Mac operating systems.MPatrol 1 tutorial document consists of all the details of downloading and installing the mBlock software and how to write computer programs using it. If interested, anyone can use the famous "Arduino IDE" software to write programs for MPatrol 1. It is also an open-source software and free to download.

MPatrol 1 Learning Kit is coming with a comprehensive tutorial, which describes Computer Programming and Electronics basics. So, by using this kit and following the tutorial, students who new to computer programming and electronics also can learn computer programming and electronics on their own without getting help from anyone else. The tutorial has written in a very simple and easy to understand way. There aretwo separate chapters for Arudino boards and mBlock software. In the Arduino chapter, I have explained the Arduino boards and how to use them in detail. In the mBlock software chapter, I have described all

the details of downloading and installing the mBlock software, writing computer programs for communicating with electronic modules,uploading your program to MPatrol 1,adding electronic modules to mBlock software and troubleshooting.

At the end of the tutorial document, you can find 25 sample projects, which cover all the electronics modules in MPatrol 1 Learning Kit. I have explained all the background electronic knowledge need for that project in each project. For each project, I have shown the computer program and has explained it by line by line. So, by following the tutorial and doing these projects, you should get good knowledge about coding and electronics by yourself. These are just example projects that cover all the electronic modules in theMPatrol 1 Learning Kit. According to your creativity and innovation ability, you can do many new projects by combining the electronic modules in MPatrol 1. I recommend reading the first chapters before starting the tutorial document projects. Because they provide the necessary knowledge for doing those projects. If you are familiar with any of the chapter topics, you can skip them and go to the project chapter.

CHAPTER 2 - Introduction to Computer Programming

What is computer programming, and what is a computer language?

Just think of your day to day activities you are doing from morning to night. Just after getup, you brush your teeth and get a wash. After you dressed up, have your breakfast, and go to school. You do your science, mathematics, and play in the break in school. After school over, you are going to swimming or football practice. After coming home, you do your schoolwork, have dinner, and go to sleep. For doing some of these activities, you are getting instructions from your mother or teacher. But most of the activities like eating, playing, doing mathematics and singing, you do on your own without getting instructions from anyone. How you do like this? Because you have a brain, so you can think on your own and do things as you wish.

Let's compare a computer with you. A computer also can do a lot of activities the same as you. It can do mathematics, draw pictures, play songs, send emails to someone, write letters and numbers …etc. But there is one main difference between you and the computer. That is, the computer cannot think as you do. So for every activity, we have to give instructions to the computer.

So the next question that comes to your mind is how we can give instructions to the computer. Can it understand English? No, computers cannot understand the ordinary English language we are speaking. Let's assume you have a friend who recently migrated from China, and he cannot understand English. So if you want to talk with him, the only way is to speak in the Chinese language. Because that is the only language, your friend can understand. Similarly, if you want to give instructions to a computer, to do a specific task, you have to use a language, which computers can understand.

There are a lot of languages which computers can understand. They are not exactly similar to the English we are using to communicate with each other, but very similar to the English language. These languages are called computer programming languages. If we want to give instructions to a computer to do a specific task, we must provide instructions using a computer programming language. Giving instructions to a computer for doing a particular work is called computer programming. Computer Programming is also called coding. Scratch, C, C++, Python, and JAVA are some famous computer programming languages nowadays.

Scratch programming language

As I mentioned in the previous paragraph, there are many computer languages available for giving instructions to computers. In this tutorial, we are using the programming language called Scratch. Scratch is a computer programming language specially designed for children to learn computer programming interactively, with visual programming blocks. Schools worldwide are now starting to teach Scratch as a part of their curriculum to prepare children to learn computer programming. In Scratch, children can join labeled programming blocks

(which serve as code snippets) to write a complete computer program, which turns computer programming into a more visually exciting process.

Scratch programming instructions.

Let's learn some scratch computer instructions we will use in this tutorial. Here we cover the basic computer instructions in Scratch, and if need any additional instructions for a particular project, I will teach them under that project chapter. We are using a software tool called mBlock for all the programming tasks in this tutorial. So below Scratch programming instructions are for programming with the mBlock software tool. If you are using another software tool for programming with Scratch, these instructions may differ slightly.

I will teach the scratch instructions with real-world examples for easy understanding. First, I mention the English language instruction and then mention the scratch programming language instruction for that. All the scratch instructions in the mBlock software tool have been implemented as visual blocks. When programming with Scratch, we have to drag and drop the particular instruction block to programming space and set the parameters inside the visual block as we wish. **E.g.,** If we want to have some wait time in our program, first, we have to drag and drop the **"wait" visual instruction block** into the programming space. (as shown below) Then have to set the waiting time inside the box as we wish. (2 seconds in below wait instruction)

Please note that we use mBlock software for programming with Scratch Programming Language in this tutorial. The Scratch Programming Instructions, which we explain below, could be slightly different in another Scratch Programming Software.

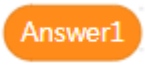
What is a Variable?

Before jump to learn scratch programming instructions, we will learn what is meant by the term **"Variable"?** Variable is a fundamental term in computer programming. If your teacher asks you to add 5 to 12, you will give the answer as 17. Then the teacher asks you again to add another 8 to that answer. So you will give the final answer as 25 (17 + 8). Let's see how you actually found the final answer. When the second time teacher asks you to add another 8 to previous answer (that is 17), the previous answer was in your brain. If you forgot the answer of the first calculation (12 + 5 = 17), you could not do the second calculation. (17 + 8 = 25). Luckily because you are a human, the first calculation answer was on your brain, so you could do the second calculation by using the first calculation's answer.

What could happen if we say computer to do the same calculation? Does the computer have a brain to memorize the answer of 5 + 12? (That is 17)? No. But the computer has an

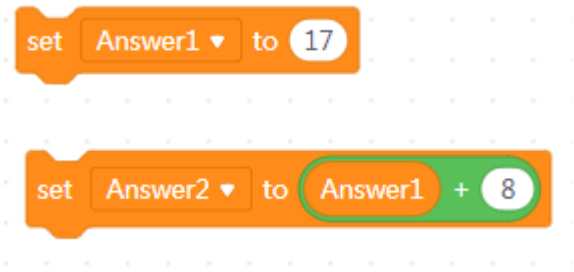
electronic memory, to remember values. As I mentioned earlier, computers cannot do anything, by itself and for every computer task, we have to give computer instructions. The computer cannot memorize the answer of the first calculation (17) by itself, and we have to provide instructions to the computer to save that answer (17) in the computer's memory. The computer can then use this answer and do the second calculation as you did. That mean computer can calculate 17 + 8 = 25 because the figure of "17" is now in the computer's memory.

So the next question is how we can give instructions to the computer for memorizing the value of "17"? Here we need a VARIABLE for this task. **VARIABLE is a small section of computer electronic memory.** We can define variables in our computer programs. Different computer programs define variables differently. Scratch programming language defines a variable as below.



Now you have defined a variable in your computer program, and you have given a name for it as **"Answer1".** So this **"Answer1"** represents a small section in your computer's electronic memory. If you assign any value to "Answer1", the computer can memorize this value, because Answer1 is part of the computer's electronic memory. In computer terminology, assigning a value to a variable (in our case, assign the figure of "17" to Variable name "Answer1"), is called saving that value (17) in computer memory. Then the computer can use this value for any further operation, because it has memorized this value. In our example, the computer can do the second calculation (that is 17 + 8 = 25). If we save the final answer (that is 25), in another variable, it also can use for any further calculation. (Think of a situation you want the computer to subtract nine from the final answer).

Now let's see how we assign a value to a variable in Scratch.



In Scratch, we can use "**set [VARIABLE] to [FIGURE]"** block for assigning a value to a variable. Here we have created two variables called "Answer1" and "Answer2". In the first block, we assign figure "17" to "Answer1" variable. In the second block, we assign the value of "Answer1 + 8" to "Answer2" variable. (That is Answer2 = 17 + 8). Since Answer1 and Answer2 are variables in computer's electronic memory, we can use the values save in those variables (Answer1 = 17, Answer2 = 25) for any further calculation. You can give any name to a variable, and you can define any number of variables as you want in a computer program. (Depend on your computer electronic memory size)**.** In the below examples

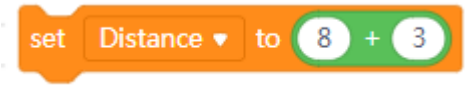
under Scratch Instructions, I have used a variable called “Distance” to save the values in computer’s electronic memory.

Basic Scratch Instructions

(1)ADDITION Instruction – Use to add a number to another number and store it in a variable. (Here, Variable = Distance)

Instruction in English – DISTANCE = 8 + 3

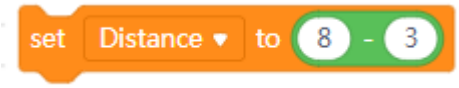
Instruction in Scratch



(2)SUBTRACTION Instruction – Use to subtract a number from another number and store it in a variable. (Here, Variable = Distance)

Instruction in English – DISTANCE = 8 - 3

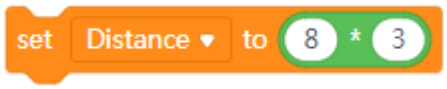
Instruction in Scratch



(3)MULTIPLICATION Instruction – Use multiplying a number from another number and storing it in a variable. (Here, Variable = Distance)

Instruction in English – DISTANCE = 8 * 3

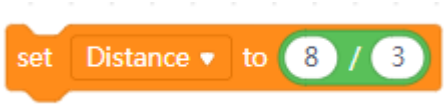
Instruction in Scratch



(4)DIVIDER Instruction – Use to divide a number from another number and store it in a variable. (Here, Variable = Distance)

Instruction in English – DISTANCE = 8 / 3

Instruction in Scratch



(5) GREATER THAN Instruction- Use this instruction to check whether a particular variable value is greater than a specific figure. In the below block, the computer checks whether the DISTANCE value is greater than 50.

Instruction in English – If Distance greater than 500?

Instruction in Scratch

(6) LESS THAN Instruction - Use this instruction to check whether a particular variable value is less than a specific figure. In the below block, the computer checks whether the DISTANCE value is less than 50.

Instruction in English – If Distance less than 500?

Instruction in Scratch

(7) EQUAL Instruction – Use this instruction to check whether a particular variable value is equal to a specific figure. In the below block, the computer checks whether DISTANCE value is equal to 50.

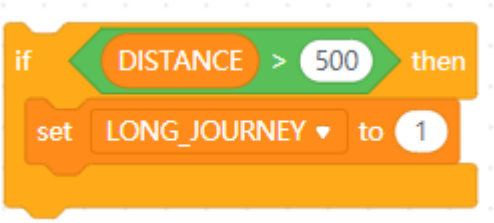
Instruction in English – If Distance equal 500?

Instruction in Scratch

(8)IF THEN Instruction - Use for do some task, according to a specific condition. In the below block, **condition is "DISTANCE > 500".** So if ONLY the DISTANCE value is greater than 500, computer does the task. That is set the LONG_JOURNEY variable value as 1. If this condition is not satisfied, computer is doing nothing.

Instruction in English–If the distance is greater than 500m, mark that journey as a long journey.

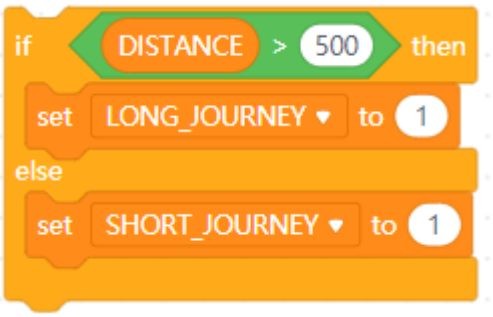
Instruction in Scratch



(9)IF THEN ELSE Instruction - Use for switch between two tasks, according to a specific condition. In the below block, the **condition is "DISTANCE > 500".** So if the DISTANCE value is greater than 500, the computer does the first task. That is set the LONG_JOURNEY variable value as 1. If the DISTANCE value is not greater than 500, **(ELSE condition)** (that is less than or equal 500), computer does the second task. That is set the SHORT_JOURNEY variable value as 1.

Instruction in English – If the distance is greater than 500m, mark that journey as a long journey. If the distance is less than 500m, mark that journey as a short journey.

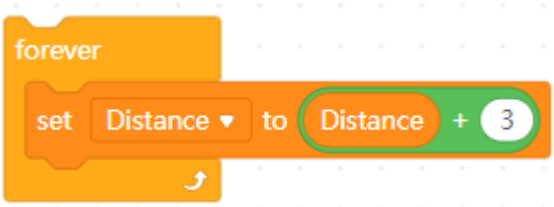
Instruction in Scratch



(10)FOREVER Instruction - Do some task or few tasks continuously, until exit from the software. So tasks, which we implement inside this block, run continuously, until we close the programming software. if we program an electronic kit, jobs inside this **"forever"** block run continuously until off the power.

Instruction in English – Continuously add 3 to the distance value.

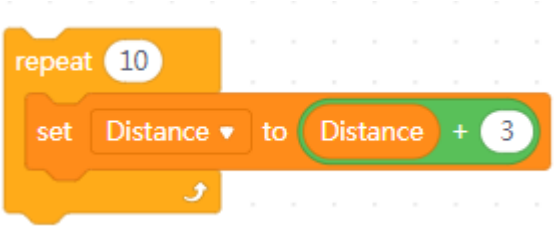
Instruction in Scratch



(11)REPEAT Instruction - Here, we can repeat the tasks, which implement inside the **"repeat**" block for a given number of times. We can input the repeat count as we want.

Instruction in English – 10 times adds 3 to the distance value.

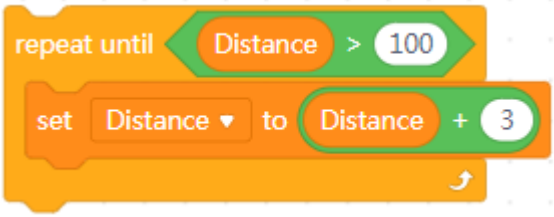
Instruction in Scratch



(12)REPEAT UNTIL Instruction - Here, we repeat the tasks, which implement inside the **"repeat until**" block, until a condition is satisfied. The computer keeps adding 3 to distance variable in the below example until distance value greater than 100. Once distance value becomes greater than 100, computer stops the addition.

Instruction in English – Add 3 to the distance value continuously until the distance value greater than 100.

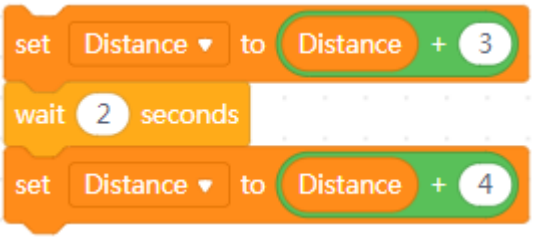
Instruction in Scratch



(13)WAIT Instruction - We use "wait" instruction to wait some time before doing the next operation. Usually computer programs run the instructions one by one without any delay between two nearby instructions. So if we need the computer to wait some time between two instructions, we can use this **"wait"** instruction. We can enter the waiting time as we wish.

Instruction in English – Add 3 to the distance value. Then wait 2 seconds. After the 2 second waiting time, add another 4 to the distance value.

Instruction in Scratch

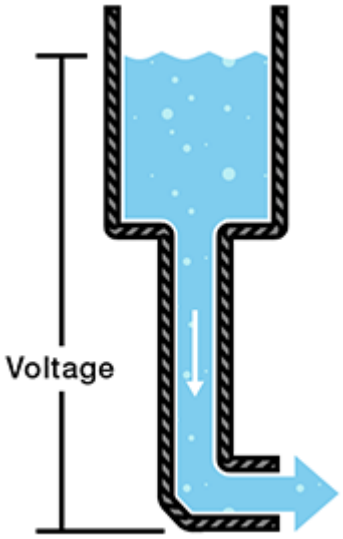


CHAPTER 3 - Basic Electronics

In this chapter, we will learn about the basics of Electronics. Electronics is a significant element in the STEM education journey. If you like to make real word applications and creative products such like Robots, Lighting Circuits, Alarm Circuits, Home Security Systems, Smart Home Applications..etc, you need to have a knowledge about Electronics as well as Programming. Now we will learn about some basic concepts of Electronics, and some commonly use Electronic Components.

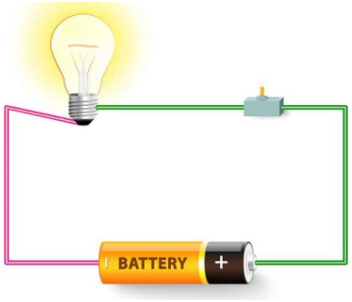
Basic Concepts in Electronics

1. What are Electric Charges, Voltage and Current?



Think about a water tank, which has an outlet at the tank's bottom. What happens when increasing the water level in the tank? The water flowing speed from the outlet also increasing, because of the increasing water pressure at the bottom of the tank. If you set a rotating wheel like a turbine at the tank's outlet, it will rotate faster, with the water level of the tank increasing. That means, in other words, the turbine will rotate more faster when increasing the water pressure.

Here voltage is like the Water Pressure in the tank. Electric Charges are like water in the tank, and current is like Water Flow Rate in the tank's outlet.



Like the water in this tank, we have Electric Charges in electronic components. If we want to do some task using an electronic component (such as turn on a bulb), we should create an electrical charge flow across that component. For example, if you want to turn on a bulb in the above picture, you have to create an electrical charge flow across the bulb. Faster the speed of electrical charges, more light generate on the bulb. Faster the speed of electrical charges means higher the current. Because the current is the rate of flowing electrical charges. Higher the current, we can see more light on the bulb.

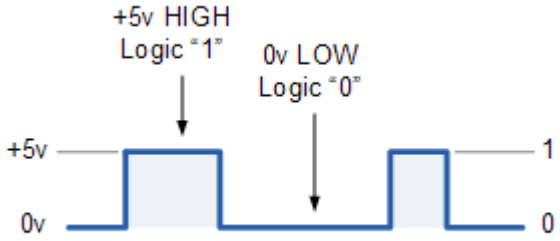
How can we increase the speed of the electric charges? As water flow rate increase when increasing the water pressure in the tank, if we can increase the electric charge pressure, we can increase the speed of the charges. That means we can increase the current. What is mean by increasing the electric charge pressure? It is called increasing the voltage. As the same as water pressure, we can say, voltage is the pressure of electric charges. More voltage across the bulb means more current flow through the bulb and more light on the bulb.

Who creates the Voltage or Charge pressure in the above Bulb Turn On project? It is doing by the battery, which connects across the bulb. The battery creates a Voltage or Charge pressure, which leads to flow a current across the bulb and battery. If we connect two batteries together, we can generate more electric charge pressure. That means two connected batteries give a higher voltage than one battery. So with two batteries, higher current flow through the bulb, and bulb becomes more bright.

What happens if you cut the wire in-between the battery and bulb? It is the same as if you cut the water tank's outlet line. If you cut the outlet line, the turbine will not get water, so it can not rotate. Because there is no water pressure at the end of the outlet. The same thing applies to the bulb. If you cut the wire, the bulb is not getting the charge pressure, so there is no electric charge flow inside the bulb. That means no current flowing inside the bulb, and the bulb is getting off.

The measuring Unit for Voltage is Volt, denoted by letter “V”. As an example, battery in the above picture gives 1.5V. If we connect 2 batteries together, we can have 1.5V + 1.5V = 3V. Measuring Unit for Current is Ampere, denoted by letter “I”. 1 A is sufficiently a large amount of current. Unlike your household items like refrigerators, Fans, Heaters, etc.., the current flowing in our electronic projects are small. So most of the time, we often use milliampere as the current unit, denoted by mA. (1 A = 1000 mA). As an example, we can say 100mA current flowing through the bulb in above picture.

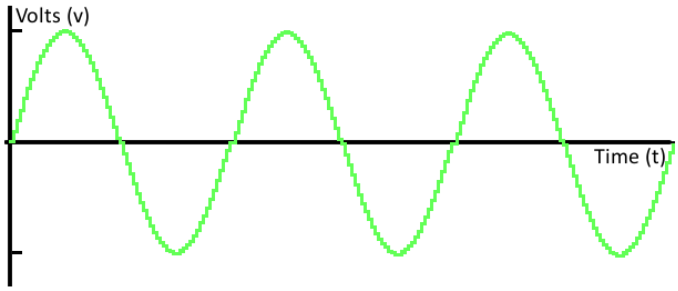
2. What is Digital Voltage?



Even you do not know what is mean by "Digital", you should have heard this word many times. You may have heard that your iPad is a digital device, or you may have heard that your computer is a digital device. So what is this "Digital" means?

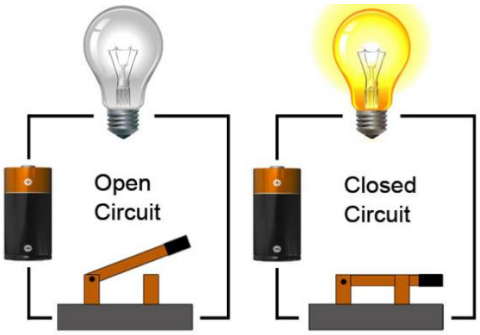
Digital means it has only two levels. So Digital Voltage means it has only Two Voltage Levels. In most cases, these two voltage levels are 5V and 0V. In electronics terminology, for the higher voltage value, we called HIGH Level or Logic "1". For the lower voltage value, we called LOW Level or Logic "0". So 5V = HIGH Level or Logic "1" and 0V = LOW Level or Logic "0". So now, what is the meaning of "your computer is a digital device"? That means inside your computer, electronic circuits are working with only two voltage levels. In the electronic circuit level, computer is doing all the tasks, by using these two voltage levels. You can learn more about Digital Voltage, when you are doing the projects with this learning kit.

3. What is Analog Voltage?



Not like Digital Voltage, Analog Voltage has many different Voltage Levels. It can clearly see from the above picture. Any Analog Voltage has a maximum and minimum voltage value. So the value of the analog signal at any particular time can be any value between its maximum and minimum value. For example, if an electronic component outputs a 0 – 5 V Analog Voltage, It can have any values like 0V, 0.1V, 0.2V, 1V, 2.5V, 3.2V, 4V, 4.5V, 5V, etc., between 0 and 5.

4. What is an Electronic Circuit?



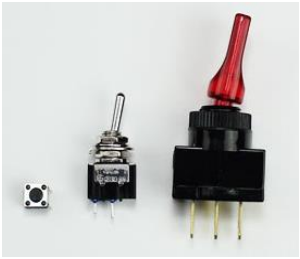
An electronic circuit is a circular path of conductors by which electric current can flow. A closed-circuit is like a circle because it starts and ends at the same point forming a complete loop. Furthermore, a closed-circuit allows electric current to flow uninterrupted around the circular path.

In contrast, if there is any break in electric current flow, this is known as an open circuit. As shown above, a switch in a circuit can cause it to be open or closed, depending on its position.

All circuits need to have three basic elements. These elements are a voltage source, conductive path and a load. The voltage source, such as a battery, is needed in order to cause the current to flow through the circuit. In addition, conductive path need to provides a route for the electric current to flow. Finally, circuit needs a load that consumes the power. The load in the above circuit is the light bulb.

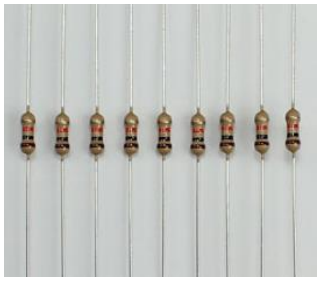
Basic Electronic Components in Electronic Circuits

1. Switch



Switches can come in many forms, such as pushbutton, toggle, rocker, momentary, etc. Their primary function is to interrupt electric current by turning a circuit on or off.

2. Resistor



Resistors are used to resist the flow of current or to control the voltage in a circuit. The amount of resistance that a resistor offers is measured in Ohms. More you want to limit the current in a circuit, you have to use a higher Ohm value resistor. Most resistors have colored stripes on the outside and this code will tell you it's value of resistance Ohms. You can use a Resistor Color Code Table or Online Resistor Color Code Calculator to determine a resistor's value.

3. Variable Resistor (Potentiometer)

A variable resistor is also known as a potentiometer. These components can be found in devices such as light dimmers or speed controllers. When you turn the shaft of a potentiometer, the circuit's resistance changes. By turning the shaft of a Variable Resistor or a Potentiometer, we can generate a Variable Voltage (Analog Voltage) or Variable Current. So we can use this Variable Voltage or Current to control some devices. Variable Resistors are coming with many different shapes as shown in the picture.

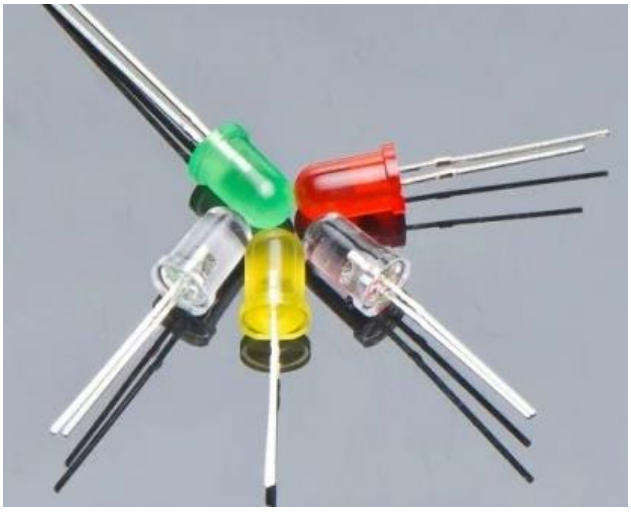
4. Light-Dependent Resistor (LDR)

A light-dependent resistor is also a variable resistor. Its resistance automatically changes according to the intensity of the light. So this device can generate a variable analog voltage, according to the intensity of the light and using this analog voltage, we can make electronic devices such like dark switching bulbs.

5. Diode

A diode allows electric current to flow in one direction and blocks it from flowing the opposite way. The diode's primary role is to route electric current from taking an unwanted path within the circuit.

6. Light Emitting Diode (LED)



LED is a very famous component among electronic hobbyists. Light-emitting diode is like a standard diode because electrical current only flows in one direction. The main difference is an LED will emit light when current flows through it. Inside an LED there is an anode and cathode. Current always flows from the anode (+) to the cathode (-) and never in the opposite direction. The longer leg of the LED is the positive (anode) side. Intensity of the light increase with higher the current.

7. Buzzer

A buzzer is a device that makes a buzzing or beeping noise. We can use buzzers in our electronic projects to generate a sound, such as in a home security alarm system or for making a siren horn. There are different kinds of buzzers in the market and the most common one is piezoelectric buzzer. When we apply a voltage across the piezoelectric buzzer, its piezoelectric plate starts to vibrate and this generates the sound. The higher the voltage applied, higher the vibration and generates more sound.

8. Electric Motor

This is a very famous component among kids. If you have a battery-operated toy car or any other toy vehicle, you should have this electric motor inside your toy. Electric motor converts the electrical energy into mechanical energy. When applied a voltage across the motor, we can see the motor shaft is rotating. If we couple a wheel to this motor shaft, wheel starts to rotate. That is how your toy car is moving when you turn on the switch. Higher the current flowing through the motor, higher the motor rotating speed.

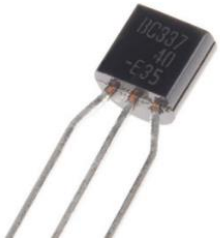
9. Sensors

This is another essential component in electronic circuits. Sensor can measure a physical quantity in our environment such like temperature, pressure, humidity, moisture level and convert it into an electrical voltage or current. Because voltage and current are the only languages that our electronic circuits can understand. In this way, our electronic circuits can measure the physical quantities such as temperature and humidity and display them for us to see. There are many sensors available in the market for measuring different physical quantities such as temperature, humidity, pressure, moisture, water etc.. With this Training Kit, we will do a lot of interesting projects with different kinds of sensors. So when you are doing the projects with this Training Kit, you can get a good understand of the sensors.

10.Capacitor

Capacitors store electric charge and then discharges it back into the circuit when there is a voltage drop. A capacitor is like a rechargeable battery and can be charged and then discharged. The value is measured in F (Farad), nano Farad (nF) or pico Farad (pF) range. As shown in the picture, they come with different packages and very common in electronic circuits.

11.Transistor

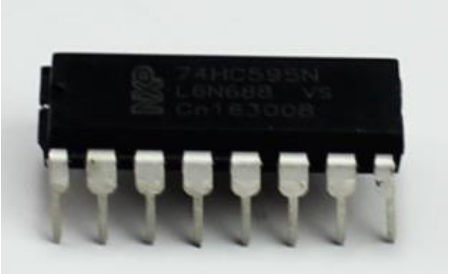


Transistors are tiny switches that turn a current on or off when triggered by an electric signal. In addition to being a switch, it can also amplify electronic signals.

12.Relay

A relay is an electrically operated switch that opens or closes when power is applied. Inside a relay is an electromagnet which controls a mechanical switch

13.Integrated Circuit (IC)



An integrated circuit is a circuit that has been reduced in size to fit inside a tiny chip. This circuit contains electronic components like resistors, capacitors and transistors but in a very smaller scale. Integrated circuits come in different variations such as 555 timers, voltage regulators, microcontrollers and many more. Each pin on an IC is unique in terms of it's function.

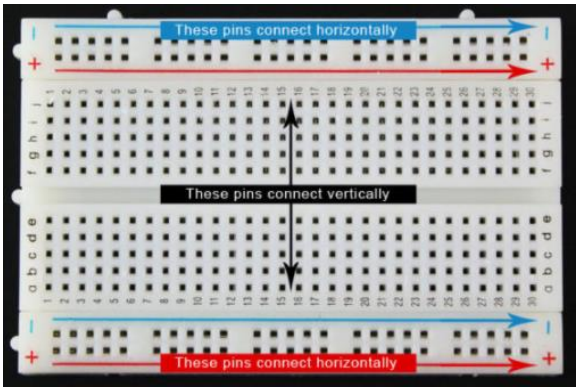
Tools for Making Electronic Circuits

1. Soldering Iron

When it is time to create a permanent circuit, you'll want to solder the parts together. To do this, a soldering iron is the tool you would use. Of course, a soldering iron isn't any good

unless you have solder to go with it. You can choose leaded or lead-free solder in a few diameters.

2. Breadboard



Breadboards are an essential tool for prototyping and building temporary circuits. These boards contain holes for inserting wires and components. Because of their temporary nature, they allow you to create circuits without soldering. Some holes in a breadboard have connected horizontally and other holes have connected vertically, as shown above.

3. Multimeter

A multimeter is a device that can be used to measure electric Current (Amps), Voltage (Volts) and Resistance (ohms). It's a great tool for troubleshooting circuits.

4. Test Leads (Alligator Clips)

Test leads are great for connecting components together to test a circuit without the need for soldering.

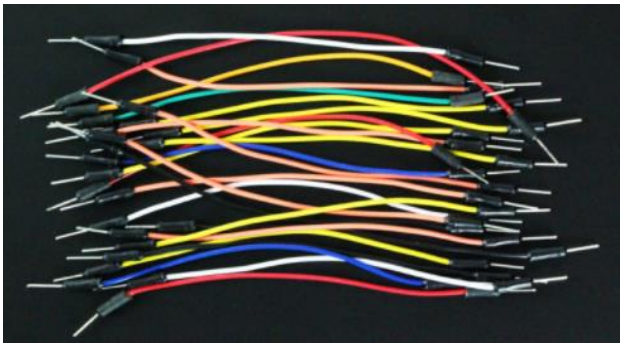
5. Wire Cutter

Wire cutters are essential for stripping stranded and solid copper wires.

6. Helping 3rd Hand

When working with electronics, it seems you never have enough hands to hold everything. This is where the helping hand (3rd hand) comes in. Great for holding circuit boards or wire when soldering or tinning.

7. Jumper Wires



These wires are used with breadboard and development boards to temporally connect electronic components together. Jumper wires can have male or female ends depending on how they need to be used.

CHAPTER 4 – Arduino Development Board

In this chapter, we will learn about Arduino Development Boards. The knowledge about Arduino is essential because the main electronic component in MPatrol1 Learning Kit is an Arduino Development Board.

What is Arduino? Arduino is an open-source electronics and software platform based on easy-to-use electronic Learning Boards and Software. Arduino is the most popular Electronic/Software Learning Tool in the world now and thousands of students worldwide use Arduino Boards for Learning Electronics, Programming, and do a lot of Creative Projects.

Thanks to its simple and accessible user experience, Arduino has been used in thousands of different projects and applications worldwide. There are a lot of Arduino boards available in the market. Arduino boards can read inputs from switches, read data from various sensors like temperature, pressure ..etc, control speed in motors, turn on lights, control robots to do different tasks, make a sound from buzzers and do a lot more.

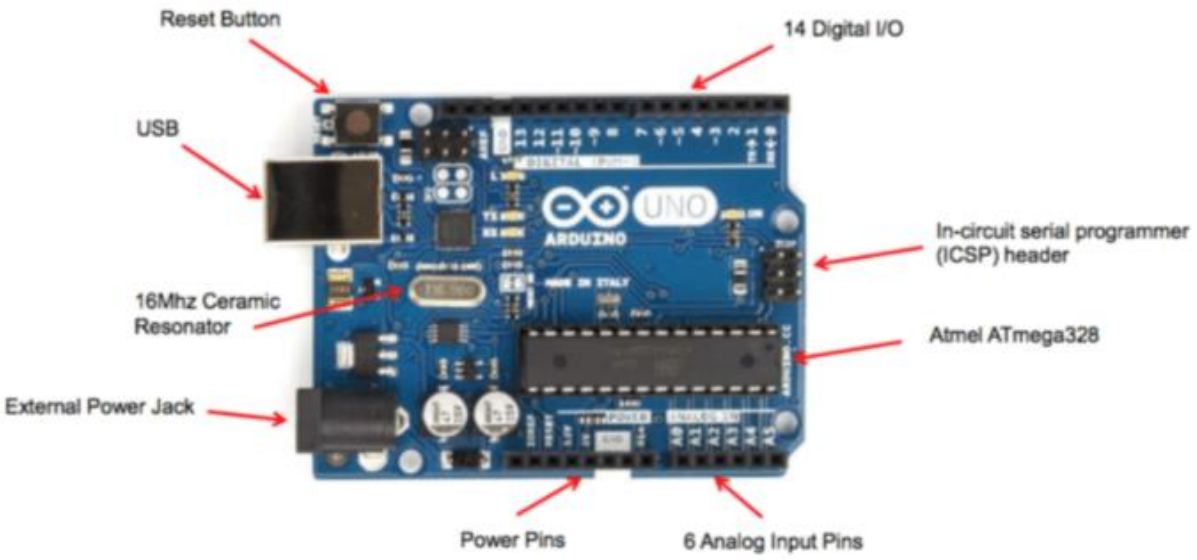
Main electronic component in an Arduino board is a Microcontroller. What is a Microcontroller? The microcontroller is a small computer. You can tell your Arduino board what to do by sending a set of instructions to the microcontroller on the board. It is called programming the microcontroller. It is the same as you do computer programming on your computer. When you write a computer program for your computer, that program runs on your computer. When you are programming a microcontroller, your program run in the microcontroller. It can do this because the microcontroller is also a tiny computer.

How to Program the Microcontroller in an Arduino Board? You can use different programming languages and different software for program an Arduino board. If you are using Arduino IDE Software for programming an Arduino Board, you can use “C” Computer Language. As “C” is an advanced programming language, we use “Scratch” Programming Language for programming the Arduino Boards in this tutorial. “Scratch” is a simple block-based visual programming language that can easily learn for beginners, who are new to the computer programming world. We use mBlock software for programming with “Scratch’ in this tutorial. All the necessary details about mBlock software have been explained in a separate chapter.

If you familiar with “C” programming and like to program with “C” Programming Language, you can still program this MPatrol1 Learning Kit using “C” Programming Language. For programming with “C”, you can use same mBlock software or Arduino IDE software. If you are using the Arduino IDE software, you have to download and install the Arduino IDE Software using below link. https://www.arduino.cc/en/Main/Software Arduino web site provides all the information about how to install Arduino Software and Configuration etc.. https://www.arduino.cc/

What is Arduino Uno?

As I mentioned earlier, there are many Arduino Development Boards in the market. They have different features and use for various purposes. Among them, Arduino Uno is the most famous board in the Arduino family.



The above figure shows an Arduino Uno Board. As you can see, it has various components. Now we will identify some important components of an Arduino Uno Board.

Atmel ATmega328 - This is the Microcontroller in the Arduino Uno Board. We can program this microcontroller for doing different tasks by using a computer programming language. We write the program on a computer and download it into the microcontroller, using the USB cable.

Digital I/O Pins – Arduino Uno board has 13 Digital Pins, and they have numbered from 0 to 13. We can connect different electronic components such as Switches, Bulbs, Motors, Sensors for these pins. Then we can control these components in different ways by programming the microcontroller in the Arduino Uno board.

Analog Pins – Arduino Uno board has 6 Analog pins, and they have numbered from A0 to A5. We can connect different Analog Sensors such as Temperature Sensors, Humidity Sensors …etc. for these pins. We can then program the microcontroller in Arduino Uno board to read the values from these sensors and display them.

USB Port - This USB port is using for several tasks. It can use for downloading the program from a computer to the microcontroller in the Arduino Uno board. It can also be used for supply power to your Arduino Board and the rest of the circuits, so you do not need a

separate power supply. The USB port also can use for reading data from the Arduino Uno board and display them on the computer. For example, you can connect a temperate sensor to one of the Analog pins and display the temperature values in your computer using the USB port.

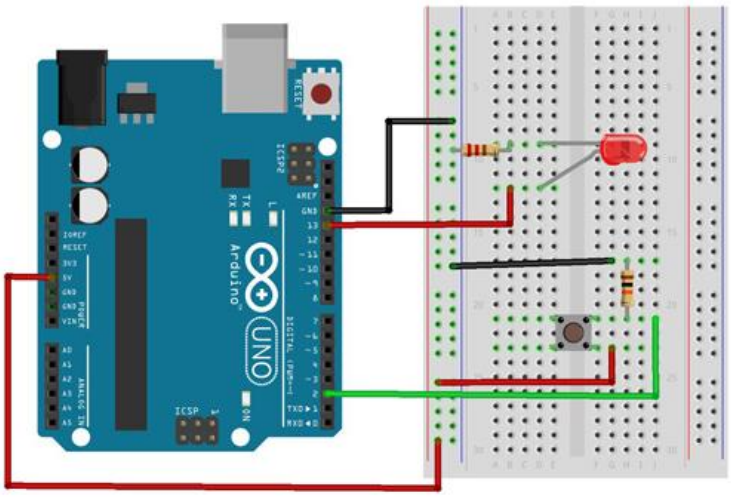
External Power Jack – This port use for supply power to Arduino Uno board and connected electronic circuits. In most of the cases, you can power your board from the USB port, by connecting it to your computer over the USB cable. (If connected electronic components do not consume a large current.)

Reset Button– This switch can be used to reset the microcontroller in the Arduino Uno board, so the microcontroller program can run from the beginning.

Power Pins – Arduino Uno board has few pins which output 5V and 3.3V. These power pins can power the attach electronic components to the Arduino Uno board, such as sensors.

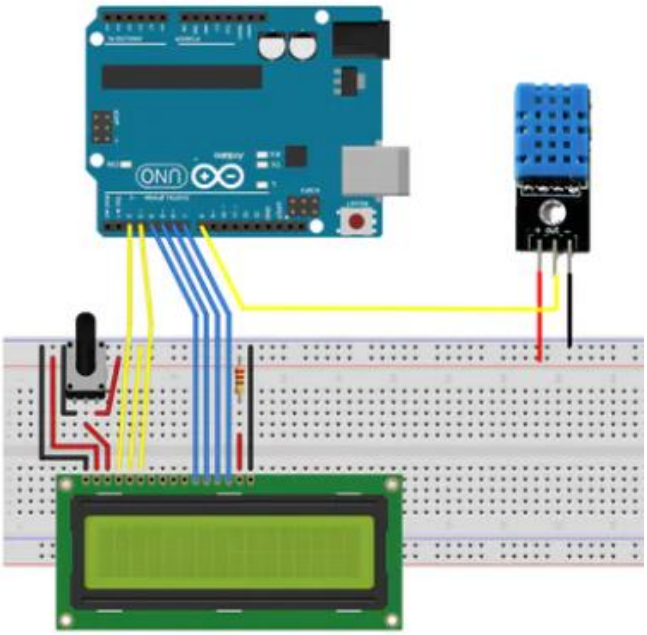
Now we will see few example circuits for Arduino Uno board.

(1) On/Off a LED Bulb using a Push Button Switch



Here we have connected a LED Bulb and a Push Button Switch to Arduino Uno board Digital Pins. We can program the microcontroller in Arduino Uno board to On the Bulb when Push Button press and off the bulb, when Push Button release.

(2) Read data from a Sensor and Display the values on a LCD Display



We have connected a temperature/humidity sensor, push button, and LCD display to an Arduino Uno board in the above figure. We can program the microcontroller in Arduino board to read temperate and humidity data from the sensor and display the values on LCD display, when push-button pressed.

Now you can see that we can do many exciting projects using this Arduino Uno board, by programming it in different ways. This is just an introduction about Arduino Uno Boards. You can learn about the Arduino Uno board and programming it in detail when doing the projects with MPatral1 Learning Kit

CHAPTER 5 – mBlock Software

mBlock is a programming software designed for Science, Technology, Engineering, Arts and Mathematics (STEAM) education. Inspired by Scratch 3.0, it supports both graphical and textual programming languages. Currently, more than 10 million people are using it to learn programming, create their own projects, and share their creations. With mBlock, you can design engaging stories, games, and animations, and program devices such as Robots, Arduino boards and Web Connected Devices. In addition, mBlock supports the Python language. You can switch to the Python mode simply with one-click. Moreover, mBlock integrates cutting-edge technologies including Artificial Intelligence (AI) and Internet of Things (IoT).

Why mBlock for us?

mBlock is the software, we are going to use for programming our MPatrol1 learning kit. We use mBlock for programming Arduino Uno boards by using visual programming blocks. (Scratch programming language). While programming with Scratch programming language, related "C" code is also generating by side. This will be helpful for students, those who like to learn advance programming languages like "C". This tutorial only focusses on programming the Arduino Uno boards using the Scratch programming language, but you can use mBlock for gain a lot of different programming and electronic skills by self-study the materials in mBlock website. Refer https://www.mblock.ccmBlock web site for more detail.

mBlock5 Software Download and Install

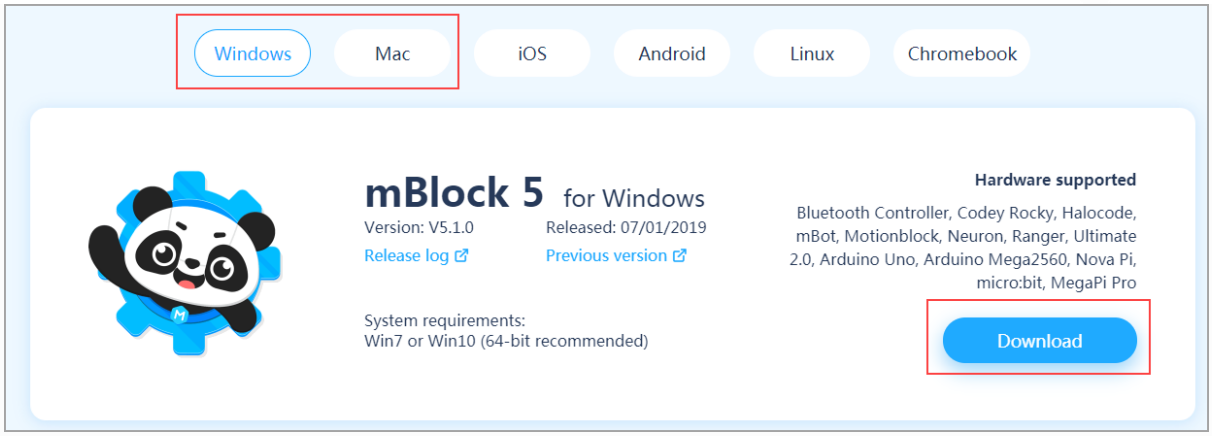
mBlock5 is the latest version of mBlock software family, Inspired by Scratch 3.0. In our tutorial we use mBlock5 for programming MPatrol1 Learning Kit in all our projects. Now we will see how to download the mBlock5 software and install in your computer or laptop. This explanation is only for the windows and Mac devices. For Android and iOS, please see **mBlock 5 for Android and iOS** link for all the installation details.

OS Requirements

Windows 7.0 or later; Mac OS X or later

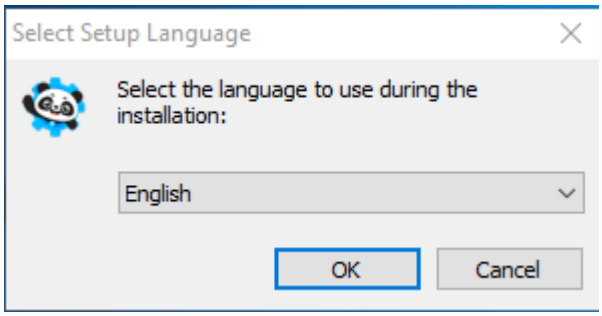
Download the software

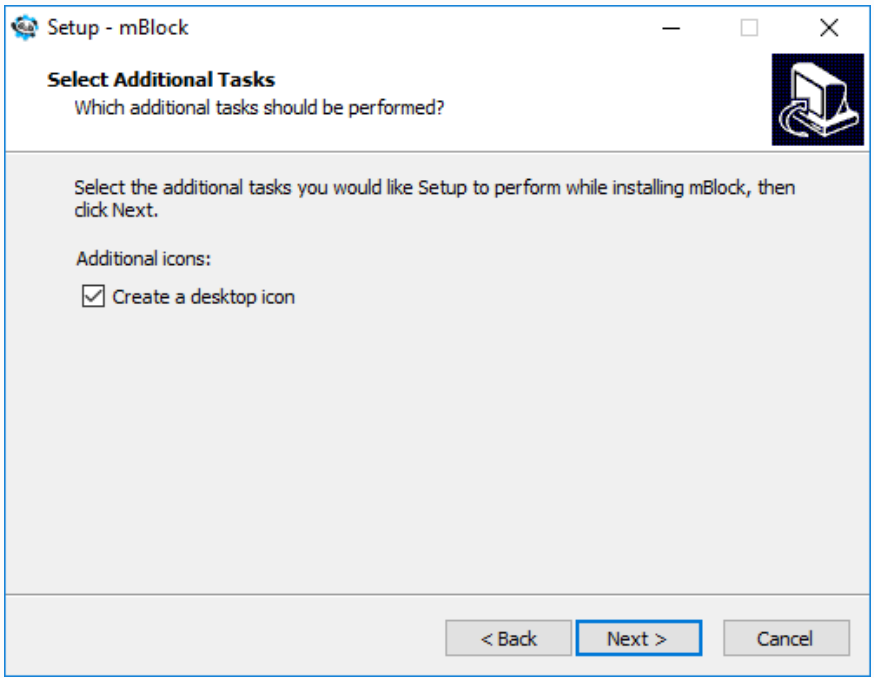
Visit https://www.mblock.cc/en-us/download to download the Installation file for PCs.



Install mBlock For Windows

(1) Double-click the installation file and follow the installation wizard.

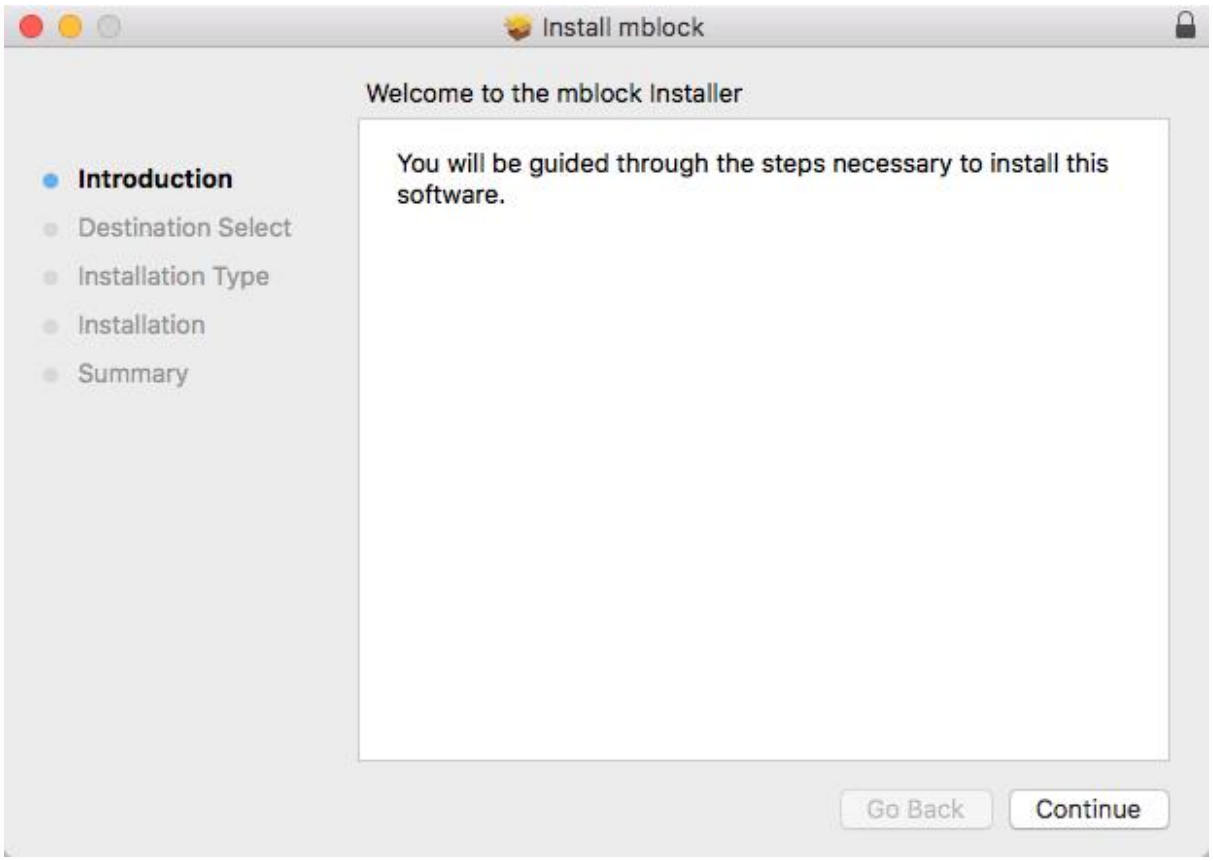


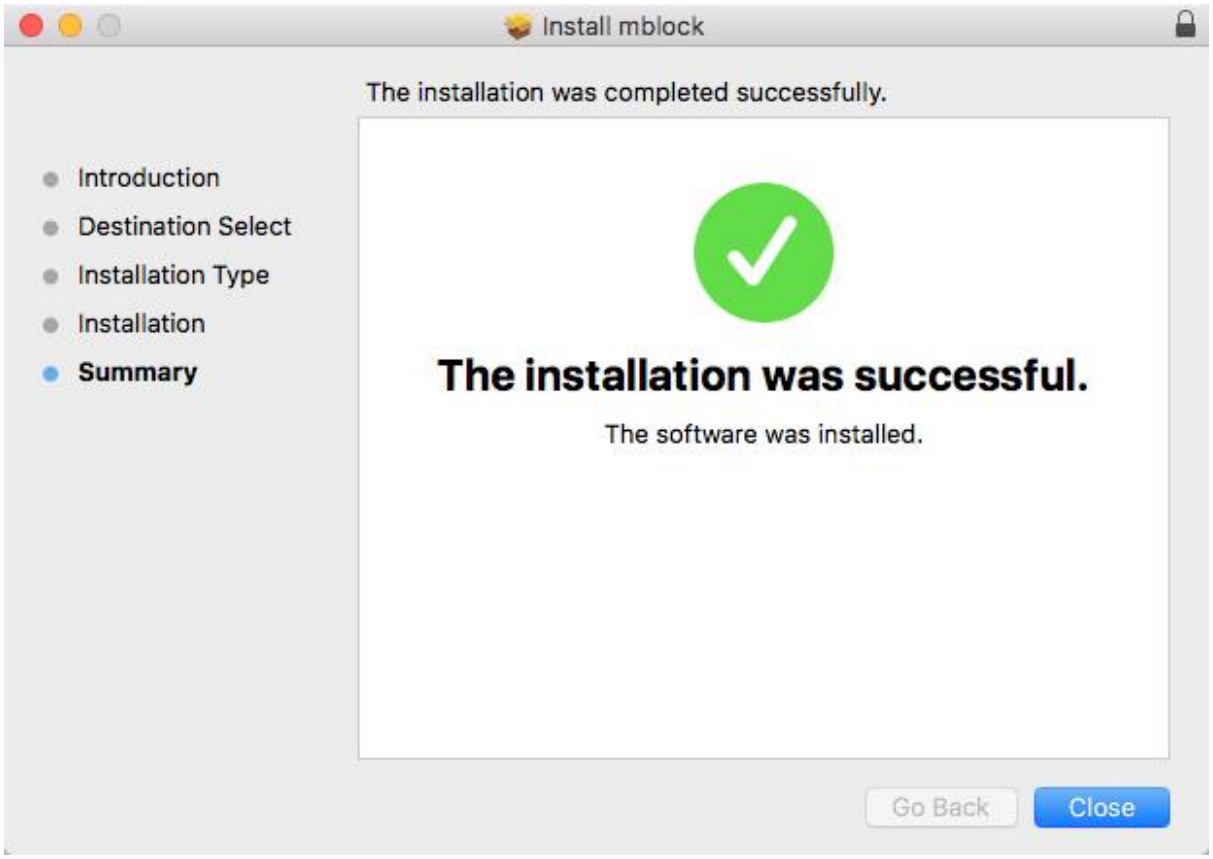


(2) After the installation is complete, the mBlock 5 icon is displayed on desktop.

Install mBlock For Windows

(1) Open the installation file and follow the installation wizard.



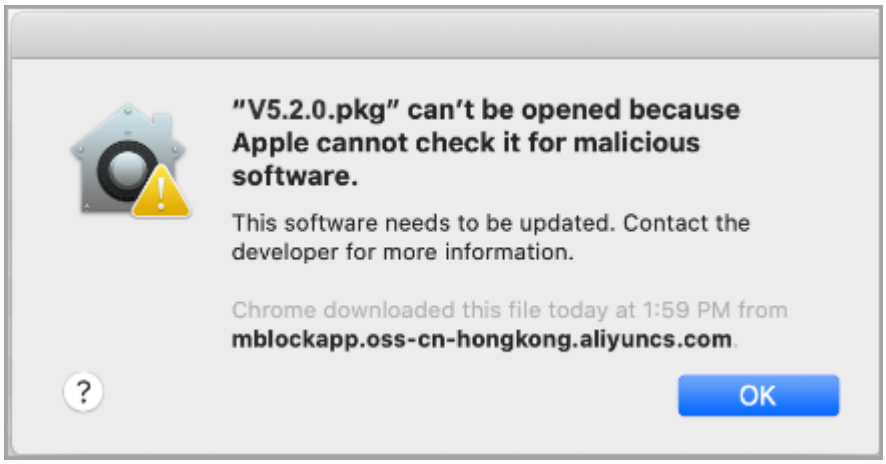


(2) After the installation is complete, the mBlock 5 icon is displayed on the launchpad and taskbar in the upper right corner.

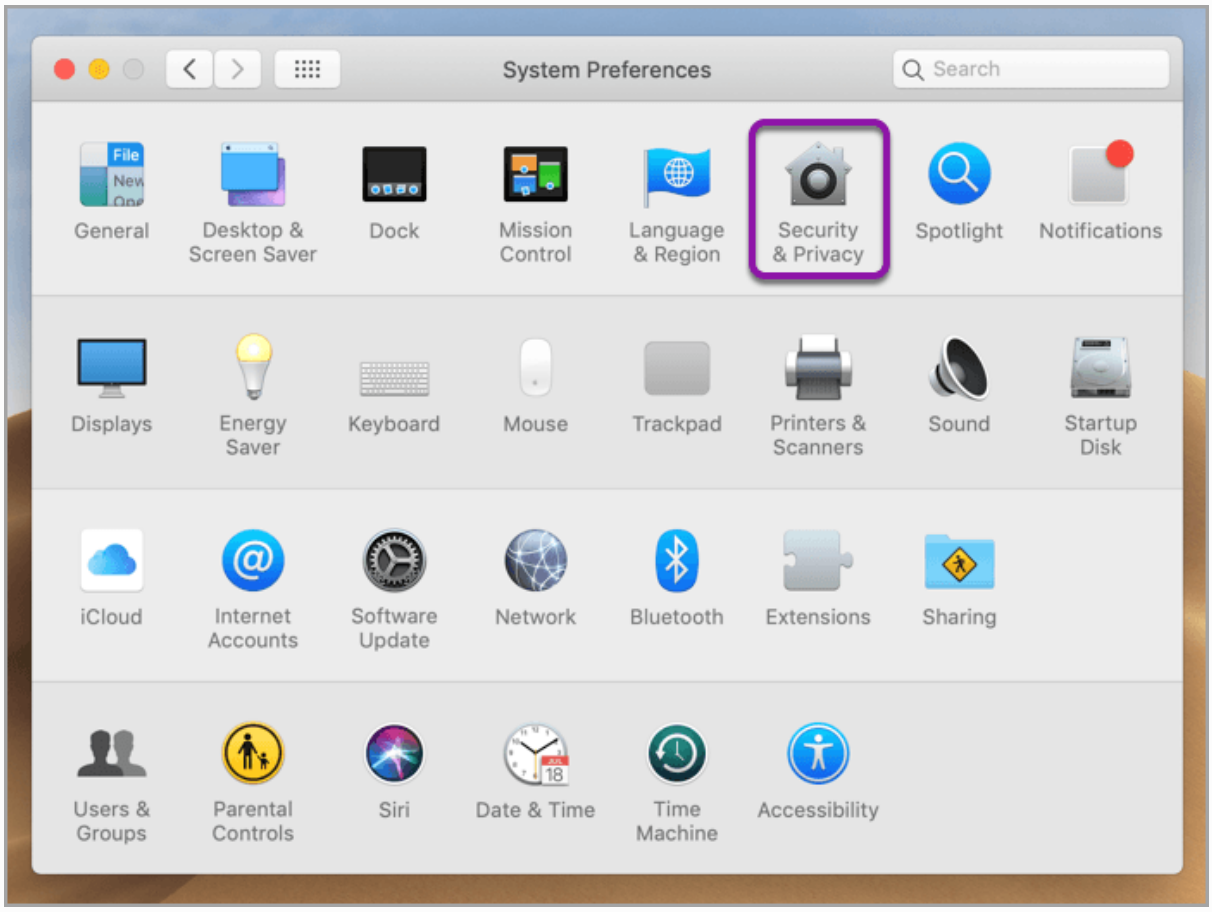
In macOS 10.15 or later systems, a warning message may be displayed after you double-click the installation package. You can install mBlock 5 by using one of the following methods.

Method 1

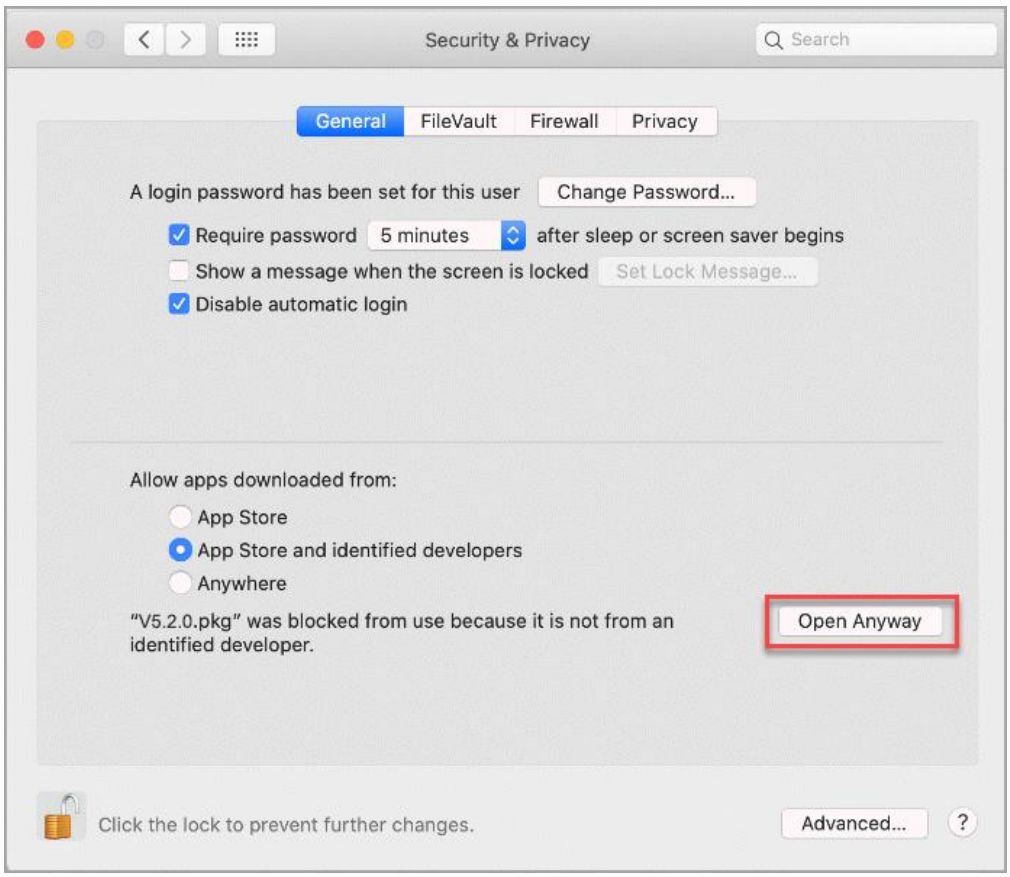
1. Click **OK**.



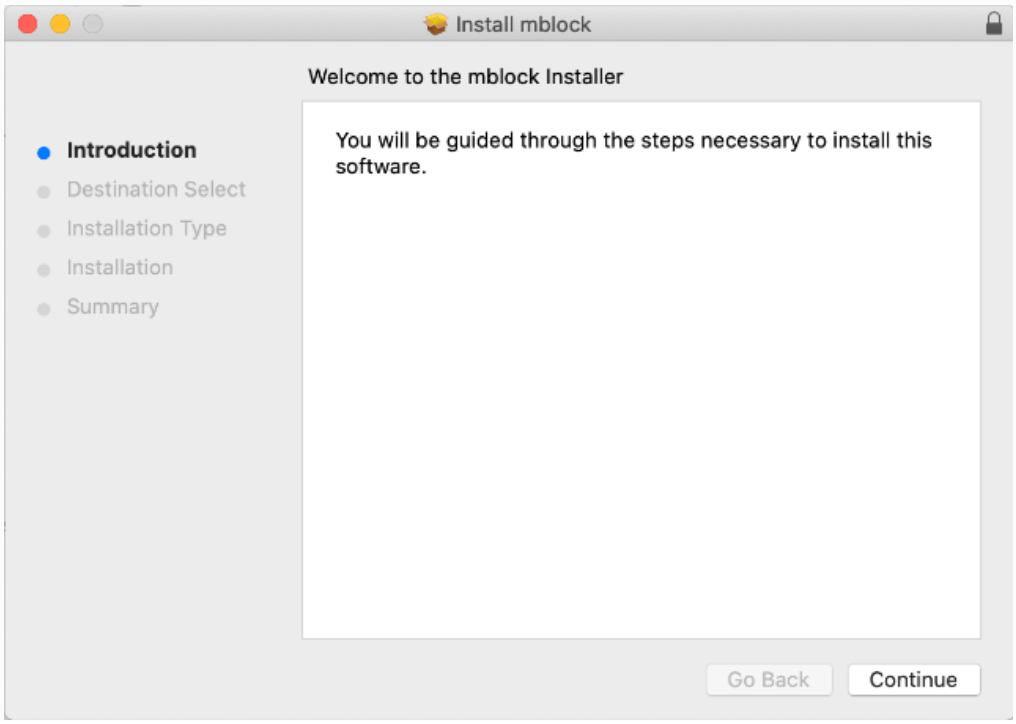
2. Choose System Preferences > Security & Privacy.



3. Click Open Anyway.

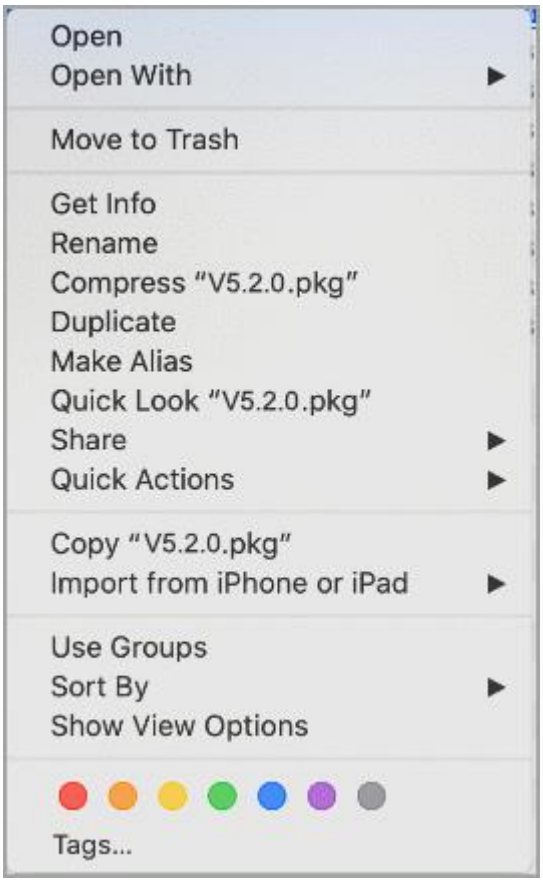


4. Start the mBlock 5 installation process.

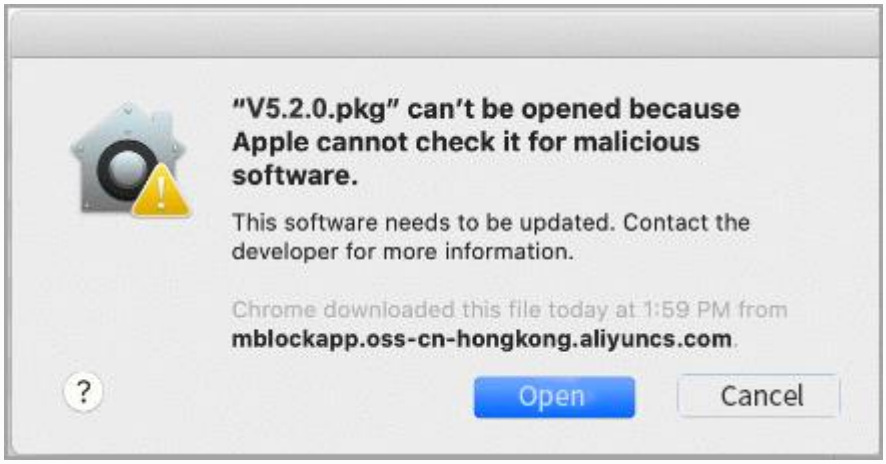


Method 2

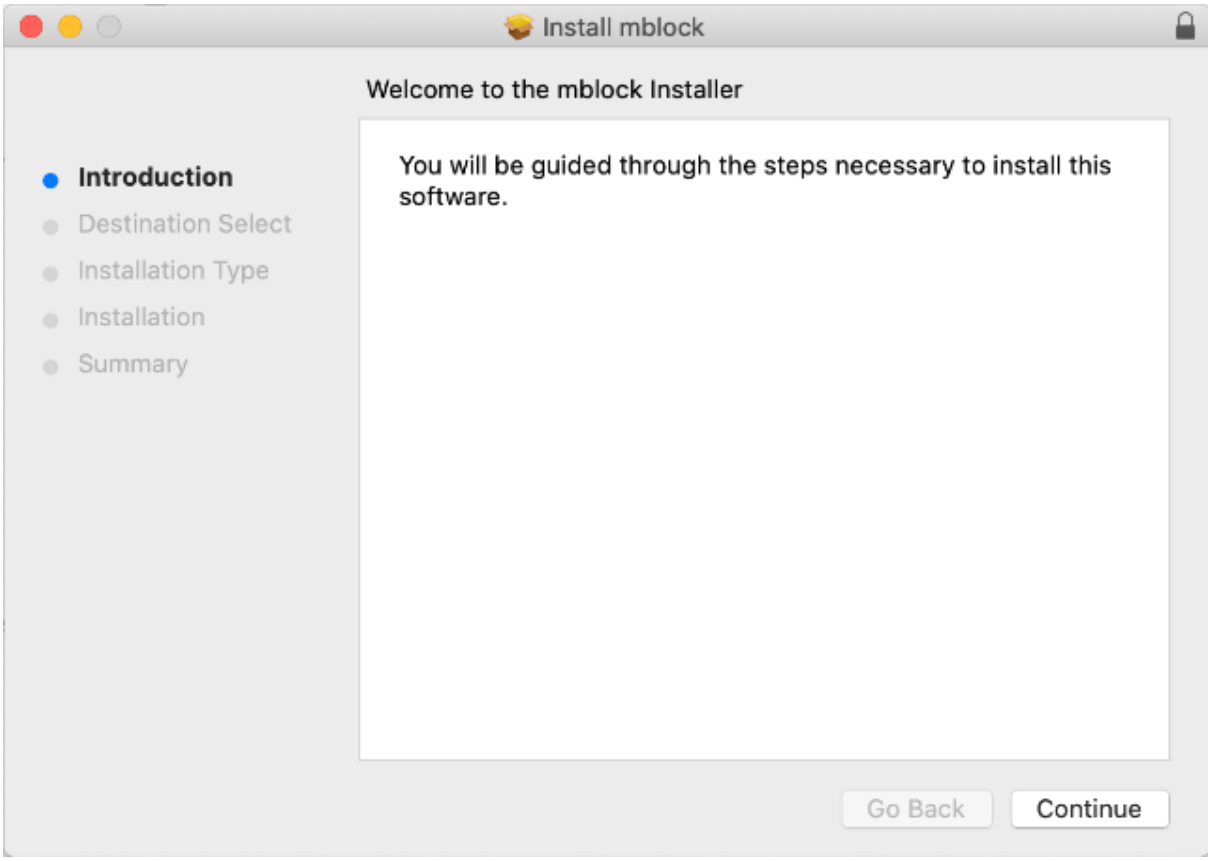
1. Right-click the installation package and choose Open.



2. Click Open



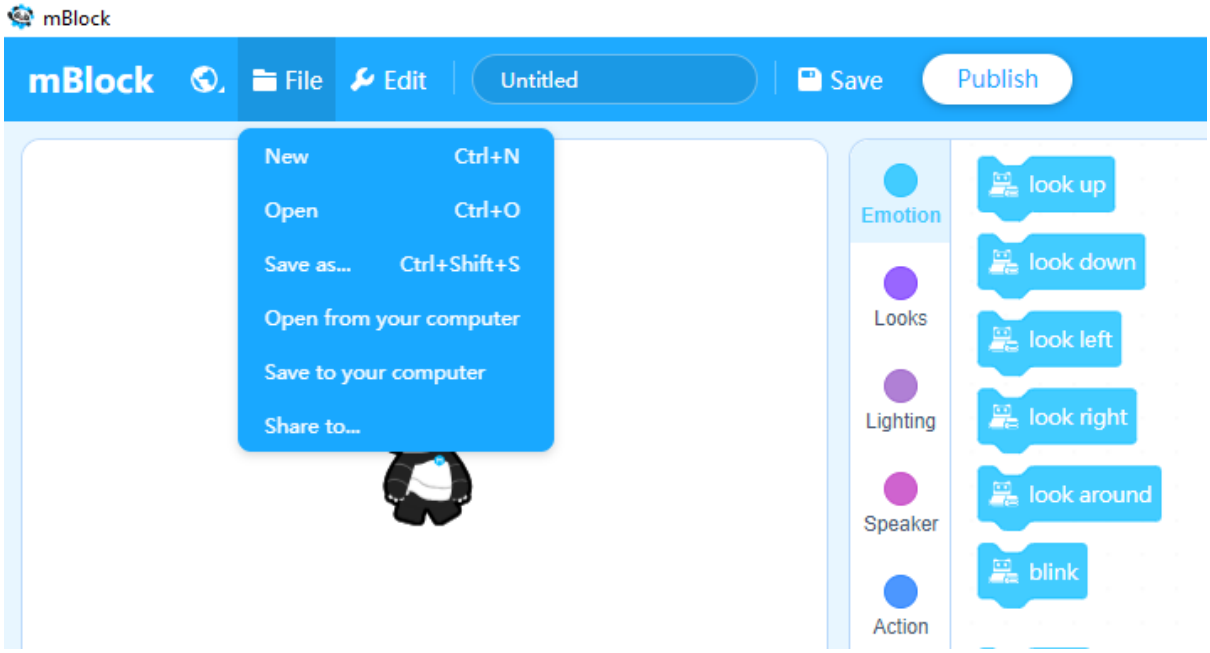
3. Start the mBlock 5 installation process.



Identify the main components of mBlock5 software

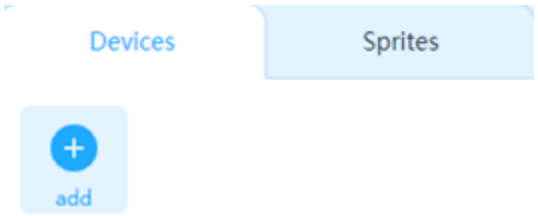
Now we will try to identify the main components of mBlock software. Double click the mBlock icon on the Desktop for open the software.

(a)File Menu



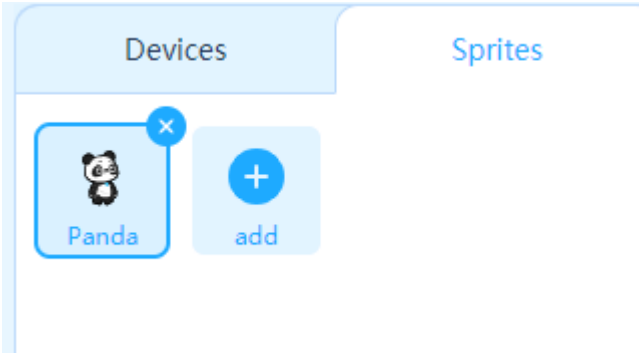
In the Top right hand corner you can see the **FILE Menu.** Using this menu, you can start a new project, can open an existing project, can save your project to computer or can save your project in a different name.

(b) Device Selection



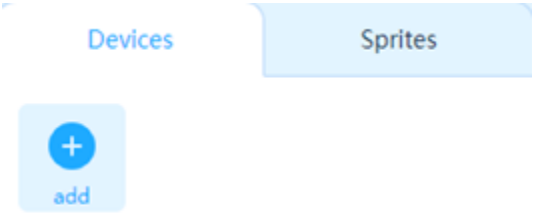
In the bottom left area of the software you can see two tables called “Devices” and“Sprites”. Under the “Devices”, you can select the device you are going to program. Under the “Sprites”, you can select the virtual device you wish to program. Using virtual devices, you can still learn programming by programing virtual devices, without any actual device connecting to your computer.

(c)Sprites Programming



These are virtual devices, which can program in different ways. You can see the virtual devices under "Sprites" Tab.Main virtual device in MBlock software is the **"Panda".**You can program this Panda to have different motions, to wear different costumes or to shout with different sounds.etc. Even you can create your own virtual devices and program them. This virtual device programming feature is very useful for the kids, those who are very new to programming for gain basic programming skills. However, scope of this tutorial is programming the actual devices connected to computer, I am not going to explain about virtual devices topic. But for those who interest about Sprites programming, there are nice and easy to learn tutorials available in the mBlock web site.

(d)Device Programming

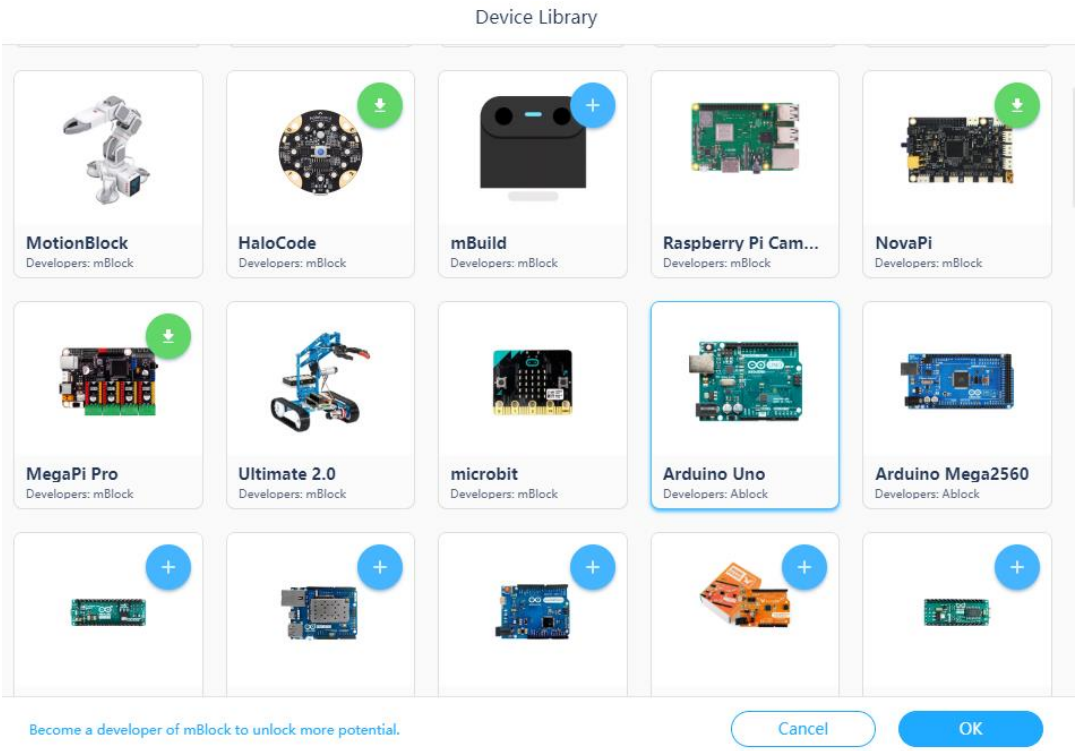


"Devices" tab contains the actual devices, we are programming by connecting to the computer. mBlock software support for programming a lot of devices. If you click the "add" button, you can see all the devices, which mBlock software is supporting for programming. As I mentioned in Arduino Chapter, in MPatrol1 Learning Kit, we are programming a "Arduino Uno" board. "Arduino Uno" is the brain of MPatrol 1 Learning Kit. So now we will see how to add the "Arduino Uno" board to mBlock devices.

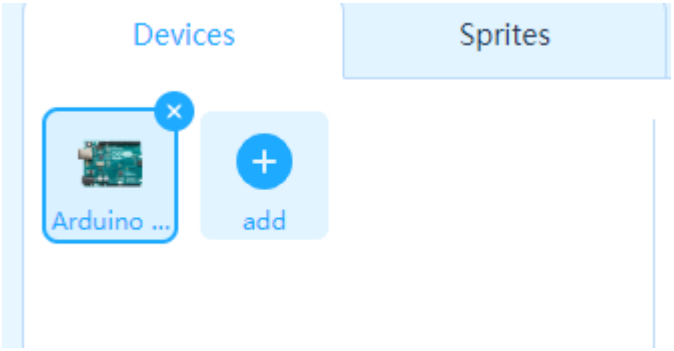
(e)Adding an Arduino Uno board

For this, first remove the "Codey", by clicking on the cross mark on topright-hand corner of "Codey". Then click the add button in "Devices" Tab for go to the Device Library. Now you can see several devices, which mBlock software support for programming. Depending

on the device, you can have “C” Or “Python” programing options in addition to Scratch Blocks programming.



Select “Arduino Uno “device and click “OK” to confirm.



Now you can see Arduino Uno board under “Devices” Tab.

Programming Blocks

In the middle of the screen, you can find all the default programming blocks, which are compatible for program the "Arduino Uno" board. Blocks have categorized into several categories depending on their task called Pin, serial port, Data, Sensor, Events, Control etc. Once you click on a category, you can see all the block programming instructions under that category. By default, first category (Pin) has selected, and you can see all the block programming instructions in "Pin" Category in above picture. Below I mention brief introduction about each instruction category. You can learn about these instructions in detail when you are doing projects with MPatrol1 Learning Kit.

(a) **Pin** – This contains instructions for read the input pin values of Arduino board, Set Arduino pin values high/low voltage, output PWM signals from Arduino Pins etc.

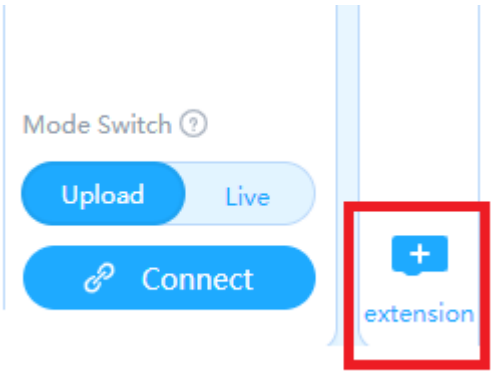
(b) **serial port** – This contains instructions for read/write data from/to Arduino board's serial port.
(c) **Data** – This contains instruction for string/number converting, number mapping etc..
(d) **Sensor** – This contains instruction for reading data from sensors and using timers.
(e) **Events** – This contains instruction for even handling
(f) **Control** – This contain program control instructions like if..Else, forever, repeat, wait etc..
(g) **Operators** – This contain mathematical instructions like addition, subtraction, multiplication, compare etc..
(h) **Variables** – This contains instructions for variable handling.

Extensions

Nowadays we have a lots of electronics components in the market. Different kind of sensors, LCD modules, Bluetooth modules, Wi-Fi modules.. Etc. Practically mBlock software cannot provide instruction blocks for all of these component and devices. So how mBlock has solved this problem? It has given a platform for public to develop instruction blocks for electronic components. These instruction blocks are called Extensions. As an example if you buy a temperature sensor from the market, you can make instruction blocks for integrate that sensor into mBlock software using mBlock's extention builder platform. Good news is most of the time you do not need to create instruction blocks for most of the electronic components you use. People around the world have already developed them and you can add these blocks to your mBlock project using the extension library.

Extension Library

You can find the Extension Library in middle bottom area of the mBlock software.



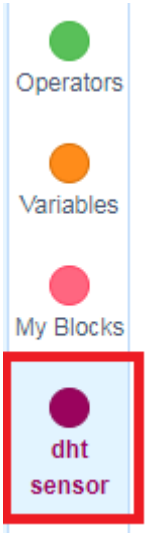
Click on the **"+"** sign to go into the Extention Library. Then you can see all the available extentions for diffrent kind of electronic devices. You need to have an internet connection for your computer to see all the extentions in Extention Library as they load in run time.

Adding Extentions

When you want to add an extention to your project

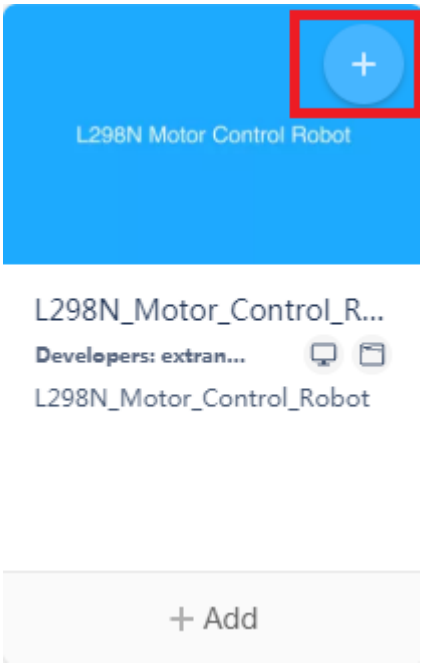
1. Goto Extention Library
2. Select the Extention you want to add to your mBlock project.
3. Click the “+Add” button in the bottom of the extention.

Then the extention will automatically add to your project as the last instruction catagory in the middle area of the mBlock software and you can straight away use the block instructions in it. Once you click on this instruction extention, you can see all the block instructions belongs to it.



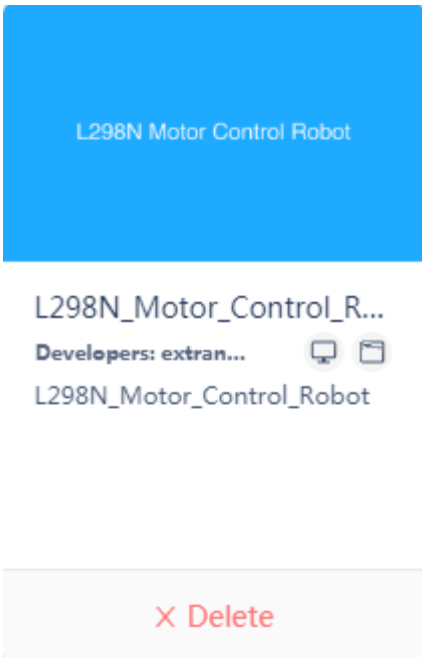
DHT temperature/humidity sensor extention after adding it to mBlock software.

Downloadable Extentions



You will notice for some extentions, you can not click on the **"+Add"** button and it has disabled. So before add this type of extentions to your mBlock project, you have to download the extention to your computer. You can do this by clicking the **"+"** sign in upper right corner of the extention. Sometimes download fail in the first attempt, so if that happened click again on **"+"** sign for retry download again. If the download is successfull you can see "successfull download message" and **"+Add"** button get enable. Then simply click on the **"+Add"** button to add the extention to your project.

Deleting Extentions



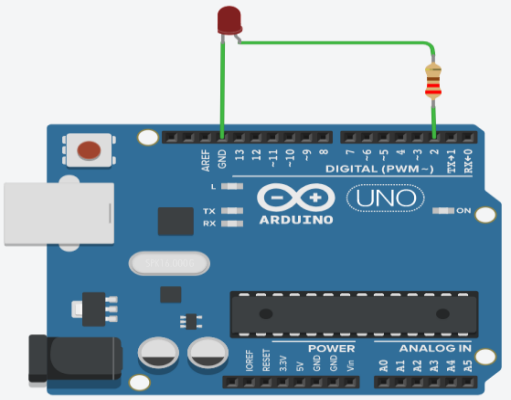
If you goto Extention Library, you can see the added extentions with word **"x Delete".** If you want to remove the extention from your project, click on the **"x Delete"** button and the extention will get remove from your mBlock project.

Writing a program in mBlock software using Scratch.

So far we learned all the tools we need to do programming with mBlock software. Now we will look how to write a program in mBlock software, using the Scratch programming language (using block instructions) and download it into MPatrol 1 Learning Kit.

LED Blinking Program

In MPatrol 1 Learning Kit, we have 2 LED bulbs in red and green colour. Lets write a programe to blink the red LED bulb. We explain this topic in detail when we start doing projects with MPatrol 1.



In MPatrol1, RED LED has connected to Pin 2 of Arduino Uno board. We connect LED to Uno board through a resistor, for the safety of the LED bulb. For turn on the LED, we have to set Pin 2 of Arduino Board, in HIGH state. For turn off the LED, we have to set Pin 2 of Arduino board, in LOW state. HIGH state means Pin 2 has 5V. LOW state means Pin 2 has 0V. So we have to do below mention steps for blink a LED.

(1)Start Arduino Program

(2)Turn OFF the Red LED (LED is off at the begining of program)

(3)Repeat the below steps continuously

(4)Turn On the LED

(5)Wait 1 second

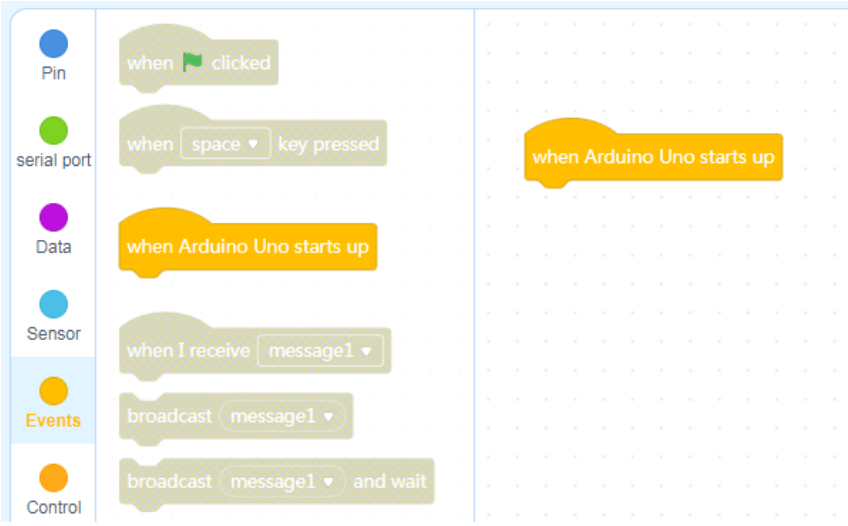
(6)Turn Off the LED

(7)Wait 1 second

(8)Go back to step (4)

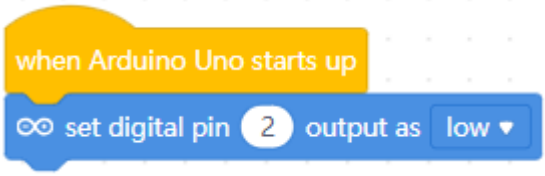
Now we will implement the above steps, using Scratch block instructions. In mBlock software, for use an instruction block, we have to drag that instruction to the right side big white space. This is the area we implement our program.

(1)Start Arduino Program – For this, we have to drag and drop the below block, from **"Events"** block catagory to right side large white area. This is the block every Arduino Uno program starts up.



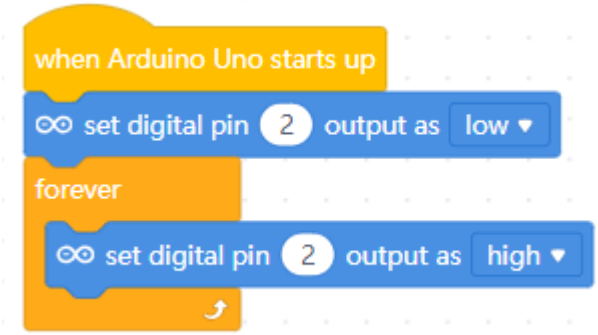
(2) OFF the Red LED – For this, drag and drop the below instruction from **"Pin**" block catagory.

Change the **"digital pin"** value as **"2"** and **"output as"** value as **"low".** Because LED bulb has connected to pin 2 of Arduino board and we want to off the bulb. Connect this block to **"when Arduino Uno starts up"** block, using the computer mouse. When you drag one block near to another block, they are connecting each other automatically.

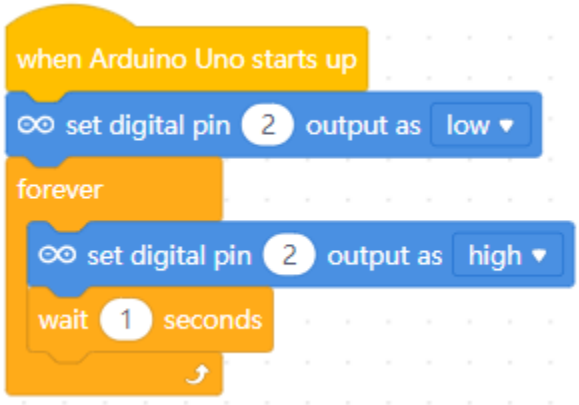


(3) Now we have to **Reapt some blocks continuouly**. This is for on and off the LED bulb continuously. For this, drag and drop the **"forever"** block from **"Control"** catagory and connect to the other blocks.

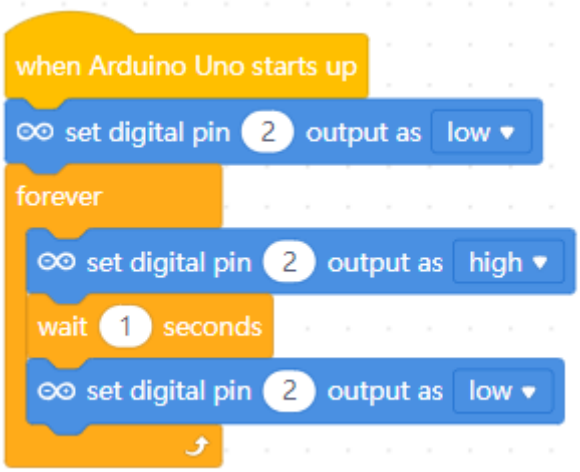
(4)Turn On the LED – These are the repeating instructions. So we have to implement them inside the forever block. For turn On the Red LED bulb, drag and drop the **"set digital pin 9 output as high"** instruction block from **"Pin"** catatory and fit it inside the forever block. Change the "**digital pin"** value as **"2".** Keep the **"output as"** value as **"high**". Because now we want to turn On the bulb.



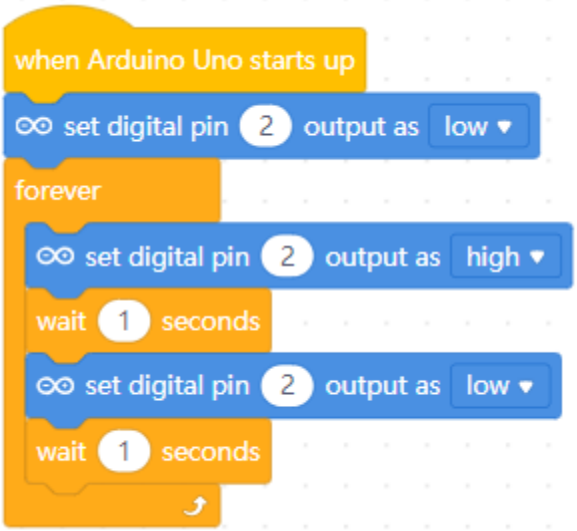
(5)Wait 1 second – This instruction keep the LED Light turn On for 1 second. For this, drag and drop the **"wait 1 seconds"** instruction from "Control" block catagory and fit it inside the **"forever"** block. We can control the blinking speed of the LED, using this "wait" instruction. If you change the value inside the "wait" instruction block, you can control the blinking speed of the LED bulb. Here keep the value "1" as it is, because our waiting time is 1 second.



(6) Turn Off the LED - For turn Off the Red LED bulb, drag and drop the **"set digital pin 9 output as high"** instruction block from **"Pin"** catatory and fit it inside the forever block. Change the "**digital pin"** value as **"2".** Chang the **"output as"** value as **"low**". Because now we want to turn Off the bulb.



(7)Wait 1 seconds – This instruction keep the LED Light turn Off for 1 second, before it turn On again. Do the same thing done in Step – 5. Again we can control the LED light blinking speed, by changing the time inside this wait block.



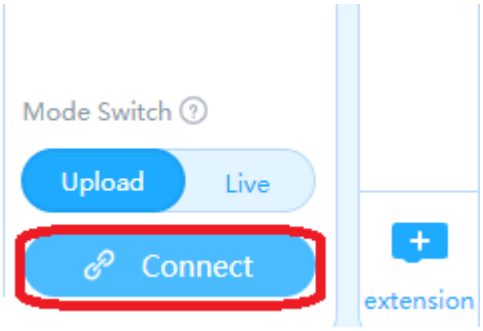
(8)Go back to step (4) – After run the **"wait 1 seconds"** instruction, program jump again to **"set digital pin 2 output as high"** instruction. Because any instruction inside this **"forever"** block, repeats again and again until power off the MPatrol 1 Learning Kit. This make Red LED blink (on and off) in one second interval, until power off the MPatrol 1 Learning Kit.

Upload the Program into MPatrol1 Learning Kit

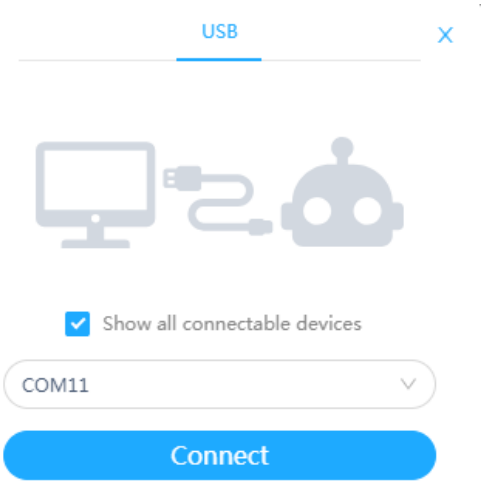
Now we have completed our LED blinking program. The next task is upload this program into MPatrol 1 Learning Kit. What actually happening here is, we are going to upload our program into the microcontroller (micro computer) in Arduino Uno board. Do below steps for this task.

(1)Connect the one end of the USB cable into MPatrol 1 Learning Kit and connect other end to computer.

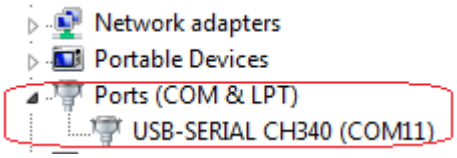
(2) Click on the **"Connect"** button, in middle bottom area of mBlock software.



(3)Tick the "Show all connectable devices" option. Then you can see all the available USB ports in your computer, which has some device connected. Select the correct USB port which your Mpatrol 1 has connected and click "Connect" button for connect the MPatrol 1 Learning Kit with your computer. If you got conection failure, try to reconect again.



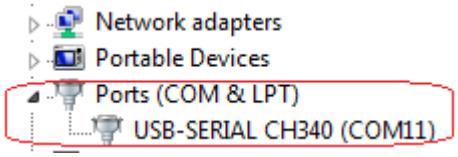
If you are not sure, which COM port, your device has connected, Goto **Device Manager -> Ports** . Then unplug your USB cable and plug it again to computer. When you unplug, your connected COM port should be disapear from the Port list. When you plug it again, the connected COM port should be re-appear again. In this way you can identify which COM, the MPatrol Learning Kit has connected.



The device driver name of MPatrol 1 is **"USB-SERIAL CH340"**. COM port number will depend on your computer. As shown in above figure, I have connected MPatrol 1 to COM port 11 of my computer.

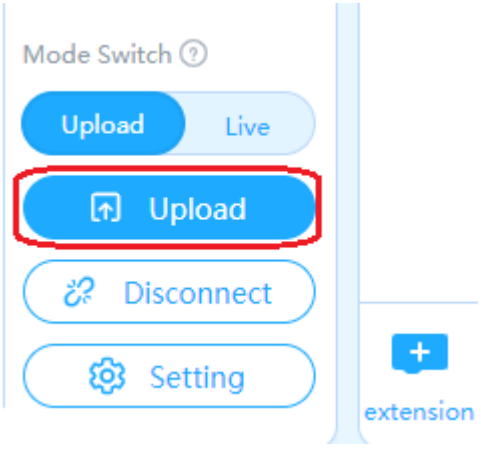
(4)If the MPatrol 1 is not connecting with your computer with any available COM port or if you can not see any available COM ports, it seems your computer has some USB-COM

port driver issue. If so Goto **Device Manager -> Ports** . Then check wether you can see a COM port called **"USB-SERIAL CH340"**. If you can not see any COM port with that name, that mean there is some USB-SERIAL driver issue. Sometimes you can see the COM port, but it has a yellow colour mark on it. That mean driver has not properly install in your computer. For both these cases, download the **"CH340" driver** from internet and install it on your computer. Or else, uninstall the mBlock software and re-install it again. When you install the mBlock software, it will ask from you, whether it is ok to install the driver. Say "YES" for that. Then try to connect to the MPatrol 1 again. If you do not have any issue with driver, you can see the connected COM port number infront of driver name. (Driver name = **"USB-SERIAL CH340"**) So in this way also, you can identify the exact COM port number, which your MPatrol 1 Learning Kit has connected. As shown in below picture, my MPatrol 1 has connected to COM port 11. This is not uniqe number and depend on your computer. So you may have a different COM port number for your MPatrol 1 Learning Kit.



(5) If the MPattrol 1 is not connecting even with the correct COM port, plug the USB cable to another USB port on your computer, identify the new COM port number and retry.

(6)Finally click on **"Upload"** button for upload your program into the MPatrol1 Learning Kit. You can find the **"Upload"** button, in middle bottom area of mBlock software.



Once you click on "Upload" button, two things are happening automatically.

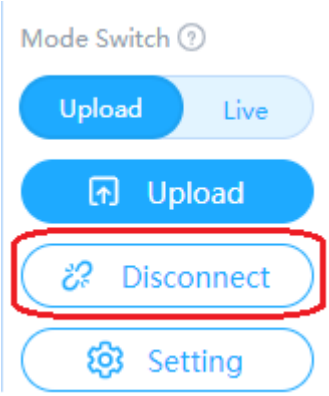
(a)First mBlock software compile your program. Compile means mBlock software convert your program into a machine language, which microcontroller inside the Arduino Uno board can undestand. If you have done some mistakes in your programme, you will get errors in this step. Then first you have to fix those errors.

(b)Second, mBlock software upload your program (machine language instructions), into the microctontroller inside the Arduino Uno board of MPatrol 1 Learning Kit. If you have connected your MPatrol 1 Learning Kit to a wrong COM port, you will get upload failure

errors in this step. Than mean your computer can not communicate with MPatrol 1 Learning Kit. If so first connect the MPatrol 1 with correct COM port in your computer. (Follow the 3, 4 and 5 steps for fix the COM port issues)

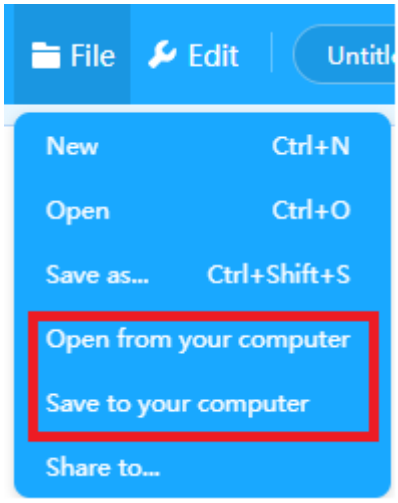
(7)After upload completed, your program automatically run inside the microcontroler of Arduino Uno board and you can see the Red LED blinking in your MPatrol 1 Learning Kit.

(8)If you want to disconnect your MPatrol 1 Learning Kit, simply click on the **"Disconnect"** button in middle bottom area of mBlock software.



Save your block instruction program in computer.

After completing your project, you can save the program code to your computer using the File menu. Click on "File" menu and select "Save to your computer" option. Then give the location in your computer and file name for save the file. If you want to re-open the saved file, double click on the saved file or select "Open from your computer" from file menu and select the saved file.

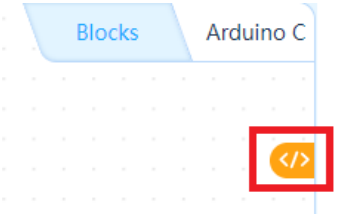


Writing programms in “C” Programming Language

If you are familiar with the “C” programming language, you can programme MPatrol 1 Learning Kit using the “C” programming langauge as well. For this click the “Arduino C” tab in top right hand corner of mBlock software and write your “C” codes in given white space. If you want to come back for write Scratch block instructions, click on the “Blocks” tab. Projects in this tutorial have only implemented in Scratch block instructions.

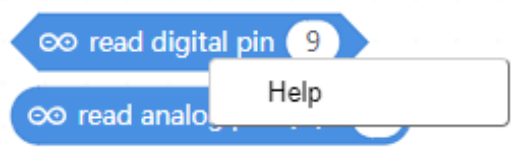
Learn “C” Language ,while programming with Scratch Blocks.

When you program with Scratch block instructions, related Arduino “C” language code for that block instructions is automatically generated by mBlock software. You can see this Arduino “C”code by clicking the two arrow mark in top right hand corner of mBlock software. In this way, you can learn advanced “C” programming language also, while programming with Scratch block instructions.

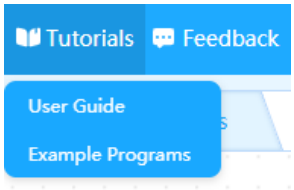


mBlock Help Documentation

Most of the blocks in mBlock software are coming with a help document. You can see this help document, by simply right click on the particular block and select “Help”. This document provide detial information about the selected block.



You can find more help documents from the “Tutorials” menu, in top right hand corner of mBlock software.



CHAPTER 6 – Example Projects

In the previous chapters, I have explained the basic knowledge about computer programming, electronics, Arduino development boards and mBlock software. Now it is the time to add all these knowledge together and do real world projects. In this chapter, I have explained 25 projects, which cover all the electronic modules in MPatrol 1 Learning Kit. So by completing these projects, you can get a good understanding about computer programming (coding) and electronics. Projects have written from simple to advance. Knowledge require to do some projects, may not explain in detail in that project, if they already covered in a previous project. So I strongly recommend doing the projects, according to the sequence written in this chapter. But if you think, you already know how to do some projects, you can skip them and go to the next one.

In the first few chapters, I have explained about Computer Programming (Coding), Basic Electronics, Arduino Development Board and mBlock software in detail. If you are not familiar with any of these topics, please read the relevant chapter or chapters before starting projects. In each project, I have explained all the required computer programming and electronics knowledge in detail and in simple manner. So I believe you can do these projects, without support from anyone.

All these 25 projects have implemented using Scratch 3.0 programming language and mBlock software. But if you familiar with the advance programming language called “C”, you can do these projects in “C” programming language as well, using the same mBlock software. You can start writing your “C” codes, by clicking the “Arduino C” Tab in top right hand corner of mBlock software. Further mBlock software automatically generates the “C” programming language code, for your Scratch 3.0 program code. You can see it by clicking the “two arrow mark” in top right hand corner of mBlock software. (Under the “Arduino C” Tab). So if interested, you can learn the “C” advance programming language also while doing the Scratch 3.0 programming. Further you can use the famous “Arduino IDE” software also, to do these projects in “C” programming language.

These 25 projects cover lots of Scratch 3.0 block instructions, available in mBlock software. Rest of the blocks you can experiment on your own with different project ideas. Further these 25 projects are just few example projects, you can do with MPatrol 1 Learning Kit, which cover the basic knowledge you need to use the MPatrol 1. Depending on your creativity and innovation ability, you can come up with a lot of project ideas and implement them on MPatrol 1 Learning Kit, using the electronics modules available in the Kit.

Project 1 - Turn On the Red LED

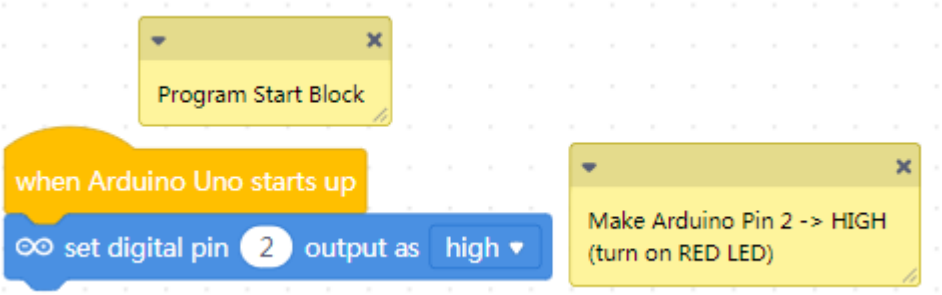
Project Details

In this project, we turn on the RED LED of MPatrol 1 Learning kit.

Arduino Uno pin connections

Red LED -> Arduino Uno Pin D2

mBlock Scratch Program Code



"set digital pin 2 output as high" block instruction drives Arduino pin 2 into "HIGH" state. That means Arduino pin 2 get 5V. Red LED has connected to Arduino pin 2, so it get 5V and turns On.

Note : For more details on mBlock Scratch programming, first read Chapter 5 of the tutorial document. (**Chapter 5 – mBlock Software**)

Project 2 - Blink the Red LED

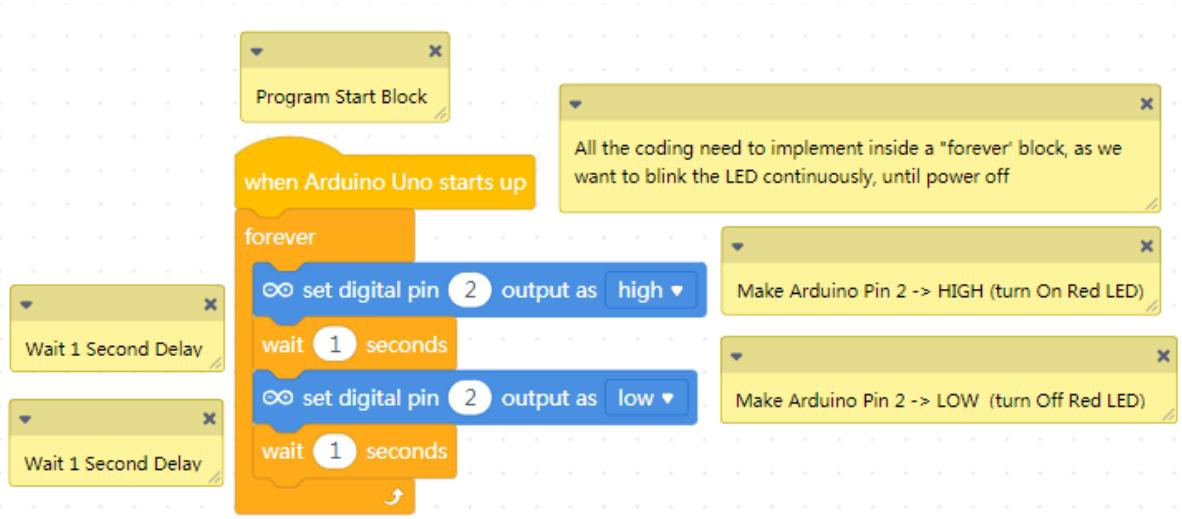
Project Details

In this project, we on and off (blink) the Red LED of MPatrol 1 Learning kit. On/Off timing (blinking rate) can change, by changing the wait delay time.

Arduino Uno pin connections

Red LED -> Arduino Uno Pin D2

mBlock Scratch Program Code



"set digital pin 2 output as high" block instruction drives Arduino pin 2 into "HIGH" state. That means Arduino pin 2 get 5V. Red LED has connected to Arduino pin 2, so it gets 5V and turns On. **"set digital pin 2 output as low"** block instruction drives Arduino pin 2 into "LOW" state. That means Arduino pin 2 get 0V. So Red LED get 0 V and turn Off.

This program blinks Red LED, in 1 second interval. Users can change the blinking rate of LED, by changing the time inside "wait" block. Eg. 0.5 second will blink the LED fast.

Note : For more details on mBlock Scratch programming, first read Chapter 5 of the tutorial document. (**Chapter 5 – mBlock Software**)

Project 3 - Blink two LEDs

Project Details

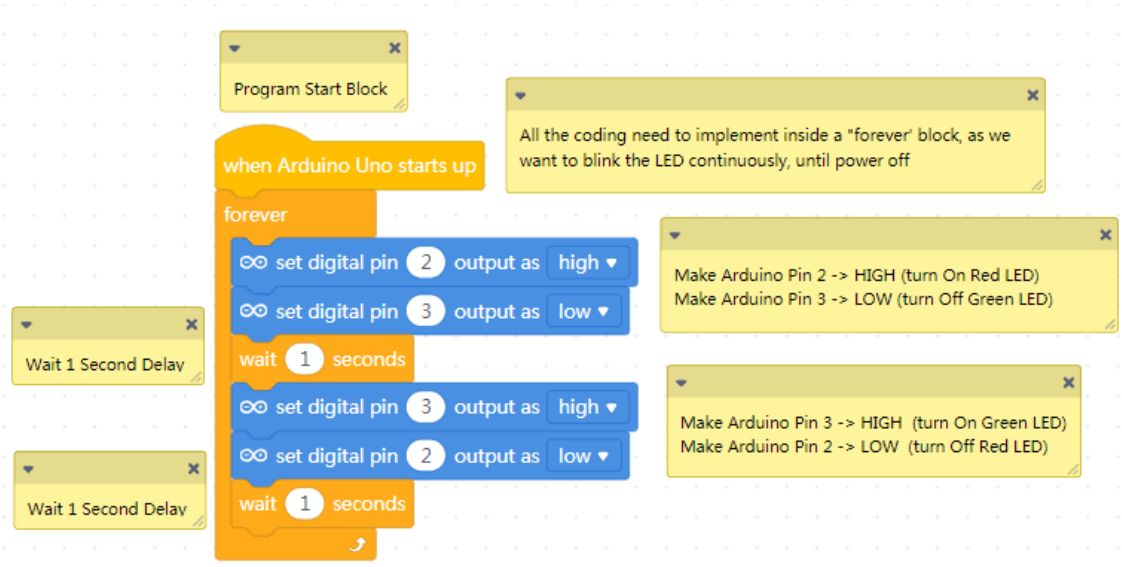
In this project, we on and off (blink) the Red and Green LEDs of MPatrol 1 Learning kit. On/Off timing (blinking rate) can change, by changing the wait delay time.

Arduino Uno pin connections

Red LED -> Arduino Uno Pin D2

Green LED -> Arduino Uno Pin D3

mBlock Scratch Program Code



"set digital pin 2/3 output as high" block instruction drives Arduino pin 2/3 into "HIGH" state. That means Arduino pin 2/3 get 5V. So connected LEDs get 5V and turn On. **"set digital pin 2/3 output as low"** block instruction drives Arduino pin 2/3 into "LOW" state. That means Arduino pin 2/3 get 0V. So connected LEDs get 0 V and turn Off. This program blinks Red and Green LEDs, in 1 second interval. Users can change the blinking rate of LEDs, by changing the time inside "wait" block. Eg. 0.5 second will blink the LEDs fast.

Note : For more details on mBlock Scratch programming, first read Chapter 5 of the tutorial document. (**Chapter 5 – mBlock Software**)

Project 4 - Generate a Buzz using the Buzzer

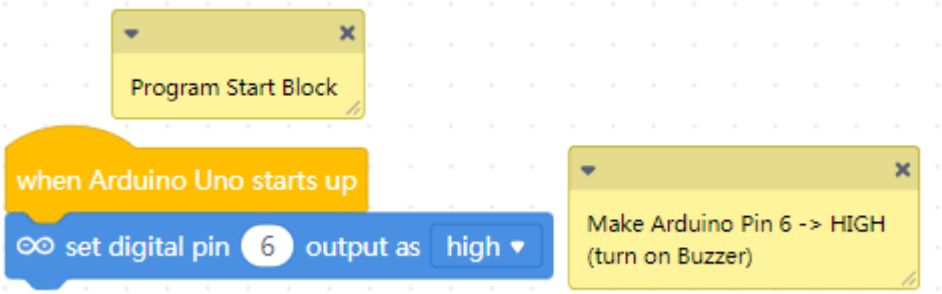
Project Details

In this project, we generate a sound using the Buzzer of MPatrol 1 Learning kit.

Arduino Uno pin connections

Buzzer -> Arduino Uno Pin D6

mBlock Scratch Program Code



"set digital pin 6 output as high" block instruction drives Arduino pin 6 into "HIGH" state. That means Arduino pin 6 gets 5V. So connected Buzzer gets 5V and turns On.

Note : For more details on mBlock Scratch programming, first read Chapter 5 of the tutorial document. (**Chapter 5 – mBlock Software**)

Project 5 -Emergency Vehicle Siren

Project Details

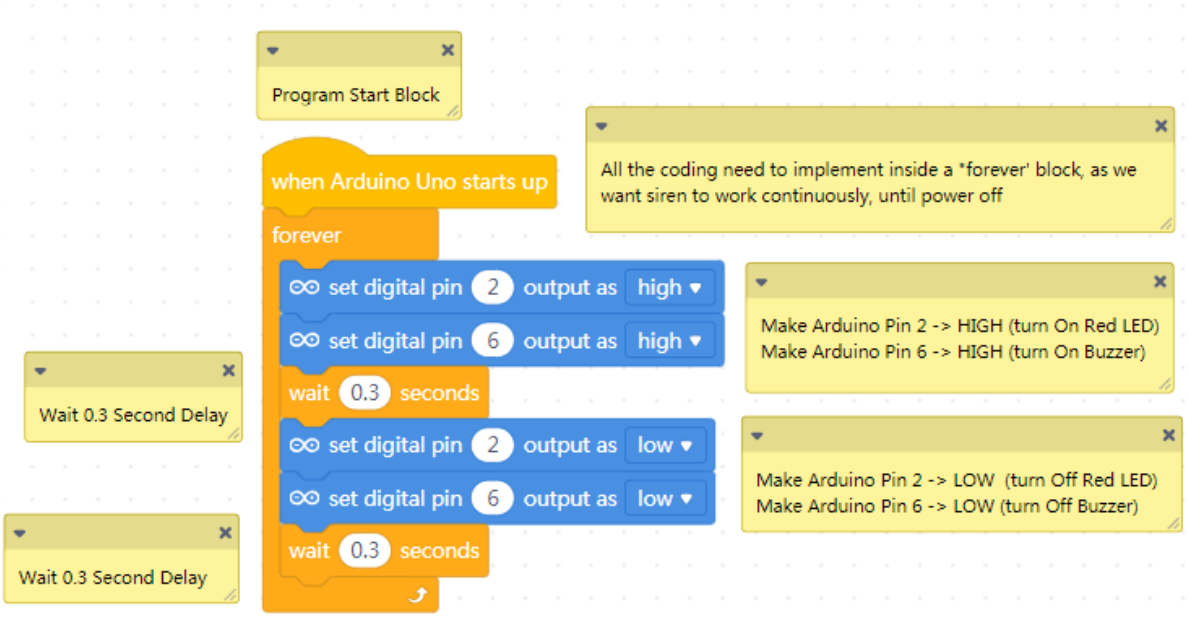
In this project, we make a Siren by blinking Red LED while on and off the Buzzer.

Arduino Uno pin connections

Red LED -> Arduino Uno Pin D2

Buzzer -> Arduino Uno Pin D6

mBlock Scratch Program Code



In this program, Siren is working with 0.3 second delay time. Users can change this siren timing, by changing the time inside “wait” block.

Note : For more details on mBlock Scratch programming, first read Chapter 5 of the tutorial document. (**Chapter 5 – mBlock Software**)

Project 6- On and Off a Bulb using a Switch

Project Details

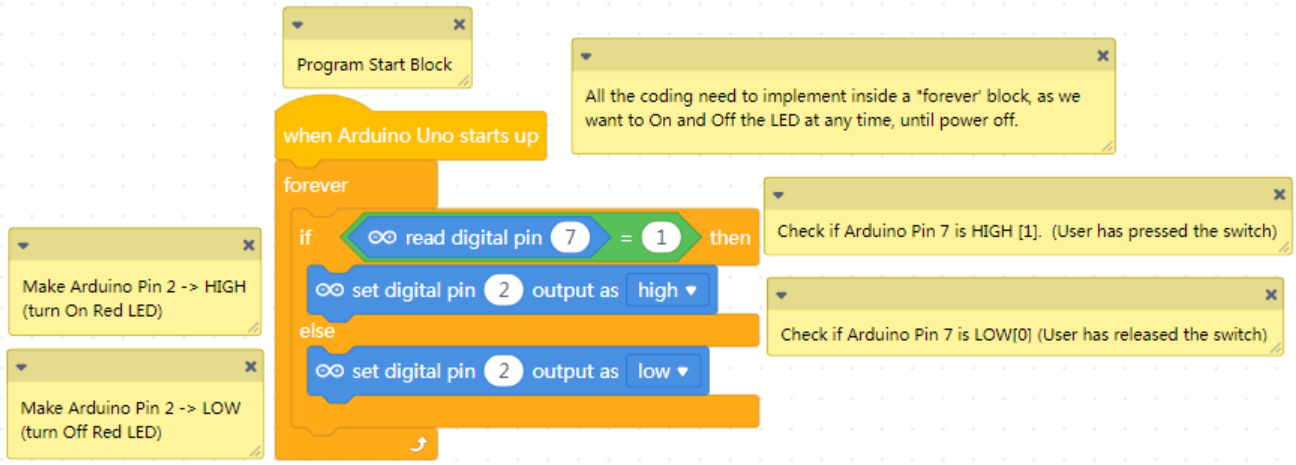
In this project, we on and off the RED LED bulb in MPatrol 1 Learning Kit, using the Push Button Switch. LED bulb turns on, when press the Push Button Switch and turns off,once release the switch.

Arduino Uno pin connections

Red LED -> Arduino Uno Pin D2

Push Button Switch -> Arduino Uno Pin D7

mBlock Scratch Program Code



Here using the **"read digital pin 7"** instruction block, we read the status of (HIGH[1] or LOW[0]) Arduino Pin 7. When press the switch, 5V connects to the Arduino Pin 7, hence pin 7 state gets "HIGH" (HIGH = 1). When release the switch, 0V connects to the Arduino Pin 7, hence Pin 7 state gets "LOW" (LOW = 0). In this way, programme can understand the status of switch, so we can do whatever we want, according to this status of the switch. Here we on and off a LED bulb, according to the status of the switch. (Status of the Switch = Press or Release)

Note : For more details on mBlock Scratch programming, first read Chapter 5 of the tutorial document. (**Chapter 5 – mBlock Software**)

Project 7 - On and Off a Buzzer using a Switch

Project Details

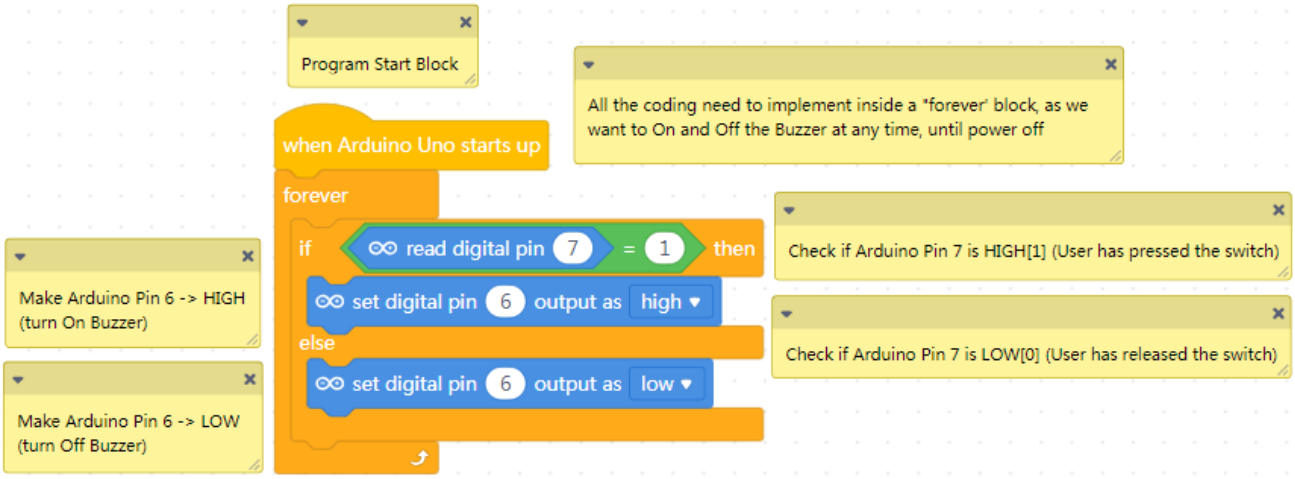
In this project, we on and off the Buzzer in MPatrol 1 Learning Kit, using the Push Button Switch. Buzzer turns on, when press the Push Button Switch and turns off, once release the switch.

Arduino Uno pin connections

Buzzer -> Arduino Uno Pin D6

Push Button Switch -> Arduino Uno Pin D7

mBlock Scratch Program Code



Here using the **"read digital pin 7"** instruction block, we read the status of (HIGH[1] or LOW[0]) Arduino Pin 7. When press the switch, 5V connects to the Arduino Pin 7, hence pin 7 state gets "HIGH"(HIGH = 1). When release the switch, 0V connects to the Arduino Pin 7, hence Pin 7 state gets "LOW" (LOW = 0). In this way, programme can understand the status of switch, so we can do whatever we want, according to this status of the switch. Here we on and off a Buzzer, according to the status of the switch. (Status of the Switch = Press or Release)

Note : For more details on mBlock Scratch programming, first read Chapter 5 of the tutorial document. (**Chapter 5 – mBlock Software**)

Project 8 - On and Off a Motor using a Switch

Project Details

In this project, we on and off the Motor in MPatrol 1 Learning Kit, using the Push Button Switch. Motor turns on, when press the Push Button Switch and turns off, once release the switch.

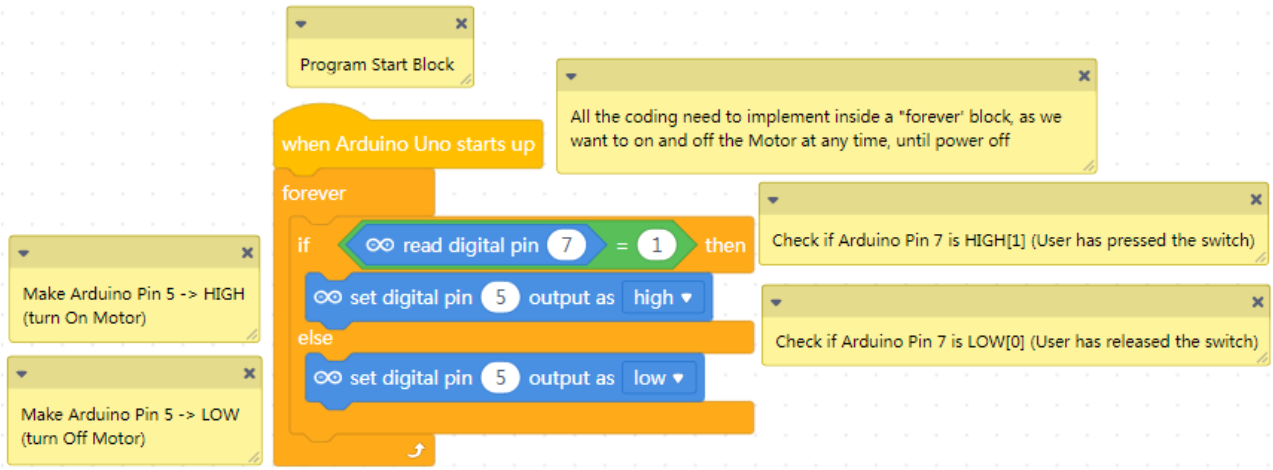
Caution: Do not close your hands to motor while doing this project.

Arduino Uno pin connections

Motor -> Arduino Uno Pin D5

Push Button Switch -> Arduino Uno Pin D7

mBlock Scratch Program Code



Here using the **"read digital pin 7"** instruction block, we read the status of (HIGH[1] or LOW[0]) Arduino Pin 7. When press the switch, 5V connects to the Arduino Pin 7, hence pin 7 state gets "HIGH" (HIGH = 1). When release the switch, 0V connects to the Arduino Pin 7, hence Pin 7 state gets "LOW" (LOW = 0). In this way, programme can understand the status of switch, so we can do whatever we want, according to this status of the switch. Here we on and off a Motor, according to the status of the switch. (Status of the Switch = Press or Release)

Note : For more details on mBlock Scratch programming, first read Chapter 5 of the tutorial document. (**Chapter 5 – mBlock Software**)

Project 9– Obstacle Detector using IR Sensor

Project Details

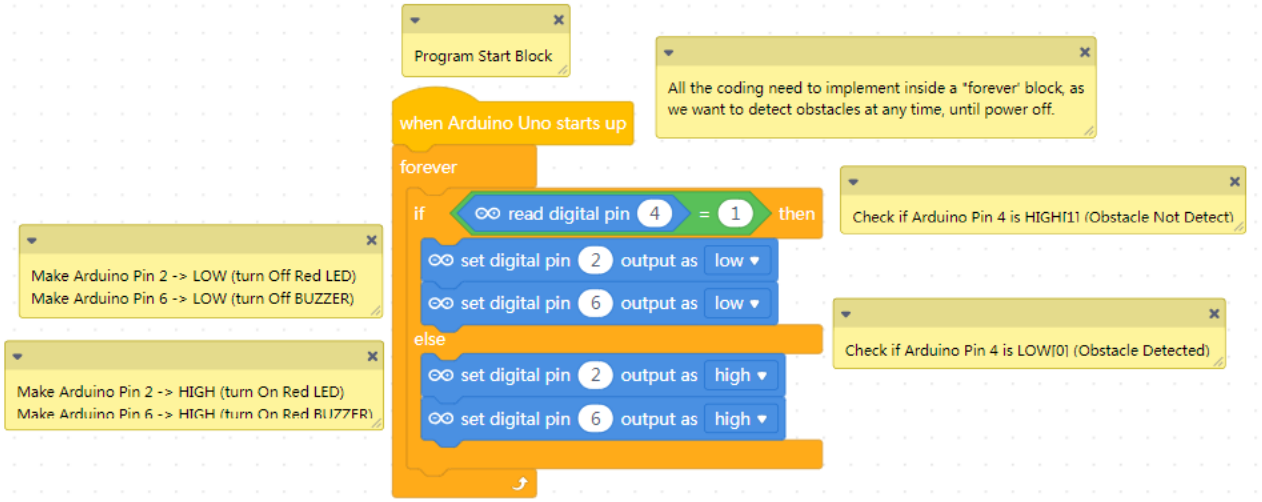
In this project we detect obstacles using the IR (Infrared) Sensor. If detect an obstacle, Red LED should turns on and buzzer should starts ringing. Then we know that we found an obstacle. Test the program by bringing an obstacle close to the IR sensor and move back.

Arduino Uno pin connections

Red LED -> Arduino Uno Pin D2

IR Sensor Data Pin -> Arduino Uno Pin D4

mBlock Scratch Program Code



IR (Infrared) Sensor in MPatrol 1 learning kit has 3 pins. They are Data, Power and Ground. Data pin has connected to Arduino Uno Pin 4. When there is no obstacle in front of IR Sensor, Data pin is in HIGH[0] state. That means voltage of Data pin is 5V. When there is an obstacle in front of IR Sensor, Data pin goes to LOW[0] state. That means voltage of Data pin is 0V. Using the **"read digital pin"** code instruction block, we read the state of the Data pin. (Wether it is HIGH or LOW). So if the Arduino Uno pin is in LOW state (LOW = 0) (Obstacle detected), then we turn on the Red LED and turn on the Buzzer. If the Arduino Uno pin is in HIGH state (HIIGH = 1) (no obstacle detected), we turn off the Red LED and Buzzer.We use **"If else"** Instruction block to handle this logic. Sensing distance to an obstacle can be adjusted by rotating the blue variable resistor in IR Sensor.

Note : For more details on mBlock Scratch programming, first read Chapter 5 of the tutorial document. (**Chapter 5 – mBlock Software)**

Project 10– Turn on a LED using a Touch Switch

Project Details

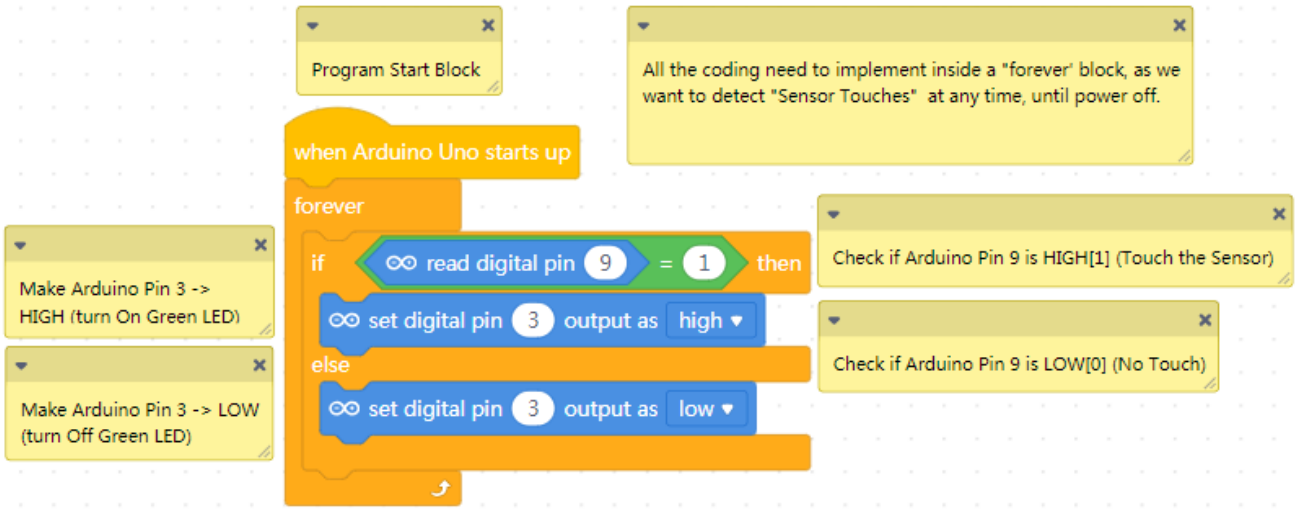
In this project we are making a Touch Switch based light on/off circuit. We are going to turn on the Green LED in MPatrol 1 Learning Kit, just touching a switch plate. For sense the touch, we use a Touch Sensor. So we have to program the MPatrol 1 Learning Kit as a Touch Switch. Once complete the programming, Green LED should turn on when you touch the Circular area of the Touch Sensor and should turn off, once you release00 the Touch Sensor.

Arduino Uno pin connections

Green LED -> Arduino Uno Pin D3

Touch Sensor Data Pin -> Arduino Uno Pin D9

mBlock Scratch Program Code



Touch Sensor in MPatrol 1 learning kit has 3 pins. They are Data, Power and Ground. Data pin has connected to Arduino Uno Pin 9. When you touch the circular area of Touch Sensor, Data pin is going into HIGH[1] state. That means voltage of Data pin is 5V. When you release your hand from sensor, Data pin goes back to LOW[0] state. That means voltage of Data pin is 0V. Using the **"read digital pin"** code instruction block, we read the state of the Data pin. (Wether it is HIGH or LOW). So if the Arduino Uno pin 9 is in HIGH state (HIIGH = 1) (Touch the Sensor), then we turn on the Green LED. If the Arduino Uno pin 9 is in LOW state (LOW = 0) (Release hand from sensor), we turn off the GreenLED. We use **"If else"** Instruction block to handle this logic. This is how we programme a touch switch.

Note : For more details on mBlock Scratch programming, first read Chapter 5 of the tutorial document. (**Chapter 5 – mBlock Software)**

Project 11– Turn on LEDs using a Sound Switch

Project Details

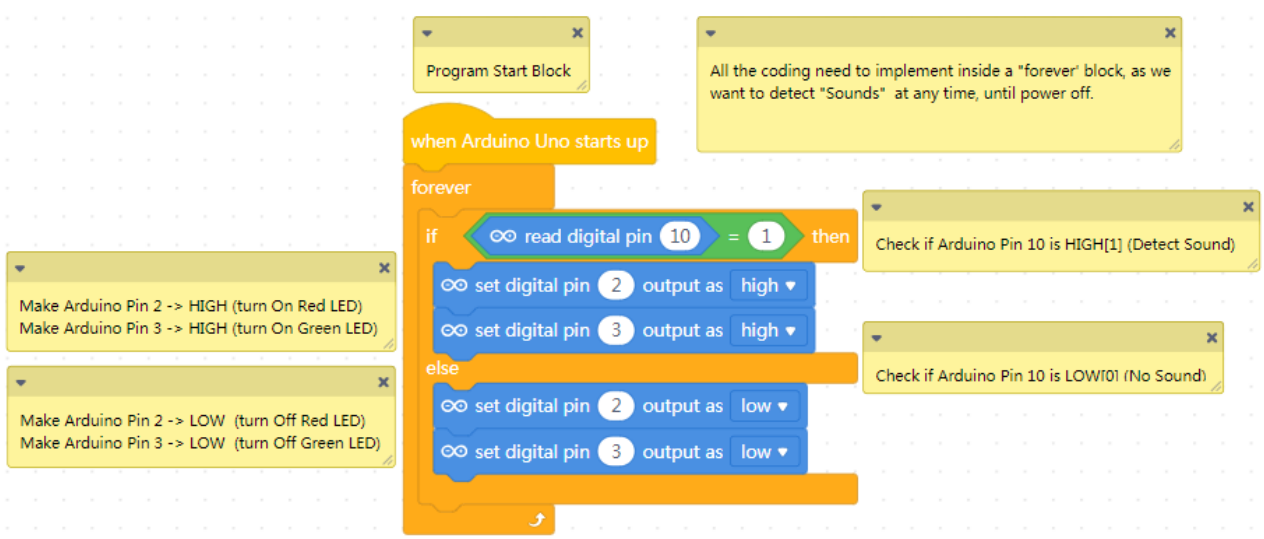
In this project we are making a Sound Switch based light on/off circuit. We are going to turn on the Red and Green LEDs in MPatrol 1 Learning Kit using a Sound command. For sense the sound, we use a Sound Sensor. So we have to program the MPatrol 1 Learning Kit as a Sound Switch. Once complete the programming, Red and Green LEDs should turn on when you clap or when you generate any sound.

Arduino Uno pin connections

RED LED -> Arduino Uno Pin D2, Green LED -> Arduino Uno Pin D3

Sound Sensor Digital Data Pin -> Arduino Uno Pin D10

mBlock Scratch Program Code



Sound Sensor in MPatrol 1 learning kit has 3 pins. They are Data, Power and Ground. Data pin has connected to Arduino Uno Pin 10. When you generate a sound by clapping or hit on the table, Data pin is going into HIGH[1] state. That means voltage of Data pin is 5V. When there is no sound, Data pin goes back to LOW[0] state. That means voltage of Data pin is 0V. Using the **"read digital pin"** code instruction block, we read the state of the Data pin. (Wether it is HIGH or LOW). So if the Arduino Uno pin 10 is in HIGH state (HIIGH = 1) (Sound detected), then we turn on the Red LED and Green LEDs. If the Arduino Uno pin 10 is in LOW state (LOW = 0) (No sound detected), we turn off the Red and Green LEDs. We use **"If else"** Instruction block to handle this logic. Sound sensitivity can be adjusted by rotating the blue variable resistor in Sound Sensor. This is how we programme a sound switch.

Project 12– Home Security System

Project Details

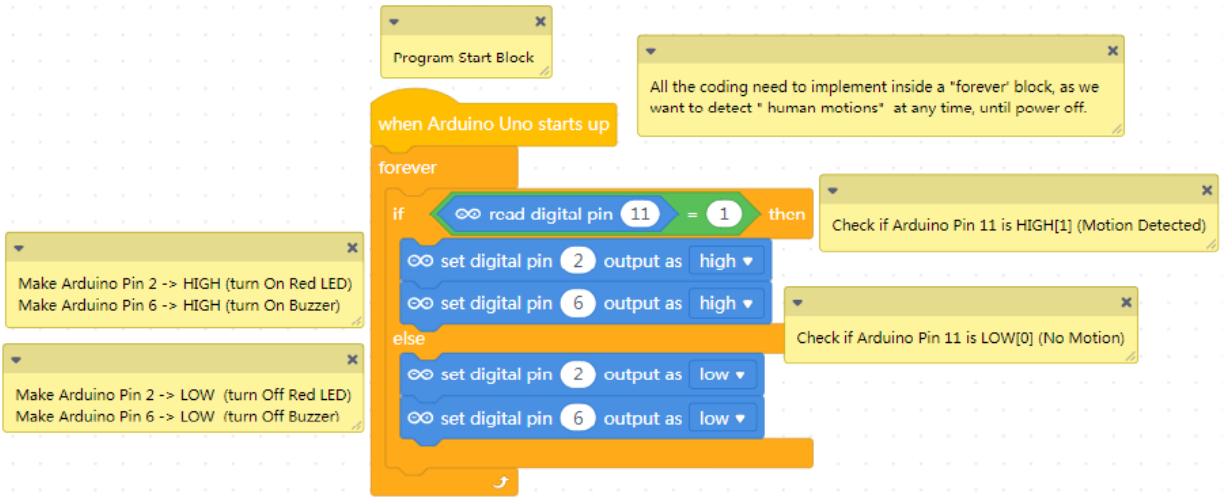
In this project we are making a Home Security System. You can turn on this system in night, so if someone come to your home in night, circuit turn on a bulb and also ring an alarm. So the circuit works as an automated security system. Now we will see how to make this home security system, using the PIR Motion Sensor, Red LED and Buzzer in MPatrol 1 Learning Kit. PIR Motion Sensor can detect human motions. So we can write a program to turn on the Buzzer and LED, when the Motion Sensor detects any human motion.

Arduino Uno pin connections

RED LED -> Arduino Uno Pin D2, Buzzer -> Arduino Uno Pin D6

Motion Sensor Data Pin -> Arduino Uno Pin D11

mBlock Scratch Program Code



PIR Motion Sensor in MPatrol 1 learning kit has 3 pins. They are Data, Power and Ground. Data pin has connected to Arduino Uno Pin 11. When Motion sensor detects a human motion, Data pin is going into HIGH[1] state. That means voltage of Data pin is 5V. When there is no motion, Data pin goes back to LOW[0] state. That means voltage of Data pin is 0V. Using the **"read digital pin"** code instruction block, we read the state of the Data pin. (Wether it is HIGH or LOW). So if the Arduino Uno pin 11 is in HIGH state (HIIGH = 1) (Motion detected), then we turn on the Red LED and Buzzer. If the Arduino Uno pin 11 is in LOW state (LOW = 0) (No motion detected), we turn off the Red LED and Buzzer. We use **"If else"** Instruction block to handle this logic. Now walk towards the Motion Sensor. When you close to the sensor, you can hear the Buzzer sound and can see the Red LED turns on. When you go away from the Motion Sensor, both Red LED and Buzzer is turning off.

Project 13– Electronic Organ

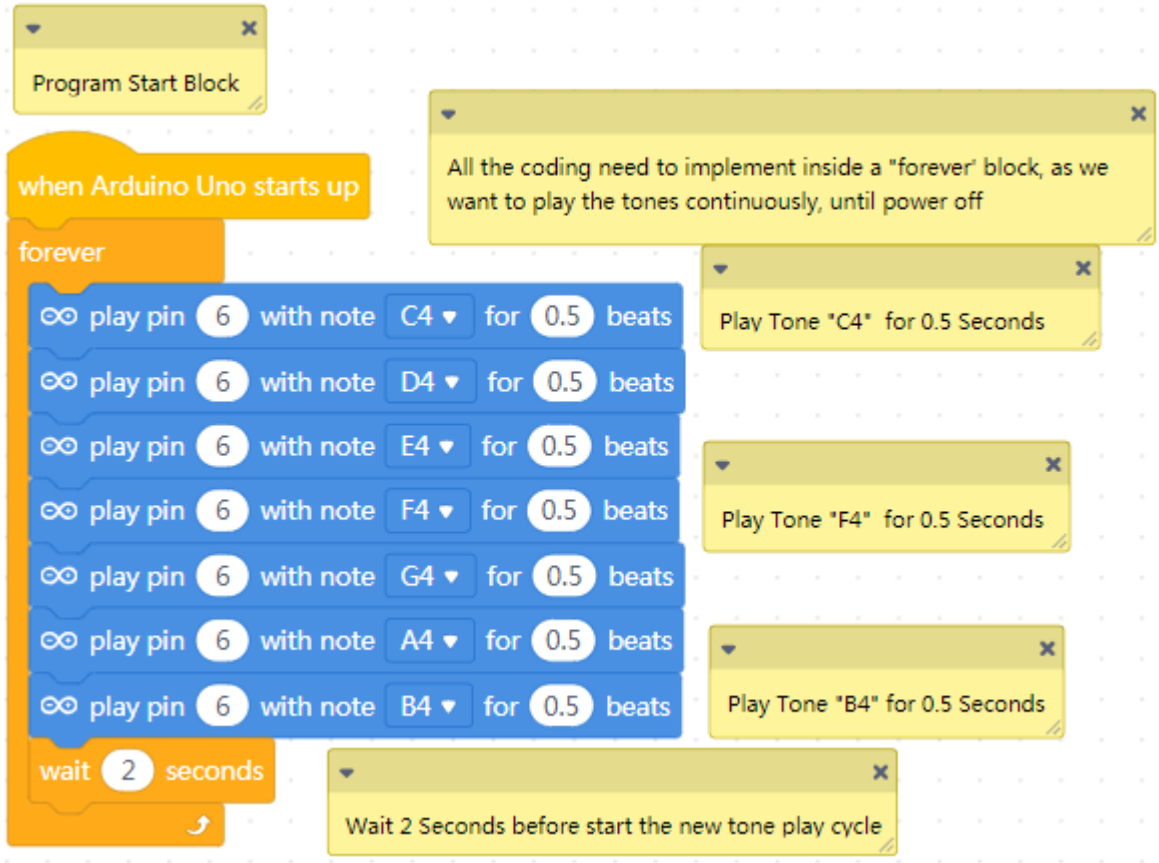
Project Details

In this project, we are learning how to make an Electronic Organ using MPatrol 1 Learning Kit. Electronic Organs generate different sounds, when you press the keys. These sounds are called music notes. Frequency of these notes is different from one note to another note. That is the reason we can here different sounds, when we press different keys in an organ. We can program an Arduino Uno pin to output different frequency notes. That means we can output different frequency waves from an Arduino Uno pin. If we connect that Arduino Uno pin to a buzzer, we can hear the music note output from that particular pin. So in this project, we make an organ by output different frequency notes from Arduino Uno Pin D6 (Pin D6 has connected to the Buzzer) using the mBlock programming instruction blocks.

Arduino Uno pin connections

Buzzer -> Arduino Uno Pin D6

mBlock Scratch Program Code



Here we have used **"play pin _ with note _ for _ beats"** instruction block for generate different music notes. As you can see there are 3 settings need to set in this block.

(1)**Arduino Uno Pin Number**–This is the Arduino Uno pin number, which connect to the Buzzer. In MPatrol 1, this is pin number 6.

(2)**note** – This is the note we want to output from the Arduino pin. There are so many different notes in music, which generate different sounds in an organ. Eg. C4, D4, E5, F6..etc. In our program we generate C4, D4,E4,F4,G4,A4,B4 notes. You can change the programme to generate different notes as you want.

(3)**beats** - This is the time in seconds, which the selected note want to play. In our programme we play each note for 0.5 seconds. In an actual organ, this is the time you press a particular key. You can change this value and see how tones are generating.

So in this programme we play C4, D4,E4,F4,G4,A4,B4 notes continuously and each note play for 0.5 seconds. At the end of playing these 7 notes, we wait 2 seconds and start the note play cycle again. You can change the notes as well as note play time and see how your electric organ works for different music notes. If you know the notes for your favourite songs, you can program the MPatrol 1 to play those songs.

Project 14– Alarm Timer

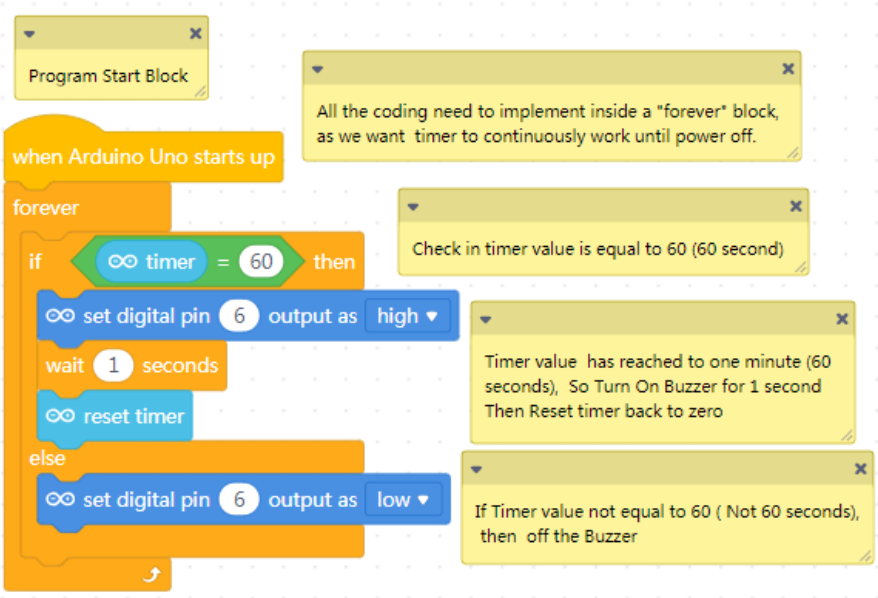
Project Details

In this project we are learning about Timers. Timers are very important unit in coding and electronics. Using timers we can do certain tasks in pre-defined intervals. If you want to do some task like turn on a bulb, ring a buzzer or reading data from a sensor in certain time interval (eg. every 30 second, every minute, every hour ...etc), we can use timers. Using the **"timer"** instruction block in mBlock software, we can implement timers in our programme. In this project, we ring a Buzzer for every one minute.

Arduino Uno pin connections

Buzzer -> Arduino Uno Pin D6

mBlock Scratch Program Code



"timer" instruction block start counting from zero and increment it's counter value by one for each 1 second. In our programme, when timer counter value equal to 60 (that mean 60 seconds or 1 minute), we turn on the Buzzer for one second. Then using the **"reset timer"** instruction block, we reset the timer. When reset the timer, timer counter value set back to zero. So it again starts counting from 0 to 60. As we have implemented all the coding inside a **"forever"** instruction block, this process repeat again and again until power off. So we can hear the Buzzer sound for every one minute. (every 60 second). Like this way, using timers, you can repeat any task in a pre-defined timer interval. (eg. reading data from a sensor for each 30 second). You can change the timer value for different figures and see how timer working for different timer intervals.

Project 15– Automatic Night Bulb

Project Details

In this project we are going to make a Night Bulb, which automatically turn on when the environment is dark. You can see this kind of Bulbs in Streets, which automatically turn on in the night and turn off in the morning. You can fix same kind of Bulb in front of your house, which automatically turn on in the night and automatically turn off in the morning.

Now let see how we can do this project. MPatrol 1 has a special resistor called LDR. (Light Dependent Resistor). We use this LDR to control the night bulb. LDR is a special kind of resistor, which resistance change according the light on it. Resistance of a LDR decreases, with increase in light intensity on it and vice versa. So according to the Ohm's Law (V=IR), Voltage across the LDR also vary according to the light intensity on it. In MPatrol 1 Learning Kit, we have connected this LDR to one of the Analog Input (Analog 1) of Arduino Uno board. So by reading the Analog Voltage value across the LDR (Using Analog 1 Input), we can decide whether the environment is dark or not. Then simply we can on or off a light as we done in previous projects.

What is a Analog Input?

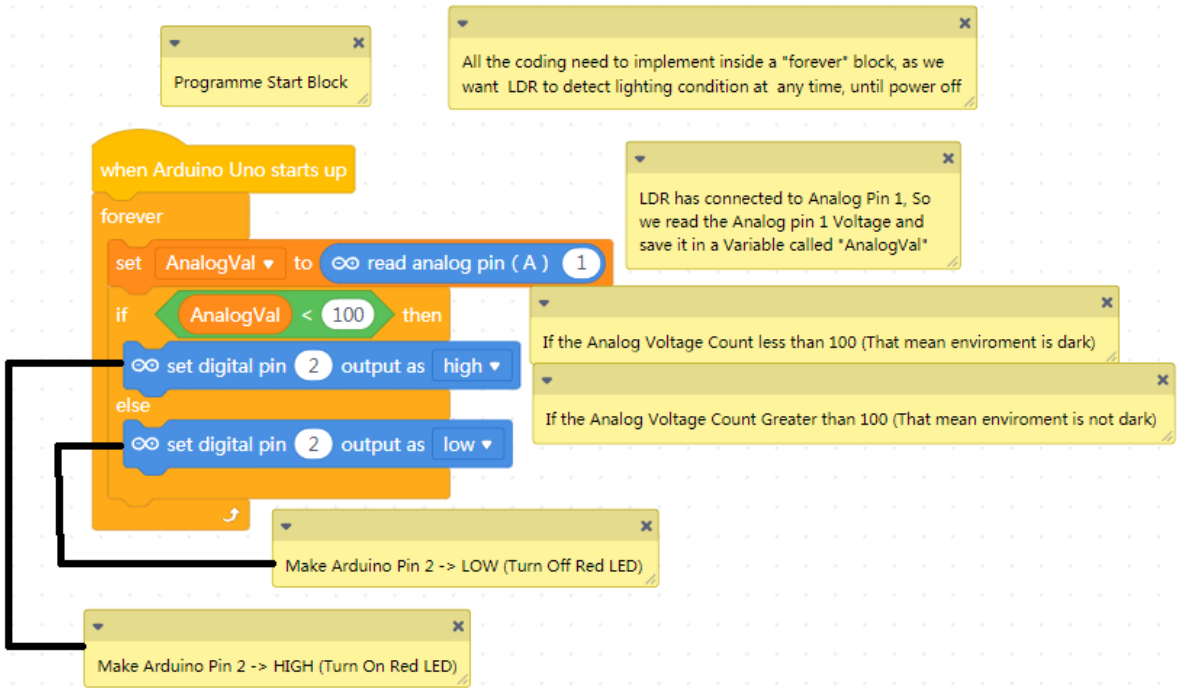
So far, all the inputs in Arduino Uno board we learned in previous projects are Digital Inputs. Arduino Uno board has 13 Digital Inputs. Digital Input has only two values at any particular time. They are HIGH or LOW. High means 5V and LOW means 0V. But an Analog Input can have any value between 0V to 5V at any particular time. As an example Analog 1 Input of the Arduino Uno board can have a voltage value of 0V, 0.5V, 1.5V, 2.75V, 3.34V, 4.58V, 5V..etc at any particular time. Microcontroller inside the Arduino Uno board (this is a mini computer, which we download our code) has a special unit called Analog to Digital Converter. This Analog to Digital Converter, converts the Analog Voltage values in to numeric values. For example Analog Voltage of range 0V to 5V, is being converted into numeric values of range 0 to 1023. Using the block instructions, we can read this numeric value for a particular analog voltage, in our programme. This is the way we read an analog voltage value when do coding. So again using the block instructions, we can do whatever we want (eg. On or Off a light), according to this Analog Voltage Value. Arduino Uno board has 6 Analog Inputs.

Note : Read the "Chapter 4 – Arduino Development Board" for more details on Arduino Uno board and Analog Inputs.

Arduino Uno pin connections

Light Dependent Resistor (LDR) -> Arduino Uno Pin A1 (Analog 1)

mBlock Scratch Program Code



"read analog pin (A) 1"instruction block read the Analog Voltage on Analog pin 1 of Arduino Uno board. It actually read the related numeric value for the applied Analog Voltage. (Value range of 0 to 1023, 0V = 0, 5V = 1023) Using **"set AnalogVal to"**instruction, we save the Analog 1 Voltage value to "AnalogVal" variable. "AnalogVal" is a variable, which can store any value. Before use this variable, first we have to create this variable from **"Variables**" instruction block category. You can give any name to this variable. Here I have given the name "AnalogVal".

By experiment I found that when the environment is dark, Analog 1 Voltage numeric value goes below 100. So using **"if then else"**instruction block, I turn on Red LED, if the Analog 1 voltage numeric value is less than 100. That is when environment is dark. Further if the Analog 1 voltage numeric value is greater than or equal 100, I turn off the Red LED. (When environment is not dark). This is how we control a Bulb according to the light level of the environment. You can change this voltage threshold value (In my code, it is 100) and adjust your Automatic Night Bulb for different lighting conditions.

Note : For more details on Variables and How to create them, read Chapter 2 of the tutorial document. (**Chapter 2– Introduction to Computer Programming)**

Project 16– Light Dimmer

Project Details

In this project, we are making a Light Dimmer circuit. Light Dimmer means we can control the brightness of the light as we want. This is very useful circuit in night time as we want to reduce the brightness of our bed lamp (dim the light) when we are sleeping. We use Green LED and Variable Resistor of MPatrol 1 Learning kit for make this circuit. After completing the project, by rotating the knob of variable resistor, we can control the brightness of the Green LED. Now will see how we can do this. We can do this brightness control using a technology called Pulse Width Modulation. (PWM)

What is Pulse Width Modulation (PWM) ?

PWM is very popular application in electronics. We use PWM lot in motor controlling for robots. Pulse width modulation (PWM) is a technology for performing digital encoding on analog signals. Here instead output a fixed voltage, we output a voltage pulse stream from Arduino Uno pins. It involves two key parameters, namely frequency and duty cycle. The frequency determines the time required for completing a single cycle and the rate of signals changing from high to low level. The duty cycle determines the time the signals stay in high level within the total period of time. By changing the duty cycle of PWM, we can change the average voltage of the output signals, thus control the power of the applied device. Some of the pins in Arduino Uno have this PWM output feature. Pin 3 of Arduino Uno board can out put a PWM pulse stream. We have connected our Green LED to Pin 3 of Arduino Uno board, so we can control the brightness of the Green LED using the PWM technology. Value of the Duty Cycle is range from 0 to 255, where 0 indicates 0% duty value and 255 indicates 100% duty value. Higher the duty cycle values means higher the output average voltage. That means higher the brightness of the LED.

In above paragraph, we understood that for control the brightness of the Green LED, we have to change the duty cycle value of pin 3 PWM output signal. Now will see how we can do this. In this project we use a variable resistor for change the duty cycle value of PWM output. Variable Resistor is a special kind of resistor, which resistance change, when rotating its knob. According to the Ohm's Law (Voltage = Current * Resistance), voltage across the resistor change when changes its resistance. (Voltage changes from 0V to 5V). When change the voltage, respective analogvoltage numeric count change from 0 to 1023. (0V = 0, 5V = 1023) (For more details about Analog Voltage Inputs, first read the project 15 – Automatic Night Bulb). We have connected variable resistor to Analog 0 Input of Arduino Uno board. So when rotates the variable resistor, we can read the variable analog voltage numeric count, using block instructions in our programme. In our programme, we assign this Analog 0 numeric count value for the duty cycle value of Pin 3 PWM output. But we cannot directly assign the Analog 0 numeric count value for the PWM duty cycle value. Because Analog 0 numeric count value has a range of 0 to 1023, but duty cycle value only accepts the values in the range of 0 to 255. Because of this issue, we have to scale down the Analog 0 numeric count values dividing by 4, before assign the value for the Pin 3 Duty Cycle.

So **Pin 3 Duty Cycle Value = Analog Input 0 Numeric Count Value/ 4.** This is the method we have used in this project for control the brightness of the LED. Now we will see how we implement this method using mBlock instruction blocks.

Arduino Uno pin connections

Green LED -> Arduino Uno Pin D3

Variable Resistor -> Arduino Uno Pin A0

mBlock Scratch Program Code

In **"set PulseWidth to read analog pin (A) 0"**instruction, we read the analog input 0 voltage numeric count value into a variable called "PulseWidth". So "PulseWidth" variable gets values, in the range from 0 to 1023, when rotating the knob of variable resistor. In **"set PWM 3 output as PulseWidth / 4"** instruction, we divide the PulseWidth variable value by 4 and assign the final value for the Pin 3 PWM Duty Cycle value. So when rotating the knob of variable resistor, Analog 0 numeric count value change from 0 to 1023 and Pin3 PWM Duty Cycle value change from 0 to 255. When change the PWM Duty Cycle value, Average Voltage output from Arduino Uno Pin 3 also change and this control the brightness of the Green LED. (Higher the Average Voltage means higher the brightness and vice versa)

Project 17– Motor Speed Controller

Project Details

In this project, we make a Motor Speed Controller. Motor speed controlling is very important in electronics, especially when we make robot vehicles. Motor speed controlling needs for your robot vehicle to move in different speeds and to move along different paths. In your house, you may need motor speed controlling for vary the speed of your fan or vary the water pressure of your garden water pump. Now we will see how we can do the motor speed controlling. Actually technology behind this is same as what we done for the Light Dimmer Project. So please first do the **Project 15 – Light Dimmer** before doing this project. As I have explained all the theoretical information indetail in Project 15, I am not going to explain them in detail in this project.

In Light Dimmer project, we controlled the brightness of a LED by rotating the knob of a variable resistor. We are doing the same thing here. We are going to control the Motor of MPatrol 1 Learning Kit, by rotating the knob of the variable resistor. As explained in Light Dimmer project, when rotating the knob of variable resistor, Analog 0 Input voltage change from 0V to 5V. Then the respective voltage numeric count value change from 0 to 1023. We divide this Analog 0 numeric count value by 4 and then apply the final value to Pin 5 PWM duty cycle value.Motor has connected to the Pin 5 of Arduino Uno board. We have to divide the Analog 0 numeric count value by 4, because PWM Duty Cycle value only accept the values in the range of 0 to 255.

(Pin 5 Duty Cycle Value = Analog Input 0 Numeric Count Value / 4)

So when rotating the knob of variable resistor, Analog 0 numeric count vary in the range of 0 to 1023 and Pin 5 Duty Cycle vary in the range of 0 to 255. When change the Duty Cycle value of Pin 5, Average Voltage value output from Pin 5 also change. Higher the Duty Cycle means higher the Average Voltage Output and Vice Versa. Higher the Average Voltage, higher the speed of the Motor and Vice Versa. This is how we can control a Motor using a Variable Resistor.

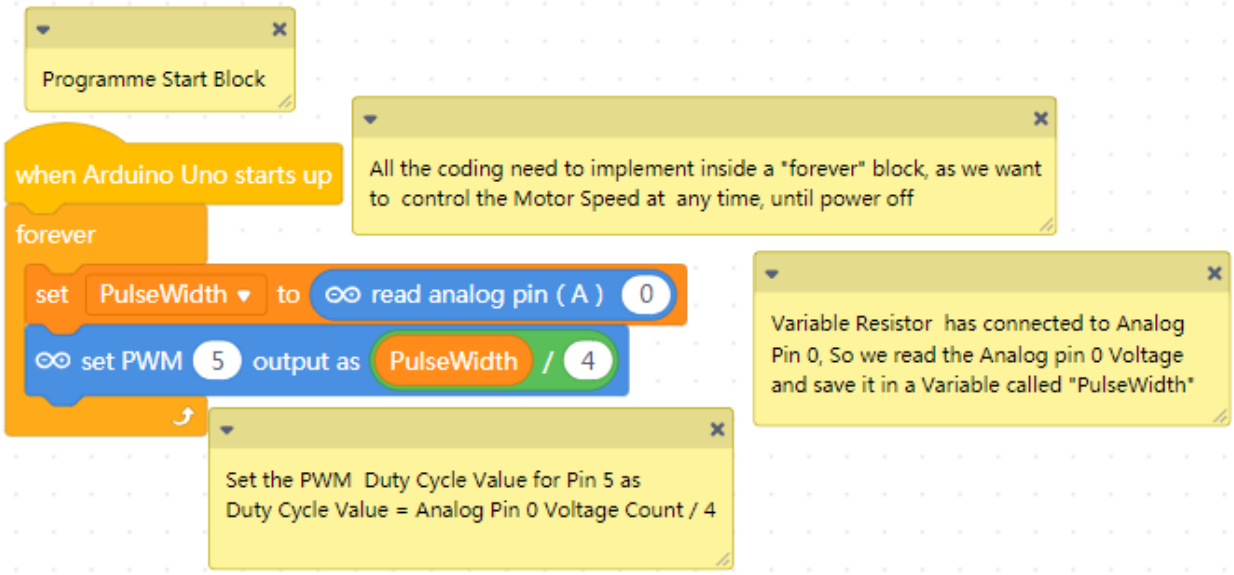
Caution: Do not close your hands to motor while doing this project.

Arduino Uno pin connections

Motor -> Arduino Uno Pin D5

Variable Resistor -> Arduino Uno Pin A0

mBlock Scratch Program Code



In **"set PulseWidth to read analog pin (A) 0"**instruction, we read the analog input 0 voltage numeric count value into a variable called "PulseWidth". So "PulseWidth" variable gets values, in the range from 0 to 1023, when rotating the knob of variable resistor. In **"set PWM 5 output as PulseWidth / 4"** instruction, we divide the PulseWidth variable value by 4 and assign the final value for the Pin 5 PWM Duty Cycle value. So when rotating the knob of variable resistor, Analog 0 numeric count value change from 0 to 1023 and Pin 5 PWM Duty Cycle value change from 0 to 255. When change the PWM Duty Cycle value, Average Voltage output from Arduino Uno Pin 5 also change and this control the speed of the Motor. (Higher the Average Voltage means higher the Motor Speed and vice versa)

Project 18– Electronics Distance Meter

Project Details

In this project, we make an Electronics Distance Meter for measure the distance from MPatrol 1 Learning kit to an object. Normally we measure the distance using a ruler. Now we will see how to measure the distance electronically. For this we use the ultrasonic sensor in MPatrol 1 Learning Kit. In this project we measure the distance to an object using the Ultrasonic Sensor and display the distance value in cm in LCD display. First will learn how to measure the distance using the ultrasonic sensor.

Ultrasonic Sensor

An ultrasonic sensor is used to detect the distance between the sensor and an obstacle. The detection distance ranges from 2cm to 400 cm. Ultrasonic sensors are applicable to automatic obstacle avoiding or other ranging projects. An ultrasonic sensor includes an ultrasonic transmitter and receiver. The timing begins when the transmitter transmits an ultrasonic wave in a certain direction and stops when the receiver receives the ultrasonic wave reflected by an obstacle. Let say an ultrasonic wave took time "t" seconds to get back to the receiver. The transmission speed of ultrasonic waves in air is 340 m/s. Therefore we can calculate the distance between the ultrasonic sensor and obstacle, S = 340 * t/2.(Speed = Distance / Time). You do not need to worry about these calculations in this project. mBlock Ultrasonic Sensor block instructions do all these calculations and return us the distance value in cm. Then we display this distance value in LCD display.

I2C 16 * 2 Liquid Crystal Display (LCD)

In MPatrol 1 Learning Kit, we have used I2C Liquid Crystal Display. I2C is the name of communication method (protocol) between LCD and Arduino Uno board. It means I2C is the language which LCD is speaking with Arduino Uno board for display data. Advantage of I2C protocol is, it use only two wires for communication with LCD. There are other types of LCDs such like parallel LCDs, which need 8 wires for communication with Arduino Uno board. So by using an I2C LCD, we can save lots of pins in Arduino Uno board for connect with other devices. In I2C LCD, two communication lines are called SDA and SCL. SDA is the data signal line and SCL is the clock signal line.16 * 2 means, this LCD has 16 columns and 2 rows. So we can display maximum of 32 (16 * 2) characters in this LCD. LCDs in the market havedifferent column and row numbers.

Adding the I2C LCD extension

I2C Block instruction is not a default block instruction in mBlock software. So you have to add a thirdparty I2C LCD block instruction library from the mBlockExtension Centre. For add extensions, read Chapter 5 of the tutorial document. (**Chapter 5 – mBlock Software)**. You can do a search as "lcd i2c" for quickly find a suitable extension. There are several third party I2C LCD extensions in the extension centre. I have noticed some I2C LCD extensions are not working correctly. You can add one of the I2C LCD extension and see whether it is working or not. If not working, you can remove that extension from the extension centre

and add another one. Make sure to select an English language extension, because there are lots of other language extensions also in the extension centre. In this project, I have used the below mention I2C LCD block instruction extension.

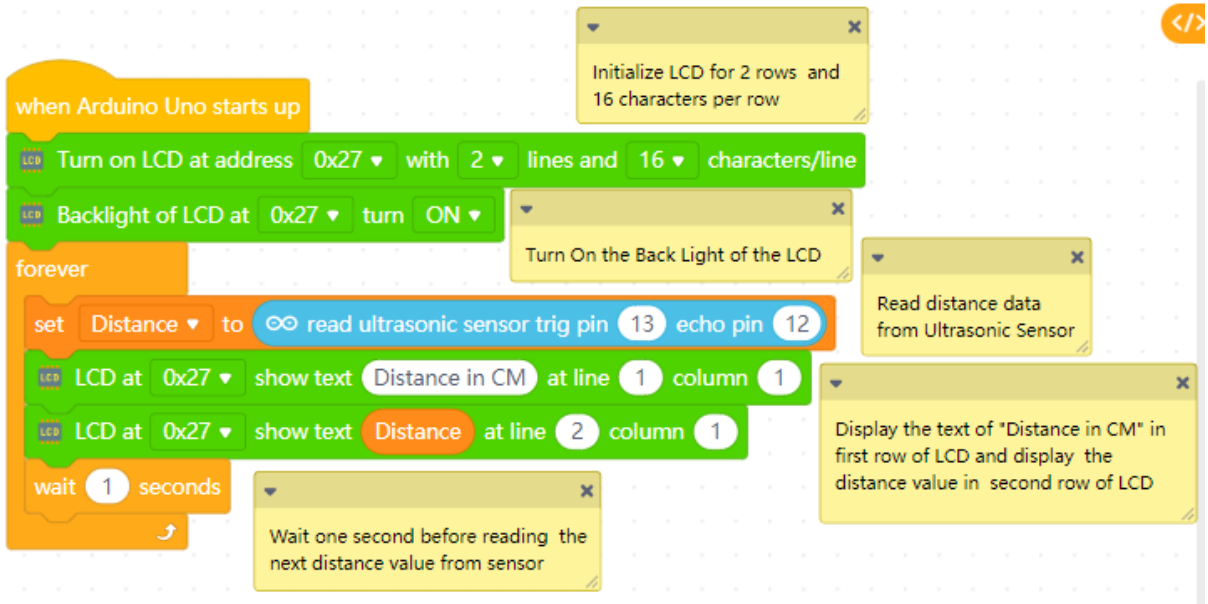
Arduino Uno pin connections

Ultrasonic Sensor Trigger -> Arduino Uno Pin D13

Ultrasonic Sensor Echo -> Arduino Uno Pin D12

LCD SDA -> Arduino Uno Pin SDA, LCD SDL -> Arduino Uno Pin SC

mBlock Scratch Program Code



This is the mBlock Scratch code for ultrasonic sensor. Now will try to understand this code. Here in **"Turn on LCD at address 0x27 with 2 lines and 16 characters/line"** block instruction, we initialize the LCD Display. For this block instruction, we have to set 3 values. First one is the I2C Bus address. In I2C protocol, we can connect several I2C devices for same SDA and SCL wires. Then we call this as a device bus. Each device in the bus should be able to

uniquely identify, otherwise we cannot communicate with all the devices using two wires. So we need a unique address for each device in the I2C bus. In our project LCD is the only one device in the I2C bus, so the LCD bus address is 0x27. Next set value is number of rows in LCD. In our LCD we have 2 rows. Final set value is number of columns in LCD. In our LCD we have 16 columns.

Next we are calling **"Backlight of LCD at 0x27 turn ON"** block instruction. Using this instruction we can turn On the LCD backlight. 0x27 is the LCD bus address which explained in the above paragraph. We can use the same block instruction for turn Off the LCD. Without the backlight, you cannot see anything on a LCD at night.

Next we read the distance from MPatrol 1 Learning Kit to an obstacle using **"read ultrasonic sensor trig pin 13 echo pin 12"**block instruction. This block instruction can be found under the **"Sensor"** block instruction category and it returns the distance to an obstacle in cm. Ultrasonic Sensor has 2 pins called Trigger and Echo. Echo pin use to transmit and receive the Ultrasonic Wave and Trigger pin use to measure the time. Trigger pin of the ultrasonic sensor has connected to Pin 13 of Arduino Uno board and Echo pin has connected to Pin 12 of Arduino Uno board. So we have to set 13 and 12 as the trig pin and echo pin values. Here first we have created a variable called "Distance" and then have assigned the distance value to this variable, which return from above block instruction.

Finally we display the distance value in LCD using "show text" block instruction. Using **"LCD at 0x27 show text Distance in CM at line 1 column 1"** block instruction, we write the text of "Distance in CM", to the LCD, starting from row1 , column 1 position. (i.e. in the first row). Next using **"LCD at 0x27 show text Distance at line 2 column 1",** we write the "Distance" variable value to the LCD, starting from row2 , column 1 position. (i.e. in the second row). At the end of the code, we wait1 second, before reading the new distance value from sensor. Here the wait delay is important, otherwise if we continuously write data to the LCD without any delay, they may not readable.

As we have implemented all these coding inside the "forever" block instruction, code repeat again and again, so we can measure the distance to an obstacle at any time until power off. This LCD block instruction library has other block instructions such like Clear LCD (which can use to erase the text on LCD display), Scroll LCD (move the text left or right on LCD) ..etc. Experiment with these other block instructions and see how they work.

Project 19– Temperature & Humidity Meter

Project Details

In this project, we are making a Temperature and Humidity Meter using the MPatrol 1 Learning Kit. We use DHT11 sensor in MPatrol 1 Learning Kit for read the Temperature and Humidity readings and display them on the LCD. DHT11 is a temperature and humidity sensor, which have 3 pins. Data pin of the DHT11 sensor has connected to Arduino Uno Pin 8. So by sending commands to DHT11 sensor, using the Arduino Uno pin 8, we can read the temperature and humidity data from DHT11 sensor, using its data pin. Here we have to use two different commands for read temperature and humidity data. In Project 18 (Electronics Distance Meter) I explained about the LCD display and how to display text and data on it in detail. So I am not going to explain them again in this project. So please do Project 18 - Electronics Distance Meter Project, before do this project.

Adding the DHT11 extension

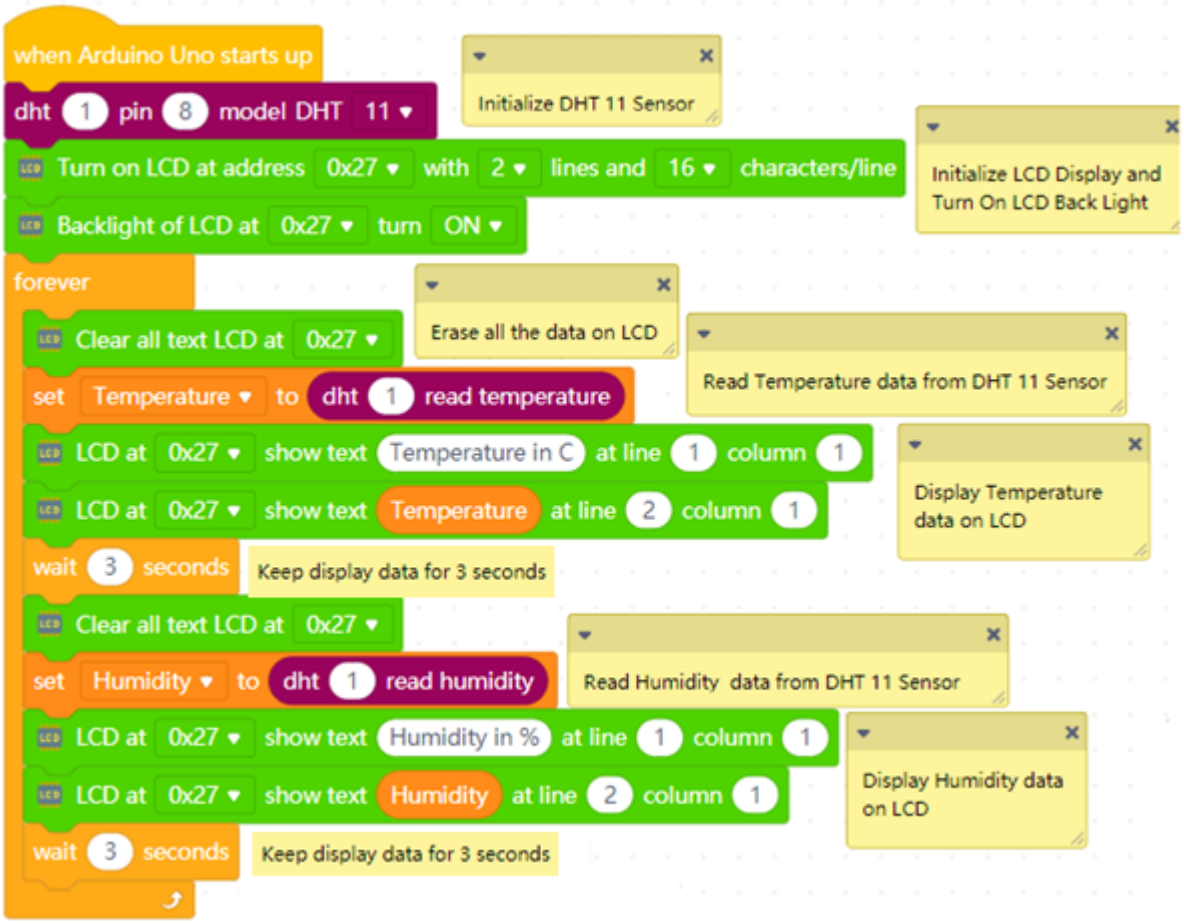
DHT Sensor Block instruction is not a default block instruction in mBlock software. So you have to add a third party DHT Sensor block instruction library from the mBlock Extension Centre. For add extensions, read Chapter 5 of the tutorial document. (**Chapter 5 – mBlock Software)**. You can do a search as “dht” for quickly find a suitable extension. There are several third party DHT Sensor extensions in the extension centre. SomeDHT Sensor extensions may not work correctly. You can add one of the DHT Sensor extension and see whether it is working or not. If not working, you can remove that extension from the extension centre and add another one. Make sure to select an English language extension, because there are lots of other language extensions also in the extension centre. In this project, I have used the below mention DHT Sensor block instruction extension.

Arduino Uno pin connections

DHT Sensor Data Pin -> Arduino Uno Pin D8

LCD SDA -> Arduino Uno Pin SDA, LCD SDL -> Arduino Uno Pin SC

mBlock Scratch Program Code



Let see how to implement the coding for our temperature and humidity meter. As usual stat the programme using the **"when Arduino Uno starts up"** block. Then use **"dht 1 pin 8 modle DHT 11"** block instruction for initialize the DHT 11 sensor. This block instruction needs 3 settings. First setting is always 1. Next setting is the Arduino pin number, which DHT sensor data pin is connected. In MPatrol 1, this is 8. Last setting is the DHT sensor type. There are three different DHT sensor types called DHT 11, DHT 21 and DHT 22. In MPatrol 1, we have used DHT 11 sensor. So, for the last setting, you have to select 11. To have these DHT block instructions, you must add the DHT sensor extension from the extension centre. Next we initialize the LCD module and turn on the back light as done in Project – 18.

We have to implement all the programming logic inside a **"forever"** block, as we need to display the temperature and humidity continuously on the LCD, until power off. Inside the "forever" block, first we clear all the text on the LCD. In this project, we have to display both Temperature and Humidity data on the same LCD, but our LCD can display only 32 characters at once. So before display the temperature data, we have to erase the humidity data and also before display the humidity data, we have to erase the temperature data. Using the **"Clear all text LCD at 0x27"** instruction block, we can erase all the data display on LCD.

Then make two variables called “Temperature” and “Humidity”. Using the **“dht 1 read temperature”** block instruction, read the environment temperature and save it in “Temperature” variable. Then display this temperature value on LCD. In the first raw of LCD, we display the text of “Temperature in C”. In the second raw of LCD, we display the value of “Temperature” variable. (As done in Project – 18). Using the **“wait 3 seconds” instruction** block, we keep display this temperature value for 3 seconds, before erase the LCD screen for display the humidity value.

Then we erase the LCD screen and read the humidity value using the **“dht1 read humidity”**instruction block and save it in “Humidity” variable. Same as previous step, then we display this humidity value on the LCD screen for 3 seconds. Finally, after 3 seconds, erase the LCD display again for display the next temperature data.

Project 20– Water Level Meter

Project Details

In this project we are making a Water Level Meter. Here we measure the water level in a container and display it on the LCD. We use the Water Sensor in MPatrol 1 for measuring the water level of the container. Water sensor is an Analog Sensor, which output an analog voltage according to the water level on the sensor. Water sensor has 3 pins called Analog Out, Power and Ground. Analog Out pin of the Water Sensor has connected to the Analog 2 pin of the Arduino Uno board. Analog voltage output from the Analog Out pin of water sensor is proportional to the water level and hence we can measure the level of water.

As explained in Project – 15, Analog to Digital Converter inside Arduino Uno board (inside the microcontroller), can covert this analog voltage into a numeric value, which represent the output analog voltage of the water sensor. So by displaying this analog numeric value on the LCD display, we can see a numeric value, which is proportional to the water level of the container. This display value should go up and down according to the water level. If you want, you can try to scale this analog numeric value into some physical units (eg. cm, mm), before display on LCD, using the mathematical instruction blocks in mBlock software.

If you are not familiar with the Analog Inputs, first do the "Project 15 – Automatic Night Bulb" project before doing this project. In that project, I have explained the Analog Inputs in detail.

Arduino Uno pin connections

Water Sensor Analog Out Pin (S) -> Arduino Uno Pin A2 (Analog 2)

LCD SDA -> Arduino Uno Pin SDA,

LCD SDL -> Arduino Uno Pin SC

Fixing the Water Sensor to MPatrol 1 Learning Kit

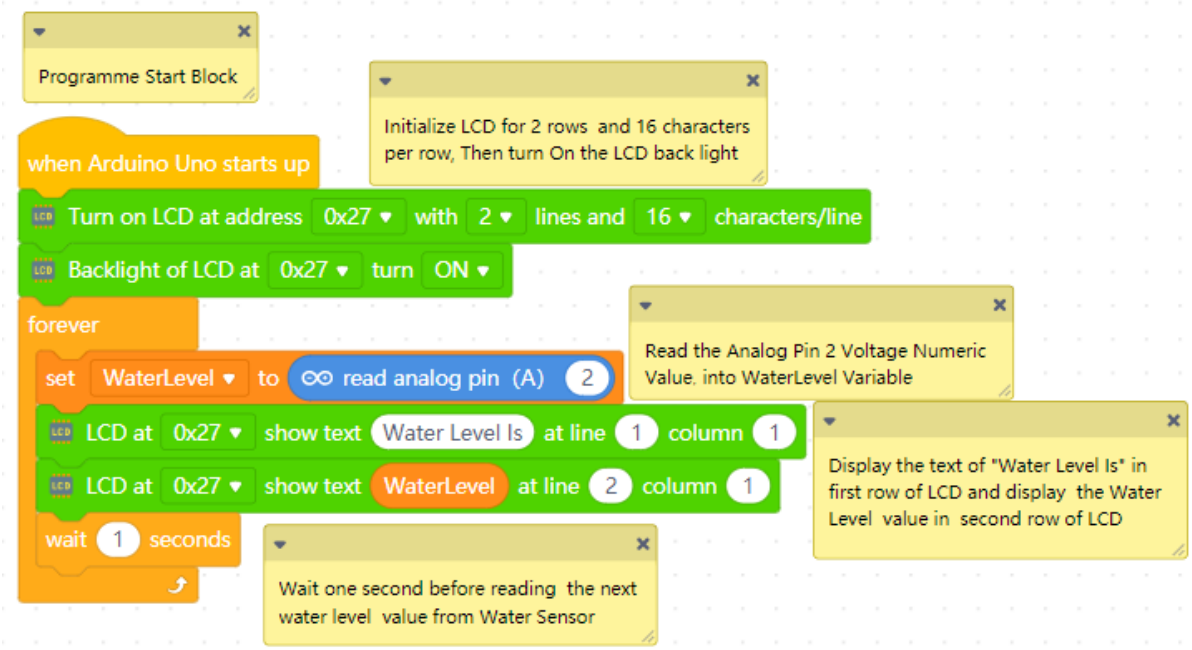
You have to connect the Water Sensor to MPatrol 1 Learning Kit, using the supplied 3 wire cable. Make the connections as mentioned below.

Water Sensor **"S"** Pin ->MPatrol 1 **OUT** Pin

Water Sensor **"+"** Pin ->MPatrol 1 **5V** Pin

Water Sensor **"-"** Pin ->MPatrol 1 **GND** Pin

mBlock Scratch Program Code



As done in the previous projects, first initialize the LCD display and turn On the LCD back light. Here also we must implement all the water sensor reading and display codes, inside a **"forever"** block. Because we want to read and display the water level continuously until power off. Using **"set WaterLevel to read analog pin (A) 2"**instruction, read the analog pin 2 voltage into "WaterLevel" variable. (We have connected the Analog Out pin of the water sensor to Analog Pin 2 of Arduino Uno board). Before using this instruction block, we have to create a variable called "WaterLevel". In the next step, display the water level value on the LCD screen. Here in the first row of LCD, we display the text of "Water Level Is". In the second row of LCD, we display the Analog 2 numerical value, which is proportional to the water level of the container. As mentioned earlier, you can scale this Analog 2 numerical value into some physical units (eg. mm, cm) before display on LCD, using the mathematical instruction blocks.Finally, we are waiting 1 second, before reading the next water level value from the water sensor. We need this delay, otherwise if we write data to LCD very fast, users cannot see them clearly.

Put the water sensor into a container and pour water little by little. Do not fill water over the marked area in the sensor and make sure not to touch any water in circuit components. Both these cases can make permanent damage to sensor. You can see the value on the LCD going up, when increasing the water level of the container. In practical situation, you can make this kind of circuit for control a water pump, according to the water level of a tank.

If you are not familiar with the LCD display related coding, first do the "Project 18 – Electronics Distance Meter" before doing this project. I have explained about the LCD display, adding LCD display extension and related codes in detail in that project.

Project 21– Moisture Level Meter

Project Details

In this project we are making a Moisture Level Meter. Here we measure the moisture level of soil and display it on the LCD. We use the Moisture Sensor in MPatrol 1 for measuring the moisture level of soil. Moisture sensor is an Analog Sensor same as water sensor we used in last project. It outputs an analog voltage according to the moisture level of soil. Moisture sensor has 3 pins called Analog Out, Power and Ground. Analog Out pin of the Moisture Sensor has connected to the Analog 3 pin of the Arduino Uno board. Analog voltage output from the Analog Out pin of MoistureSensor is proportional to the moisture levelof soil and hence we can measure the level of moisture amount in soil.

As explained in Project – 15, Analog to Digital Converter inside Arduino Uno board (inside the microcontroller), can covert this analog voltage into a numeric value, which represent the output analog voltage of the moisture sensor. So, by displaying this analog numeric value on the LCD display, we can see a numeric value, which is proportional to the moisture level of soil. This display value should go up and down according to the soil moisture level. If you want, you can try to scale this analog numeric value using a mathematical equation, before display on LCD, using the mathematical instruction blocks in mBlock software.

If you are not familiar with the Analog Inputs, first do the "Project 15 – Automatic Night Bulb" project before doing this project. In that project, I have explained the Analog Inputs in detail.

Arduino Uno pin connections

Moisture Sensor Analog Out Pin (AOUT) -> Arduino Uno Pin A3 (Analog 3)

LCD SDA -> Arduino Uno Pin SDA,

LCD SDL -> Arduino Uno Pin SC

Fixing the Moisture Sensor to MPatrol 1 Learning Kit

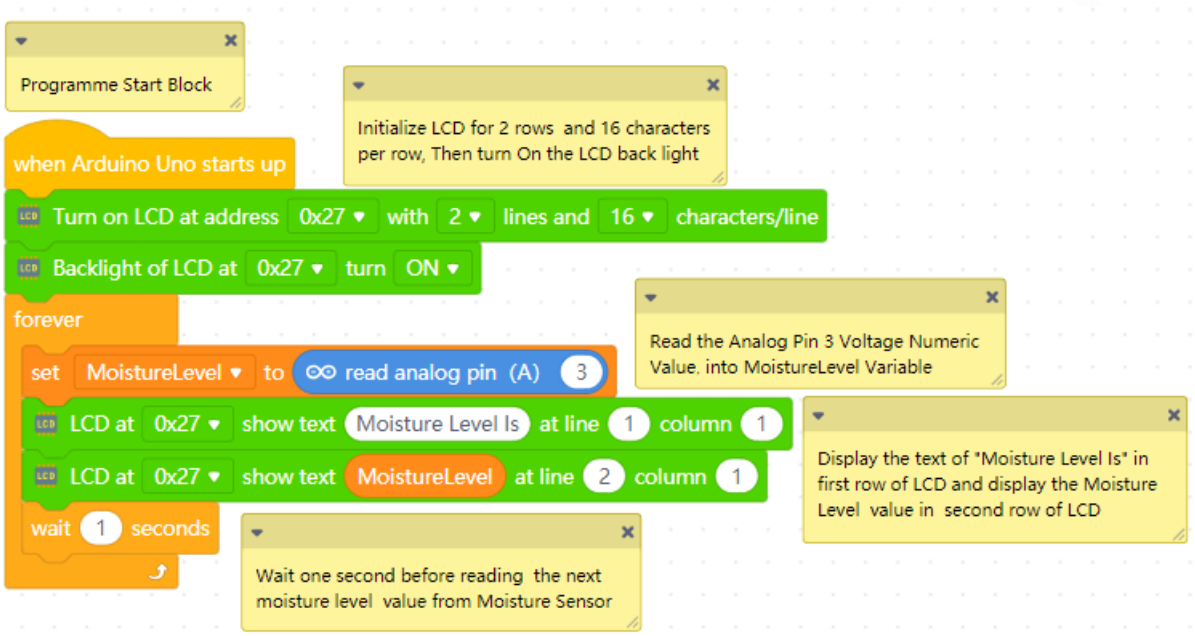
You have to connect the Moisture Sensor to MPatrol 1 Learning Kit, using the supplied 3 wire cable. Make the connections as mentioned below.

MoistureSensor "AOUT**"** Pin ->MPatrol 1 **OUT** Pin

Moisture Sensor **"VCC"** Pin ->MPatrol 1 **5V** Pin

Moisture Sensor **"GND"** Pin ->MPatrol 1 **GND** Pin

mBlock Scratch Program Code



As done in the previous projects, first initialize the LCD display and turn On the LCD back light. Here also we must implement all the moisture sensor reading and display codes, inside a **"forever"** block. Because we want to read and display the moisture level continuously until power off. Using **"set MoistureLevel to read analog pin (A) 3"** instruction, read the analog pin 3 voltage into "MoistureLevel" variable. (We have connected the Analog Out pin of the moisture sensor to Analog Pin 3 of Arduino Uno board). Before using this instruction block, we have to create a variable called "MoistureLevel". In the next step, display the moisture level value on the LCD screen. Here in the first row of LCD, we display the text of "Moisture Level Is". In the second row of LCD, we display the Analog 3 numerical value, which is proportional to the moisture level of soil. As mentioned earlier, you can scale this Analog 3 numerical value using some equation, before display on LCD, using the mathematical instruction blocks. Finally, we are waiting 1 second, before reading the next moisture level value from the moisture sensor. We need this delay, otherwise if we write data to LCD very fast, users cannot see them clearly.

Fix the Moisture Sensor to a flowerpot and pour water with some interval. Do not insert the moisture sensor into soil, over the line mark on the sensor. Also make sure water not touch with the electronics components of the sensor. Both these cases can make permanent damage to sensor. You can see the value on the LCD going down, when increasing the moisture level of soil. In practical situation, you can make this kind of circuit for control a garden water tap, according to the moisture level of soil.

If you are not familiar with the LCD display related coding, first do the "Project 18 – Electronics Distance Meter" before doing this project. I have explained about the LCD display, adding LCD display extension and related codes in detail in that project.

Project 22– Communicating with a Computer

Project Details

In this project we are learning about how to communicate with a computer. Here I use the word of computer, but it could be a personal computer or laptop. MPatrol 1 can send and receive data from/to a computer, so we can monitor data such like temperature, humidity, distance..etc on a computer screen. Further if we want to save these data for a future use, we can log these data to a file as well. Here we can use the computer screen instead of a LCD display. So we have a larger space to display our data, which is one of main limitation in LCD screens.

Serial Data Communication

Now we will see how MPatrol 1 can communicate with a computer. As we already know MPatrol 1 has a Arduino Uno board. This Arduino Uno board has a mini computer called a microcontroller. (Microcontroller is the heart of the Arduino Uno board and it controls all the operations of an Arduino Uno board) Microcontroller is doing this communication using two data lines. One line is for send data to the computer. We called it Transmission Line. Other line is for receive data from a computer. It is called Receive Line. In short form we called these lines as TX and RX lines. The way TX and RX lines exchanges data is called Serial Data Communication. Serial communication means, data bits are sending and receiving one by one. (In Parallel Data Communication, several data bits are sending and receiving at once)

To establish a serial communication with an external device like MPatrol 1, computer should have a serial communication port. We called it as a COM port. A computer can have several COM ports, so we have to select the correct COM port which our MPatrol 1 has connected. This is the same COM port you are selecting, when connect the MPatrol 1 to mBlock software for download your code. We can use the same USB cable and same computer USB port for this serial communication. In this case USB port acts as a Virtual COM port that enables us to do serial communication through a USB port. Virtual COM port driver is automatically installed, when you install the mBlock software.

Serial Monitor Software

Now we know that computer send and receive serial data using COM ports. We need some special software for display the data receive from computer COM port and send data out from same COM port. This software is called Serial Monitor Software. Using a serial monitor software, we can see the data coming through a COM port and also we can send data out from computer. Some serial monitor software support data logging as well. In that case we can save receive data into files for future use. When you open a serial monitor software, there are two important settings which you need to set, before doing any communication. Those are called COM port number and Baud Rate. There can be several COM ports in a computer, so you have to select the correct COM port which MPatrol 1 has connected. (This is the same COM port, which you use to download programme code from mBlock software, so easy to find). Baud Rate is the speed of communication. That means

bit transmit and receive speed in bit per second. There are several standard serial communication speeds such like 9600, 38400, 57600, 115200 etc. Arduino Uno board is communicating with the speed of 115200 bps (bit per second), so we have to use the same speed in computer as well. Both devices should communicate with same speed, otherwise it generates a lot of communication errors. So in the serial monitor software, Baud Rate should be selected as 115200.

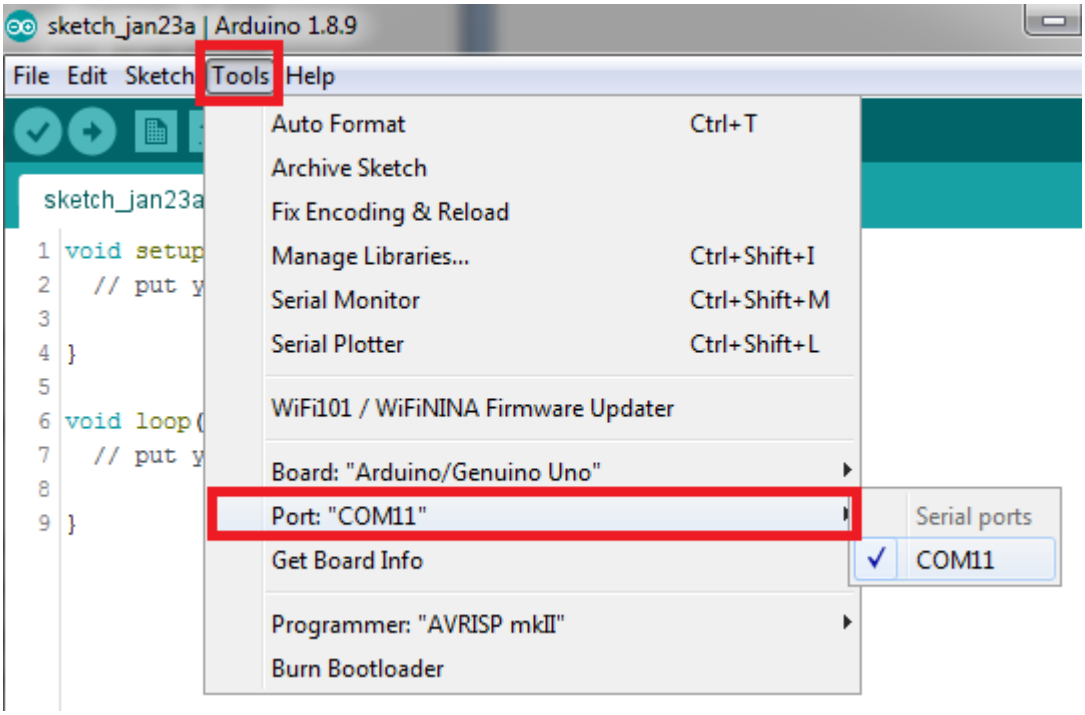
There are a lot of open source and paid serial monitor software are available to download in the internet. In this project, I have used the serial monitor software comes with Arduino Software IDE. Arduino Software is totally free to download and easy to use. Further, if you are familiar with the advanced programming language called "C", you can programme MPatrol 1 Learning Kit using the Arduino Software IDE as well. But it is not necessary to use the Arduino Serial Monitor software and you can use whatever serial monitor software you like.

Arduino Serial Monitor Software

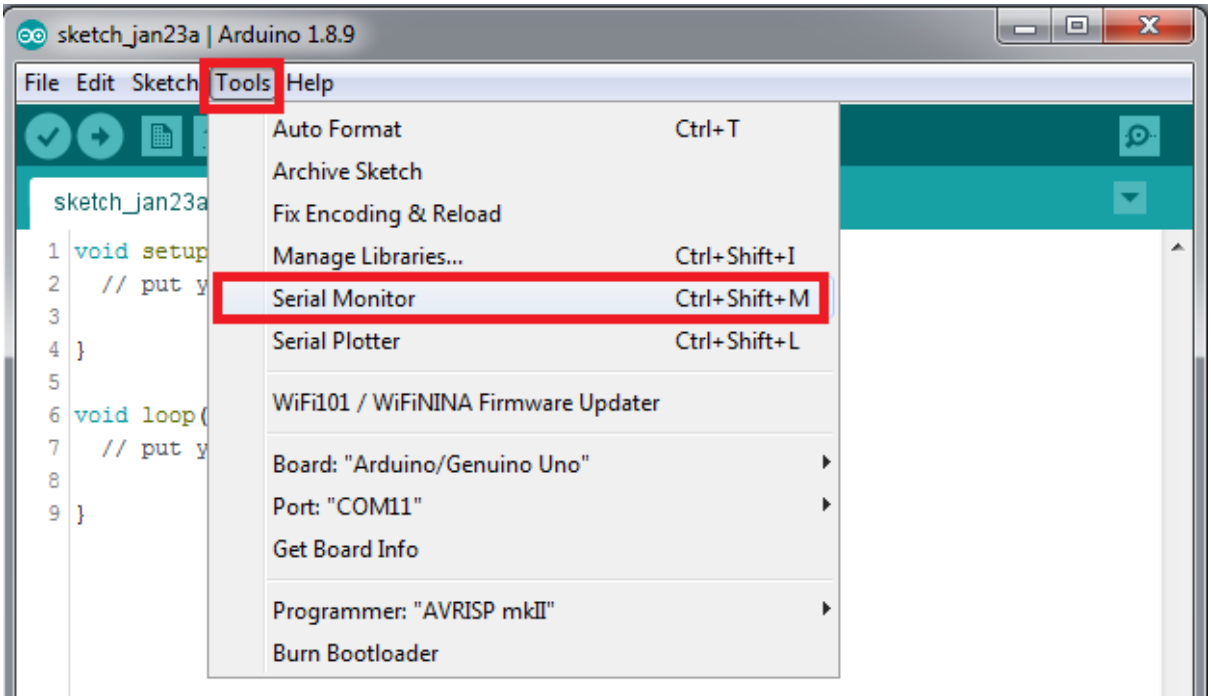
Now we will see the details of installing the Arduino IDE software and use the Arduino Serial Monitor. Go to the https://www.arduino.cc web site and download the Arduino IDE for your operating system. Then install the software using the downloaded setup file. After finish the installation, open the software.

(1) Select the COM port, which you have connected the MPatrol 1 to your computer. In this moment, make sure you have disconnected the MPatrol 1 from mBlock software. Because computer does not allow to connect multiple software to same COM port at same time. You can only connect one software at a time.

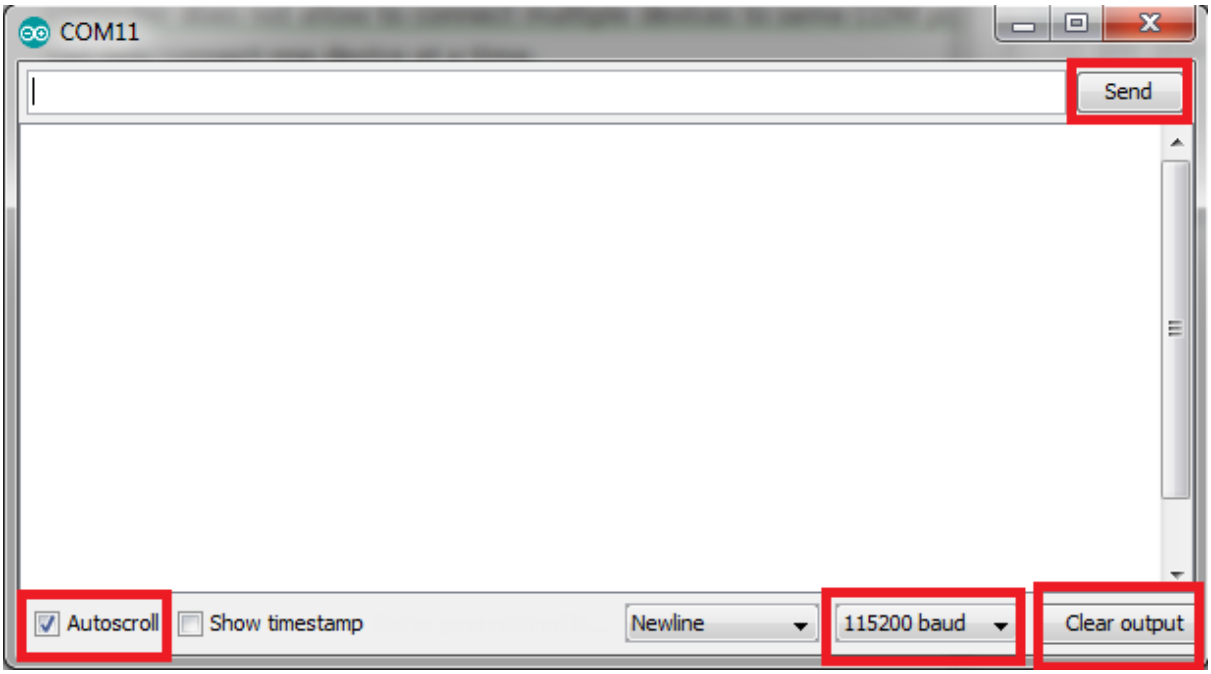
Go to **Tools -> Port :** and select the COM port, you have connected the MPatrol 1 to your computer. Make sure you have connected the mPatrol 1 to computer at this time using the USB cable, otherwise you may not able to see the correct COM port in COM port selection list.



(2) Now go to Tools -> Serial Monitor for open the Arduino Serial Monitor software.

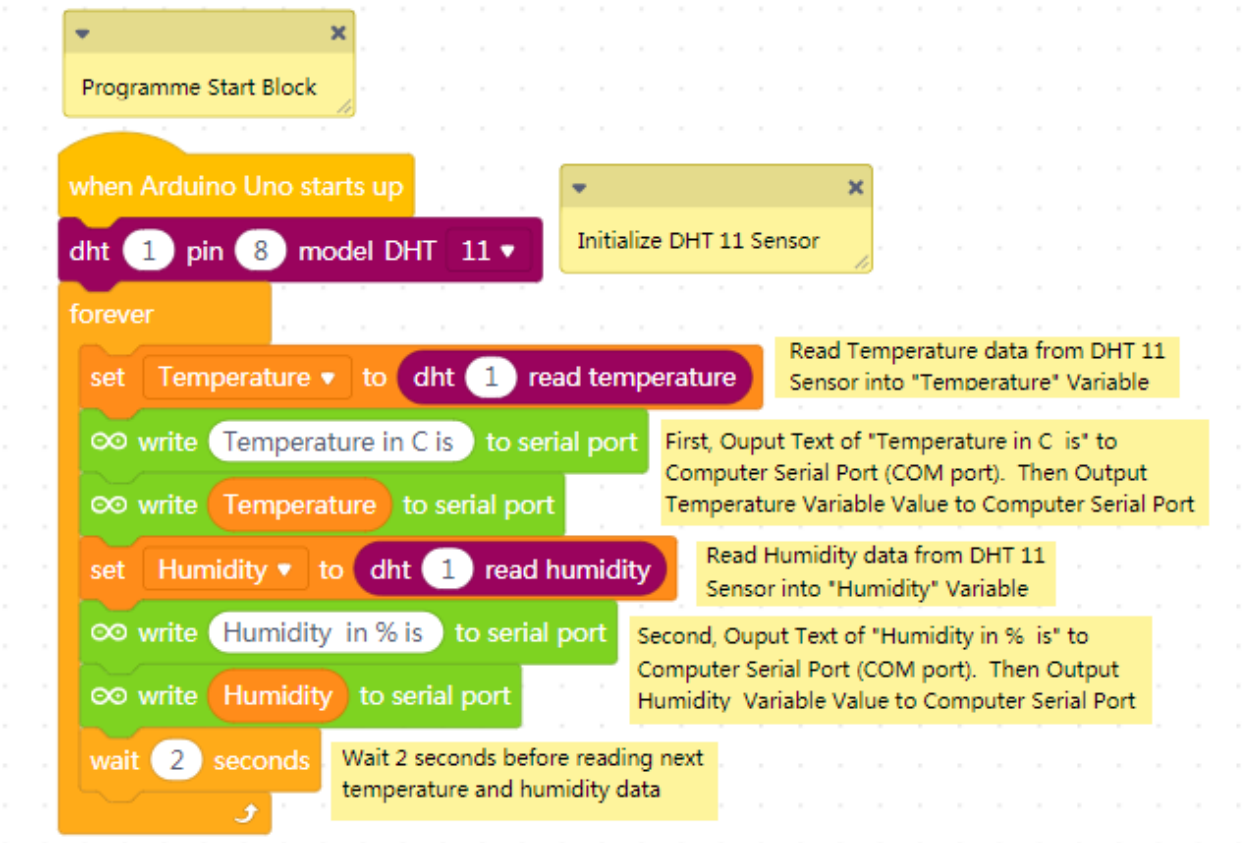


(3) Finally select the correct serial communication speed (Baud Rate) from bottom right hand speed selection list box. As MPatrol 1 is sending and receiving data in 115200 bps (bits per second), here we have select **"115200 baud"** as the data communication speed. Now you are ready to send and receive data to/from mPatrol 1. Whatever data send by the mPatrol 1, you can see in the middle large white area. Click on **"Autoscroll Check Box"** for automatically scroll the receive data in display area. If you want to erase the data in display area, click the **"Clear output"** button. If you want to send some data to MPatrol 1, type them in the box near to "send" button and click the **"send"** button. You can see the connected COM port in the top left hand corner of the serial monitor window.



Now we will see how to output data to computer from MPatrol 1, using the block instructions of mBlock software. Make sure to close this serial monitor window, before download the programme from mBlock software to MPatrol1. Otherwise computer will not allow you to connect MPatrol 1 to mBlock software, as we cannot connect two software to same COM port at same time.

mBlock Scratch Program Code



In this Programme, we read temperature and humidity data from DHT 11 sensor and display them on a computer. In "Project 19 – Temperature and Humidity Meter" project, I explained about the DHT 11 sensor in detail. So if you are not familiar with the DHT 11 sensor, please first do "Project 19 – Temperature and Humidity Meter" project, before this project. Further in "Project 19", we displayed temperature and humidity data on a LCD display. Because of the space limitation on LCD display, we first displayed temperature data on LCD and then displayed the humidity data after erasing the temperature data. But in this project we display temperature and humidity data on a computer screen, so we do not have any space limitation here and we can display whatever data we want at once.

As usual we start the programme with **"when Arduino Uno starts up"** block and then initialize the DHT11 sensor with **"dht 1 pin 8 model DHT 11"** block. Rest of the code, we have to implement inside a **"forever"** block, as we are going to output temperature and humidity values continuously to computer until power off. Before implement anything inside the "forever" block, create two variables called "Temperature" and "Humidity". Then using the **"set Temperature to dht 1 read temperature"** block, read the temperature value into the "temperature" variable. **"write Temperature in C is" to serial port"** block send the text of "Temperature in C is" to computer serial port. (COM port). Then **"write Temperature to serial port"** block send the value of "Temperature" variable to computer serial port. So we can see these data in computer using serial monitor software. So as you

can see what ever inside the **"write ____ to serial port"** block, is being sent to the computer serial port. This is how we output data to the computer serial (COM) port.

Finally read the humidity data into "Humidity" variable using the **"set Humidity to dht 1 read humidity"** block instruction. Then output these humidity data to computer serial (COM) port, as same way we output temperature data to computer serial port. At end end of the programme, we need some delay. Because if we output data to computer continuously, without any delay, those data will display on computer screen very fast, so users may not able to see them clearly. Here we have set 2 second delay, before reading the next temperature and humidity data.

Please make sure before uploading the programme code to Arduino Uno board, close the serial monitor software. Also before open the serial monitor software for monitoring data, disconnect the Arduino Uno board from mBlock software. Because computer is only allow to connect one software for a particular serial (COM) port, at a time. If you try to connect multiple software to same serial (COM) port, you should get a connection error message.

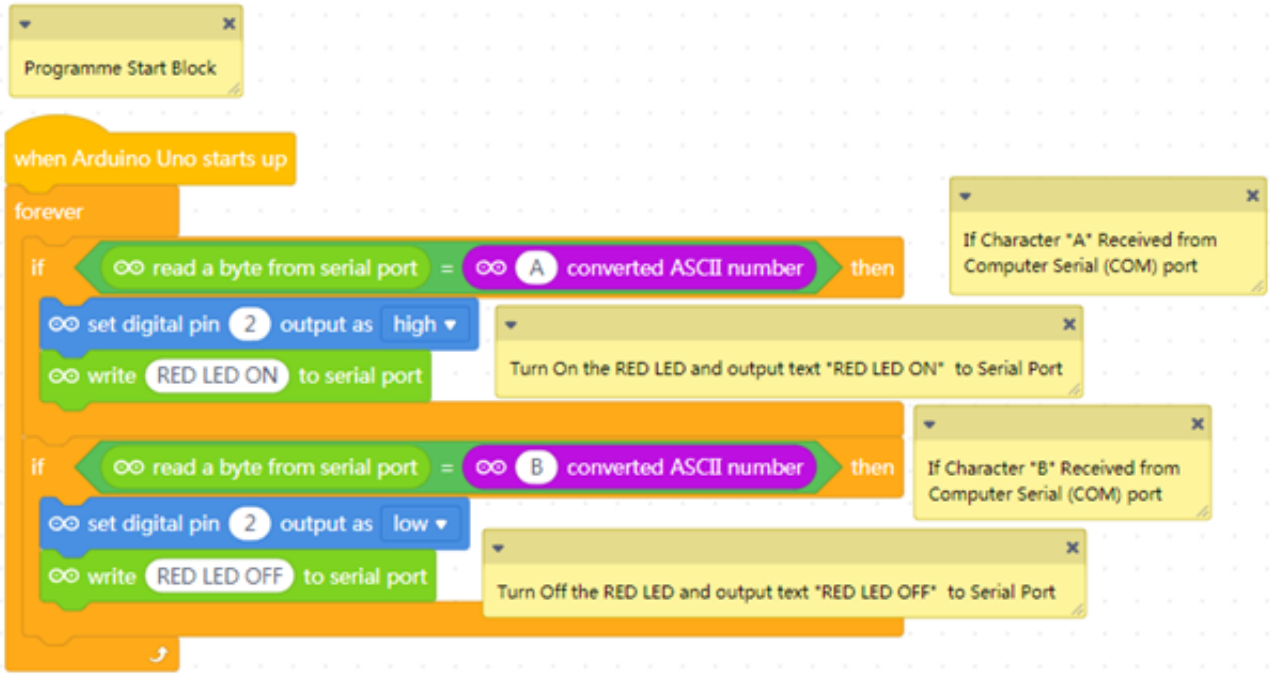
Project 23– Device Controlling Using Computer Commands

Project Details

In this project we are learning about how to control Electronic Devices using Computer Commands. Here we are going on and off the RED Led of MPatrol 1 using the, commands send from Computer Serial (COM) port. In same way you can control other devices such as Motor, Buzzer ..etc, using the commands send from Computer Serial (COM) port. In "Project 22 – Communicating with a Computer" project, we learned about Serial Communication and related Computer Software in detail. So I am not going to explain about them in detail, in this project. So please first do the "Project 22 – Communicating with a Computer" project, before do this project.

In this project, we are going to use same serial monitor software (Arduino), which we used in last project, for send commands to MPatrol 1. But you can select any serial monitoring software you like, for send serial commands. Now we will see how to implement the programme for control devices using Computer Serial Commands.

mBlock Scratch Program Code



In this project I am going to on and off the Red LED, using two separate commands. I send character "A" from computer serial (COM) port for turn On the Red LED. Same way I send character "B" from computer serial (COM) port for turn Off the Red LED. So in the serial monitor software, if you type "A" and hit the "Send" button, Red LED should be turn On. If you type "B" and hit the "Send" button, Red LED should be turn Off. Here I have selected

character “A” and “B” as the On/Off commands, but you can select whatever characters you like as the serial commands.

As most of our projects, here also we have to use a “forever” block to implement the programme logic codes. Because our programme should be able to read the computer commands at any time, until power off. In the programme code, we have used two **“if then”** blocks to implement the LED on/off logic. We use a **“if then”** block to check, wether a condition has got satisfied or not. Now let see what is the condition we check in first “if then” block. Here we check whether the data received from computer serial (COM) port is equal to the character “A”. We use **“read a byte from serial port**” block instruction for read the data bytes sending from computer’s serial (COM) port. Then what is the function of **“A converted ASCII number”** instruction block? This instruction block converts the character “A” into its numerical equivalent figure. Why we cannot directly compare the character “A” with the received serial data?

Each character in alphabet has its own numeric figure. As example numerical value of A is 65, B is 66, C is 67, a is 97 etc... This numeric figure of a particular character is called the ASCII value of that character. Microcontroller inside the Arduino Uno board cannot understand the characters directly. It can only understand the ASCII value of the character. Because of that, first we have to convert the character “A”, into its ASCII value (equivalent numeric figure), before compare with received serial data. We use “**__ converted ASCII number”** instruction block for covert characters into its equivalent ASCII value.

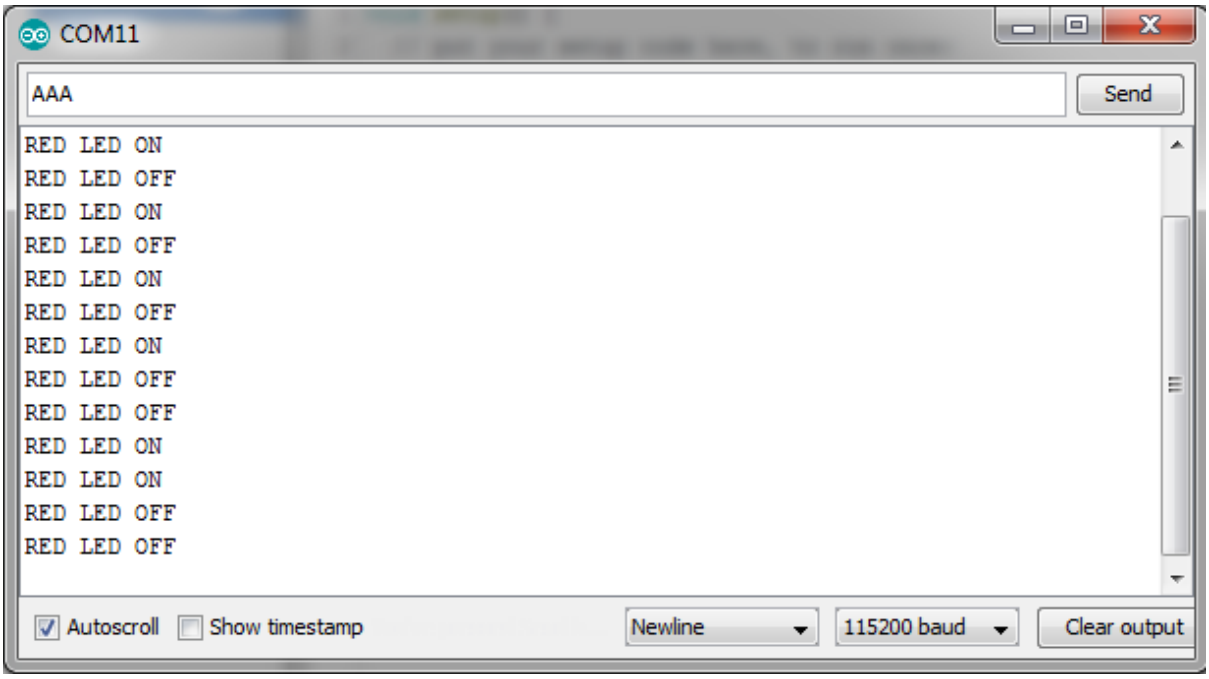
If the condition satisfied, that mean if we have received the character “A” from computer’s serial (COM) port, then we do two things. First using the **“set digital pin 2 output as high”** instruction block, we turn On the Red LED. Second, using the **“write RED LED ON to serial port”** instruction block, we display the text of “RED LED ON” in our serial monitor software. On this way, we can get a feedback message for our serial command. Function of the second if block is similar to the first if block. Here we check that, if the character “B” has received from the computer serial (COM) port. If so, we turn Off the Red LED and display the text of “RED LED OFF” on serial monitor software.

Now let see how we test this programme code. After uploading the programme code to MPatrol 1, open the serial monitor software. Then type character “A” in serial data out box and press the “SEND” button. You should see the Red LED turns On. You should also see the text of “RED LED ON” in serial monitor software’s receive data window. When you send the character “B” as same way describe above, you should see the Red LED turns Off and the text of “RED LED OFF” should display on serial monitor software. This process will repeat again and again, when you send the “A” and “B” characters.

It is always a good idea to send multiple “A” or “B” characters (2 or 3), rather than sending only one character. Then if the microcontroller inside the Arduino board miss the first character, it can still read the second or third character and responds to the serial command. Eg. Type AAA in serial data out box and press the “SEND” button.

Please make sure before uploading the programme code to Arduino Uno board, close the serial monitor software. Also before open the serial monitor software for sending serial

commands, disconnect the Arduino Uno board from mBlock software. Because computer is only allow to connect one software for a particular serial (COM) port, at a time. If you try to connect multiple software to same serial (COM) port, you should get a connection error message.



Project 24– Working with Mathematical Blocks

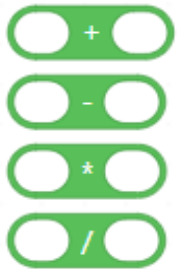
Project Details

In this project, we are learning some mathematical blocks, which are available in mBlock software. This project is not directly related to any electronic circuits, but you can use these mathematical blocks in your electronic projects for various purpose. As mentioned in water level meter project, you can use these mathematical blocks for convert raw analog numerical data into physical units such like mm or cm. If you want to display a sensor value on LCD without any decimal places, you can remove the decimal places using the mathematical blocks. Or else, if you want to increase the range of a sensor data values, you can multiply the raw analog numerical values by some factor. These are just only few examples. So mathematical blocks are very useful in electronic projects.

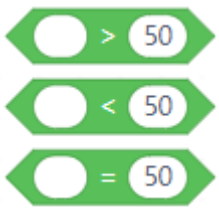
mBlock Mathematical Blocks

You can find the mathematical blocks in mBlock software under the **"operators"** blocksubcategory.

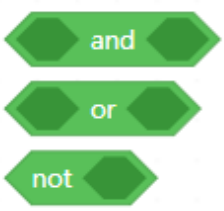
(1) Arithmetic Blocks – These blocks can use for addition, subtraction, multiplication and dividing two numbers. You already used some of these blocks in previous projects.



(2) Condition Blocks – These blocks can use for checking Greater Than, Less Than and Equal conditions. You already used some of these blocks in previous projects.



(3) Logical Operation Blocks – These blocks can use for AND, OR and NOT logical operations.



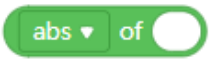
(4) Special Operation Blocks

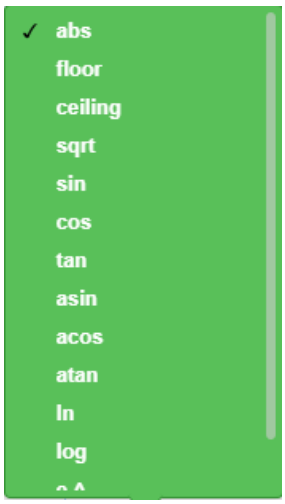
(a)pick random number – This block can use for generating a random number between two figures.

(b) mod – This block calculates the modules of two numbers. (remainder of dividing two numbers)

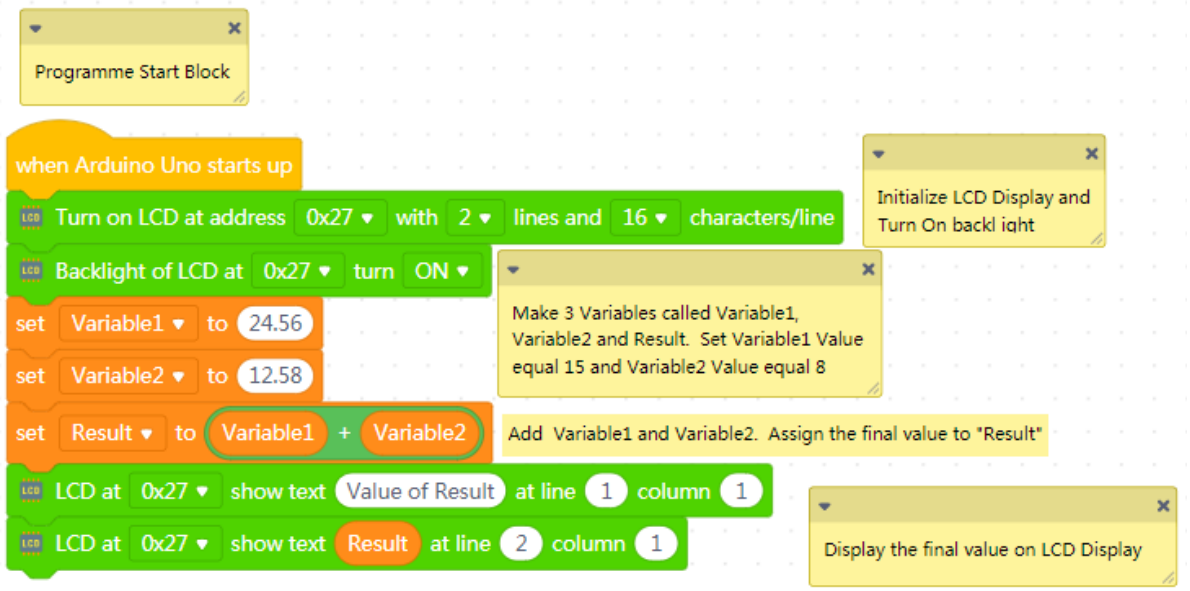
© round – This block rounds the values with decimal places to the nearest integer.

(d) Advance Mathematical block - This block is a collection of advanced mathematical operations compacted into a single block. By clicking the down arrow, you can select the mathematical operation you like to perform. This block contains a lot of mathematical operations such like power calculation, square root calculation, absolute value calculation, trigonometry calculation, exponential calculation, logarithm calculation …etc.





mBlock Scratch Program Code



In this programme, first we initialize the LCD display as usual and turn On the LCD back light. Then we make 3 variables called Variable1, Variable2 and Result. Next assigns 15 to Variable 1 and assigns 8 to Variable2. (using **"set Variable to "** instruction block). In **"set Result to Variable1 + Variable2"** instruction block, we use **"Addition"** mathematical block to add Variable1 and Variable2 together and assign the final added value to "Result" variable. Finally, we display the value of "Result" variable on LCD display.

Now change the Varible1 and Variable 2 values for different figures and monitor the final value display on LCD. You can use values with decimal figures as well for Variable1 and Variable 2.

Next change the programme for different mathematical blocks and for different variable values and observe the result value on LCD. In this way you should be able get a good understand about mathematical function blocks and how to use them.

In this programme, we did not implement the codes inside a "forever" block. The reason is, here we do not want to do the same mathematical calculation again and again. So, we do all the mathematical operations only one time.

Project 25– Working with String Manipulation Blocks

Project Details

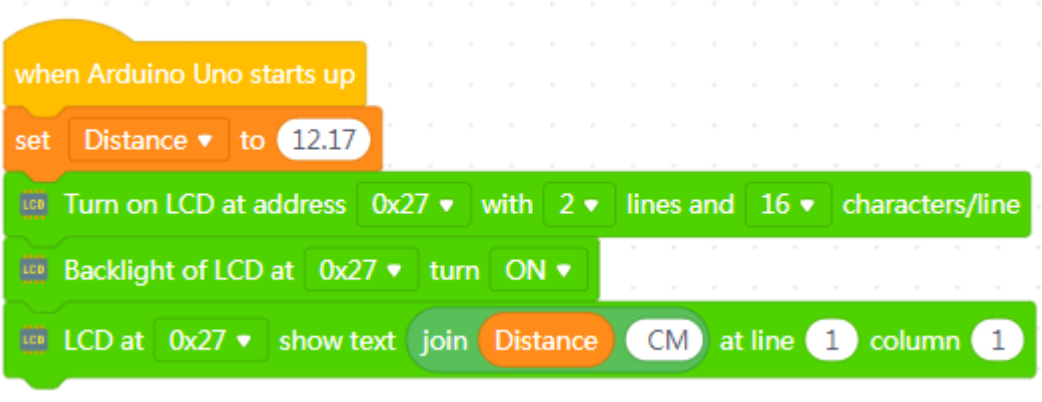
In this project, we are learning about some string manipulation blocks in mBlock software. This project also not directly related to any electronic circuits, but string manipulation operations can help to improve your electronic projects in various manner. As an example, you can nicely format text on LCD screens or on a Computer Serial Monitors, using the string manipulation blocks.

String manipulation is one of the main topics in computer programming and computer programmers have to use string manipulation functions very frequently in their programs. Some of its useful applications are adding two words or sentences together, find the number of characters in a word or sentence, find a word or sentence within another word or sentence …etc.So, the knowledge about the string manipulation is very important for a computer programmer.

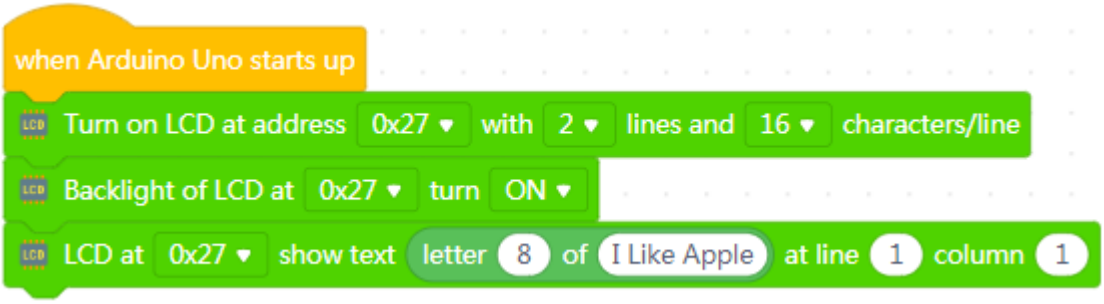
String Manipulation Blocks in mBlock Software

Now we will study the string manipulation block instructions, available in mBlock software. You can find the string manipulation blocks in **"Operators"** block sub category.

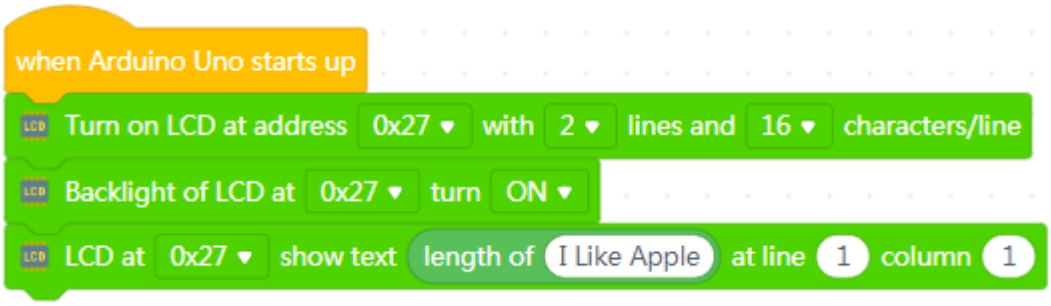
(1)Word Connecting Block – With this block, you can connect two numbers/words/sentences together. In below example, we join Distance variable figure and text "CM" together, so they can display on the LCD in same row. If you want to keep a space between Distance figure and "CM" text, put a space before the "CM" text. In here you can see "12.17 CM" on the first row of the LCD display.



(2)Display Letter by Index Block - Using this block, you can display a particular letter in a word or sentence, according it index. In below example, 10th letter of the sentence "I Like Apple" is letter "A". So you can see the letter "A" on the LCD display.



(3)Length Calculating Block - Using this block, you can calculate the number of letters in a word or sentence. In below example, sentence "I Like Apple" contains 12 letters. So you can see the figure 12 on the LCD display.



(4) Search Word/String with in another Word/String Block – Using this block, you can search any particular letter, word or string, within another word or string. In below example, if the sentence "I like Apple" contains the word "App", condition for **"if then"** block gets TRUE. If not, condition for **"if then"** block gets FALSE. So according to that, you can see, "YES" or "No" text on the LCD display. So in below example, as the sentence "I like Apple" contains the word "App", you can see the text of "YES" on the LCD display. If you change the "App" text to "APP", then the text "NO" will display on the LCD display. You can modify the main string and search string and see how this block instruction searches the words and sentences.


when Arduino Uno starts up
Turn on LCD at address 0x27 with 2 lines and 16 characters/line
Backlight of LCD at 0x27 turn ON
if I Like Apple contains App ? then
LCD at 0x27 show text YES at line 1 column 1
else
LCD at 0x27 show text NO at line 1 column 1