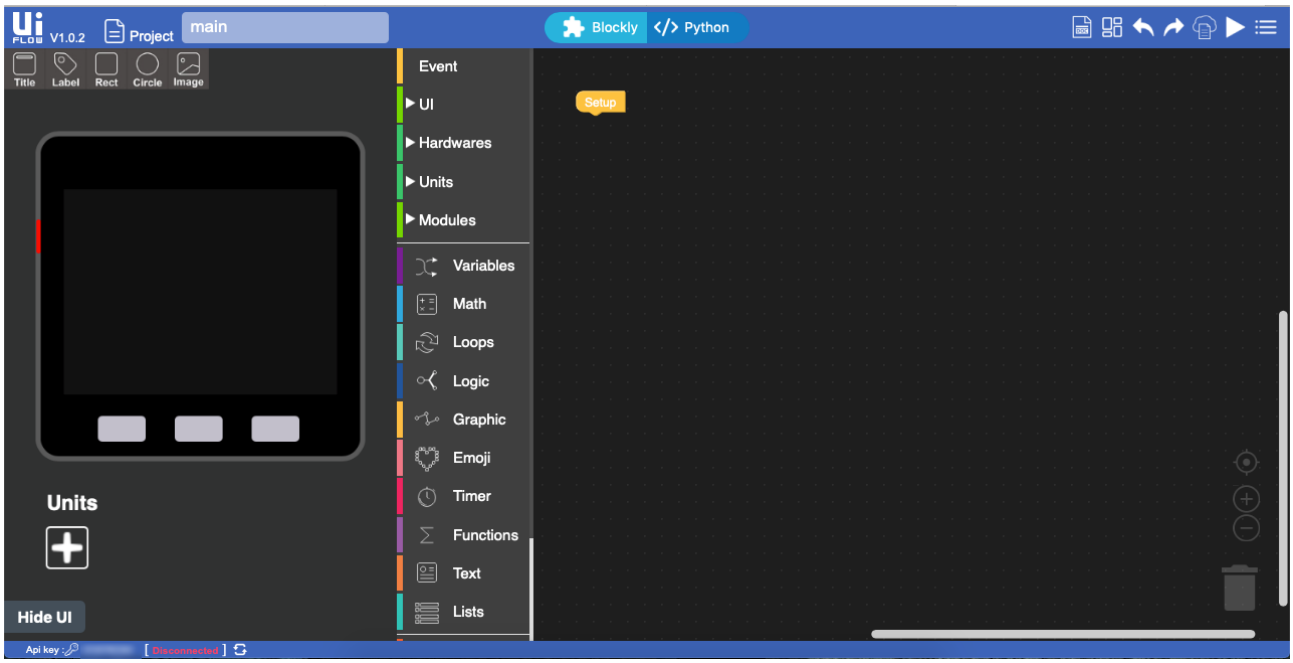
UIFlow Handbook



Written by
Adam Bryant

Contents.

Introduction,
Setup,
Exploring the Blocks.
  Event,
    Loop,
    Button,
    Obtain Button,
U.I. (user Interface)
Hardwares,
    Speaker,
    RGB,
    IMU,

Units,
    Environmental,
    Angle,
    PIR,
    Neopixel,
    Joystick,
    Light,
    Earth,
    Make,
    Servo,
    Weight,
    Tracker,
    Button,
    Duel Button,
    RGB,
    Relay,
    Heart,
    ADC,
    Colour,
    DAC,
    IR,
    NCIR,
    Thermal,
    TOF,
Modules,
    Stepper Motor,
    Servo,
    Bala,
    Bala Motor,
    Lego Motor,
Variables,
Maths,
Loops,
Logic,
Graphic,
Emoji,
Timer,
Function,
Text,
Lists,
Advanced,
    Pin,
    GPIO,
    PWM,

ADC,
DAC,
UART,
I2C,
Execute,
Network,
MQTT,

Remote,

Introduction.

This handbook is really just a collection of notes that I have assembled while learning M5Stack's UIFlow programming environment.
As of writing the current version is 1.1.0 and I'm running OSX 10.14.02 (Mojave).
UIFlow is only available on the M5Stack server with no plans to release a stand alone version due to security issues.

Throughout this hand book I will refer to the M5 Series as the M5Stack which consist of the Core (Black), Gray, Go, and fire. The M5Stick as the white or gray M5Stick versions.
As of writing, there is no UIFlow firmware for the M5Camera.

About the M5Stack.

The M5 Stack system is based around a 50mmX50mm (2"X2") base unit with a variety of add-on modules based around the ESP32 by espressif.
The M5Stack allows for rapid research and development of projects and products and along with the UIFlow and Micro Python programming environments lowers the age limit for learning programming.

The models and their differences are highlighted in the following table.

Difference	M5Core	M5Core Gray	M5Core White (Go)	M5 Fire	M5 Stick White	M5 Stick Gray
Mems	None	MPU9250	MPU9250	MPU9250	Optional	MPU9250
Ram	520KB	520KB	520KB	520KB+4M	520KB+4M	520KB+4M
Flash	4MB	4MB	4MB	16MB	4MB	4MB
Screen	320x240 Colour	320x240 Colour	320x240 Colour	320x240 Colour	64x128 Mono	64x128 Mono
Speaker	Yes	Yes	Yes	Yes	No	No
Base Plate	Connectors	Connectors	Go Base	Go Base	N/A	N/A
Grove Ports	1	1	3	3	1	1
RGB LED	None	None	10x SK6812	10x SK6812	None	None
Battery	150mAh	150mAh	550mAh	550mAh	80mAh	80mAh

Disclaimer, these specification are subject to change by M5Stack.

Colour Setting.

The WS2812, RGB LED, and the colour screen on the M5Stack share the same colour setting functions. Colour can be set using the colour picker or but setting the individual channels R, G, and B, with a value of 0 to 255 for a total of 16 Million possible shades and colours.

You are not restricted to manually putting in the values for colours and brightness, the value boxes allow for the use of a variable block to control the values from code calculation.

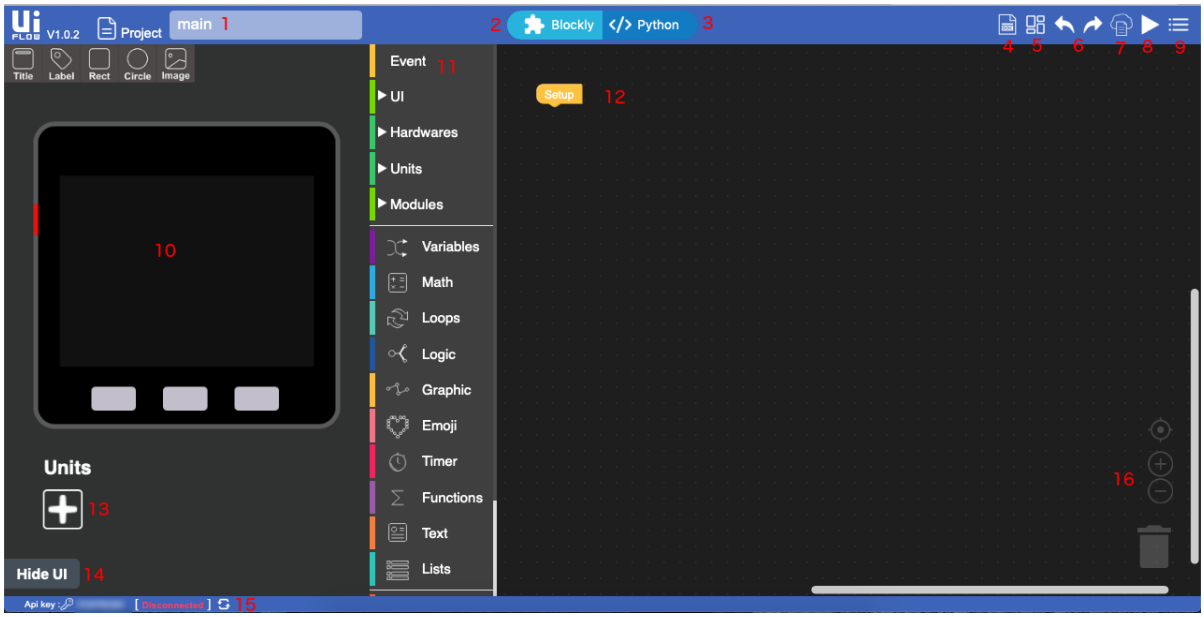
The top block shows the default options for the colour value block. The light blue blocks are one you add the values from 0 to 255. In the bottom block I have added the purple Variable blocks R,G, and B, which are place holders for the variable that can be controlled.

The colour picker only has a choice of seventy colours and shades which are connected to the predefined colour pallet defined in Micropython.

In addition to the colour selection, we also have a separate brightness block where we can set the value manually or via a variable.

Getting Started with UIFlow on the M5Stack and M5Stick.

UIFlow contain two programming environments, the visual, block environment, which consist of the UI Designer, the Block Menu and the code window.



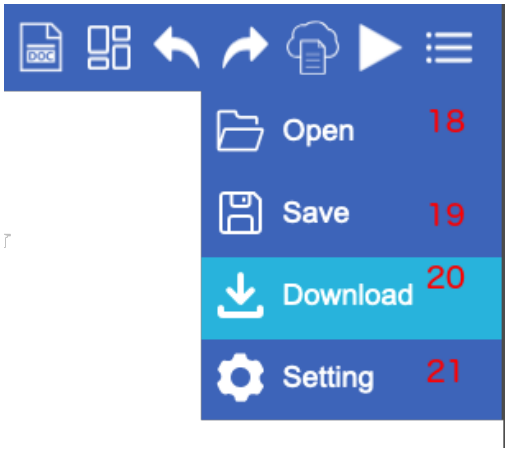
and the Micro python environment

The individual parts of the interface are

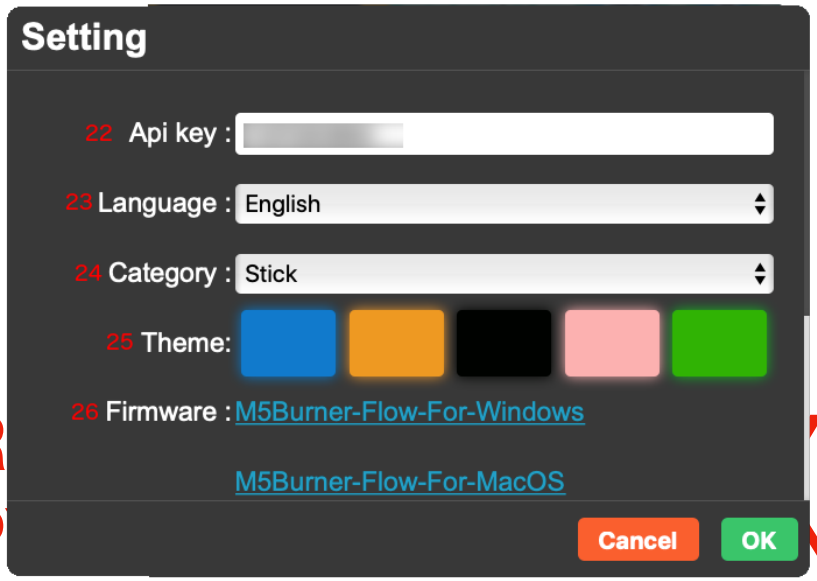
1 - The Project name,
2 - Blocky window display,
3 - Python Window Display,
4 - UIFlow Documentation,
5 - Demo Program List,
6 - Undo and Redo buttons,
7 - Resource Manager,
8 - Play/Test code,
9 - Settings,
10 - UI Designer,
11 - Code block menu,
12 - Blocky project window,
13 - Unit select area,
14 - Show/Hide UI Designer,
15 - M5Stack/M5Stick Status display,
16 - Workspace controls,
17 - Python Code window,
18 - Opens a file saved to a users computer,
19 - Saves a copy of the code to a users computer,
20 - Downloads the program permanently to the M5Stack or M5Stick,
21 - Settings window,
22 - Api Key entry box,
23 - Language Select,
24 - M5Stack or M5Stick mode selector,
25 - M5Burner firmware installer selector,

Before we start using UIFlow, we need to install the UIFlow firmware on our M5Stack or Stick.

First we need to click on (9) the setting window to drop down the setting list.



The we need to click (21) Setting to bring up the setting window.



For windows users you will need to click the windows link for the windows firmware installer, for OSX you will need the MacOS installer.
At present, there is no Linux version that I know of.

Important Note 1

Sometimes there is an issue with the installing of the CP20X Virtual serial device driver, sometimes it is connected to OSX and sometimes it is the USB lead.
Always suspect the USB lead is at fault first.

Important Note 2

On OSX there is a security issue that causes M5 burner to pop up a message saying it is corrupt. To get around this you need to open the command prompt and type
`sudo spctl --master-disable`
Download and run M5Burner (don't do anything else), Once finished you need to type
`sudo spctl --master-enable`
in the command line to reset security settings. I'm not sure what is causing the corrupt message but, disabling and reenabling spctl just for this app has not caused any problems.

If you are still having connection trouble (even in Arduino) you need a new USB C lead, I recommend a good quality lead from Anker.

Download or Test?

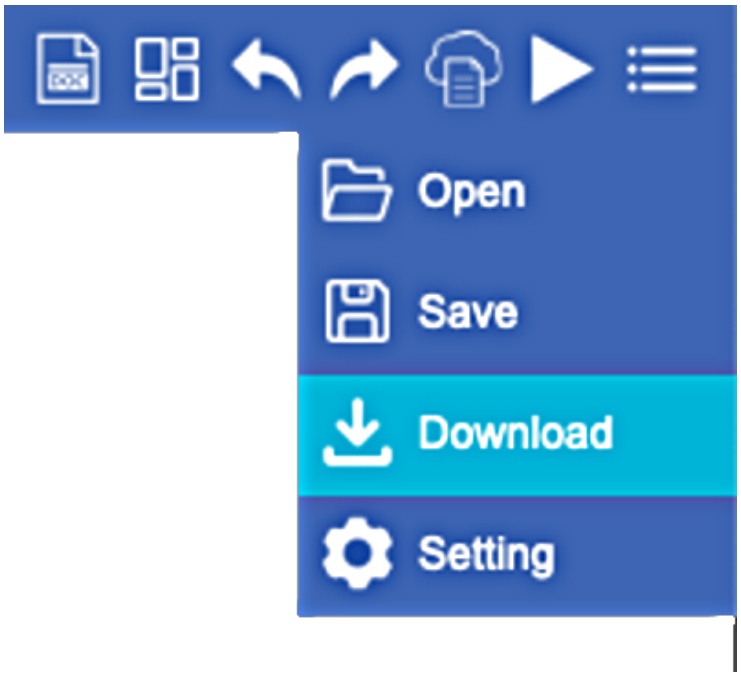
There are two ways to run a program on the M5. The first is "Play" which temporarily downloads the programs to the m5 which will retain it in memory until the M5 is powered off. The second mode is "Download", this downloads the program into the M5's program storage space permanently.

Some programs (Like MQTT programs) can only be tested when stored on the M5.

To play the program without downloading it to the M5's memory we click on the "Play" button (highlighted blue here due to colour scheme), after a while, if there are graphical UI elements, they will appear on the screen.

<insert photo>

To permanently store a program that you have written on the M5 Stack or stick you need to click the Download button (hidden under the three lines) and wait for the device to finish receiving the file and restart.



In most cases, if the download worked, the device will restart and automatically load the program. If, However, it didn't run you can manually run the program by pressing the middle button of the M5 Stack (including the Go and Fire) to open the app menu and use the right button to move down to the downloaded program and then press the middle button to run the program.

Exploration of blocks.

Now that we have had a play with some of the units and blocks, let's explore the blocks and see if we can understand what they do.
Blocks in UIFlow can be separated into three distinct groups. I refer to these groups of blocks as loops, functions and values.

Loops,
Loops will continue to run through code that is placed inside them. There are several different kinds of loops within UIFlow, The common loop is the main program loop that contains our code that controls everything else.

Functions,
Functions are the commands of our program that make the M5Stack do things. These functions can be internal commands like maths or they can be external commands that controls the hardware plugged into the M5.

Values,
Values are numbers returned by calculations run in the code, values we specify that a program must use or they can be values given to us by the external hardware.

Events

The first block you will see whenever UIFlow is opened is the setup block and is the only block that can not be thrown in the bin.

This is the equivalent of Arduino's Void Setup function and contains functions that you only want to run once at the beginning of the program.

The equivalent in Micropython to the setup block is as follows.

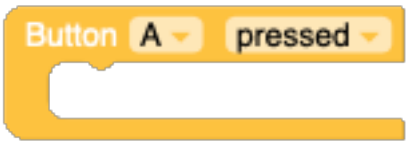
```
From m5stack import *
from m5ui import *

clear_bg(0x111111)

btnA = M5Button(name="ButtonA", text="ButtonA", visibility=False)
btnB = M5Button(name="ButtonB", text="ButtonB", visibility=False)
btnC = M5Button(name="ButtonC", text="ButtonC", visibility=False)
```

After this we have the main program loop. Inside of this block we place the main program that we want to continuously repeat.

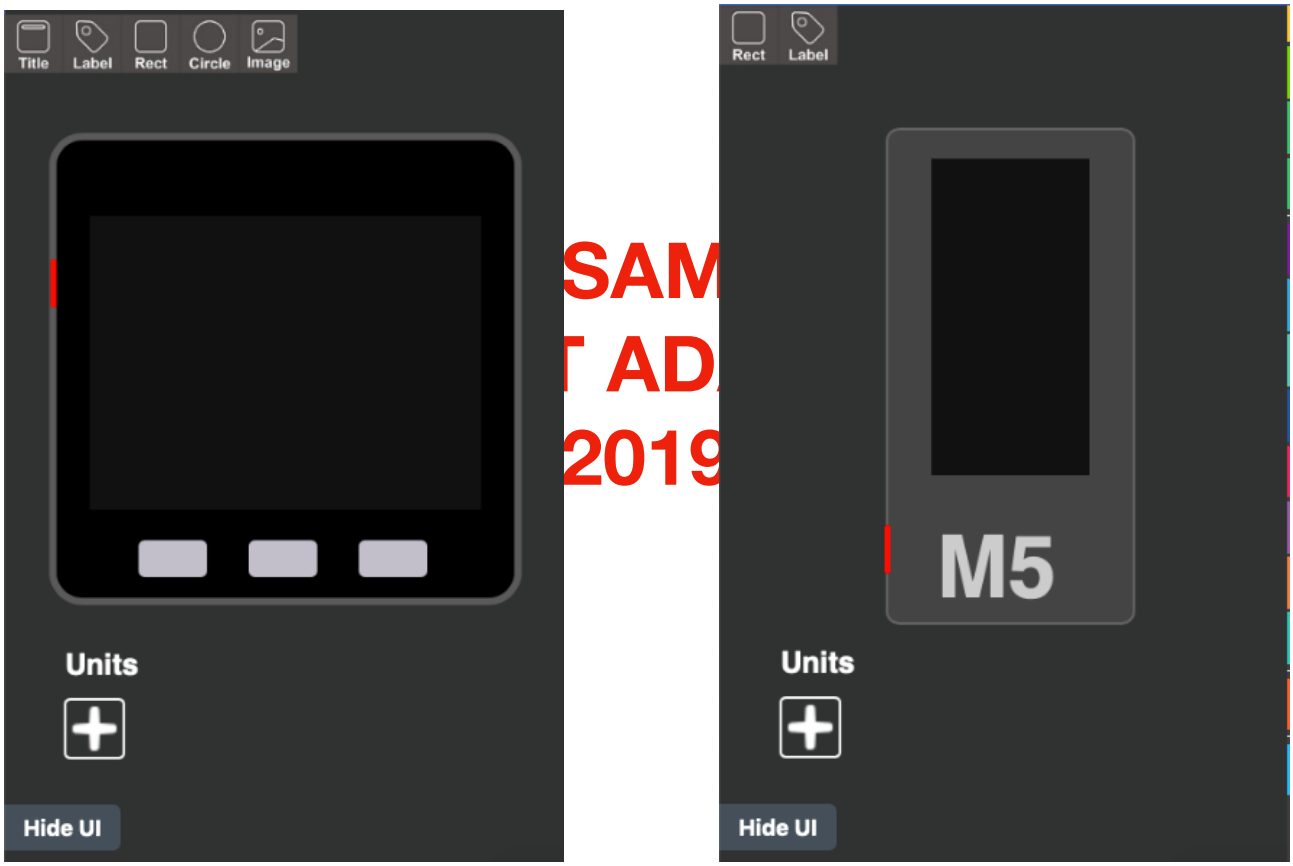
As well as the main loop, we have the button loop that continuously runs code when one of the three button on the from of the M5Stack, M5Go, and M5Fire.



U.I. (User Interface)

The UI Menu only shows code blocks connected to the User Interface blocks that have been added to the screen of the virtual M5Stack Core or Stick.

To switch between the Core and Stick UI Designer, you need to go into the settings and select which mode you want from the drop down list.



M5Stack Core UI Designer

M5Stick UI Designer

Some code blocks are used by all Interface elements, the following table shows which UI items use which function blocks. This table also shows which UI elements are usable on the Core and the stick.

Code Block	Title	Label	Rectangle	Circle	Image
Show Text	Yes	Yes	Yes	Yes	Yes
Show	Yes	Yes	Yes	Yes	Yes
Hide	Yes	Yes	Yes	Yes	Yes
Set Colour	Yes	Yes	Yes	Yes	Yes
Set Colour RGB	Yes	Yes	Yes	Yes	Yes
Set Background Colour	Yes	Yes	Yes	Yes	Yes
Set Background Colour RGB	Yes	Yes	Yes	Yes	Yes
Set Width and Height	No	No	Yes	No	Yes
Set Width	No	No	Yes	No	Yes
Set Height	No	No	Yes	No	Yes
Set X and Y Position	No	No	Yes	Yes	Yes
Set X Position	No	No	Yes	Yes	Yes
Set Y Position	No	No	Yes	Yes	Yes
Set Radius	No	No	No	Yes	No
Available to M5 Stack	Yes	Yes	Yes	Yes	Yes
Available to M5 Stick	No	Yes	Yes	No	No

Table 01 UI Block compatibility.

The U.I blocks.

Show text - This block print a string of characters on the screen.
Show - Displays the element.
Hide - Hides the element.
Set Colour - This block Uses the colour picker to set the element colour.
Set Colour RGB - Allows you set the colour by defining each channel a value between 0 and 255.
Set Background Colour - This block Uses the colour picker to set the element colour.
Set Background Colour RGB - Allows you set the colour by defining each channel a value between 0 and 255.

These are not the only draw functions available to use. In a later chapter I will cover the LCD graphical draw functions.

Off Note - While the colour functions are available for the M5Stick to use, the functions are not really of any use because the M5Stick only has a mono screen.

Internal Hardware

The various core units can come fitted with some additional hardware, the hardware available is as follows

Speaker

Music can be played on the M5 range through an internal speaker. to produce sound in UIFlow we can use the **Speaker Beep** function and the **Play Note** function blocks. The Micro-python equivalent to these blocks are **speaker.tone()** and **speaker.sing()** functions.

Speaker Beep

The speaker Beep block allows us to make sounds by setting the frequency of the sound and the duration of the sound.

Play Tone

The Play Tone block allows users to play musical notes and set the duration in beats or fractions of Beats.

Speaker Volume

Speaker volume allows us to set the volume of the speaker commands.

RGB

Only available on the M5Go Base Plate

The M5Go base plate has two strip of five SK6812 RGB LED's mounted on each side. Unlike the Neopixel (see the Neopixel section) they are controlled blocks found in the Hardware>RGB section. The colours of each individual LED can be set as explained in the Colour section discussed earlier.

The blocks we use for the RGB LED's are as follows.

Set RGB Bar Colour,
Set the colour of all LED's using the colour selector.

Set RGB Bar Colour R,G,B,
Sets the individual colour channels of all LED's using values of 0 to 255.

Set (Side) RGB Bar Colour
Set the colour of all LED's on one side of the base using the colour selector.

Set (Side) RGB Bar Colour R,G,B,
Sets the individual colour channels of all LED's on one side of the base using values of 0 to 255.

Set (individual) RGB Bar Colour,
Set the colour of individual LED's using the colour selector.

Set (individual) RGB Bar Colour R,G,B,
Sets the individual colour channels of separate LED's using values of 0 to 255.

Set RGB Brightness,
Sets the RGB LED Brightness.

IMU (Inertial Measurement Unit)

Available on the M5Go Base Plate M5 Stick Gray or as an optional extra on other units.

The IMU or Inertial Measurement Unit is a device that is used to measure movement, and angle that the M5 devices are placed in. Unlike the sound or RGB LED's the IMU doesn't have functions (as such) added to it but, instead returns values for other functions to use.

The block used for the IMU are as follows.

Get X,
Gets the X axis data from the IMU for other functions to use.

Get Y,
Gets the Y axis data from the IMU for other functions to use.

Get Z,
Gets the Z axis data from the IMU for other functions to use.

Level,
Value representing the M5 laid on its back.

Stand,
Value representing the M5 laid on its edge.

Left Tilt,
Returns a value when tiled left.

Right Tilt,
Returns a value when tiled right.

Other Side,
Returns a value when placed face down.

Exploration of M5Stack's Units.

The M5Stack system has a range of small add-ons that can be plugged in via the grove ports, these little add-ons are called units and can be used to increase the activities that can be done with the M5 units. Not all M5Stack core units are made the same which means that some of the add-on units will work not work with all of them.
These incompatible units will have the **Not Compatible in M5Stick and M5Camera** line at the top of their descriptions.

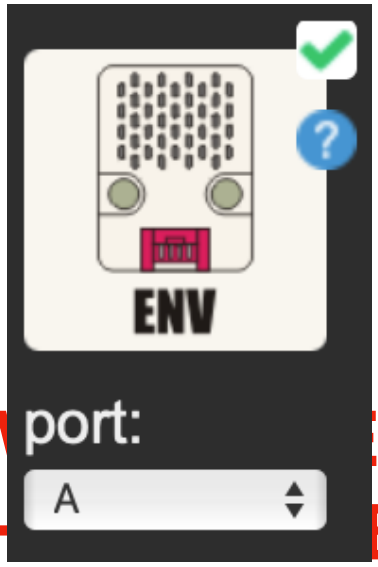
Dependent on which core model you have, you may have one or more grove ports. On the Core and Stick models with only one port, you will need to select Port A when adding units.

As of writing, the current list of units are.

Environmental,
Angle,
PIR,
Neopixel,
Joystick,
Light,
Earth,
Make,
Servo,
Weight,
Tracker,
Button,
Duel Button,
RGB,
Relay,
Heart,
ADC,
Colour,
DAC,
IR,
NCIR,
Thermal,
TOF,
M5 Camera,

As of version UIFlow 1.1.2 the heart sensor, colour sensor and thermal sensor are not usable.

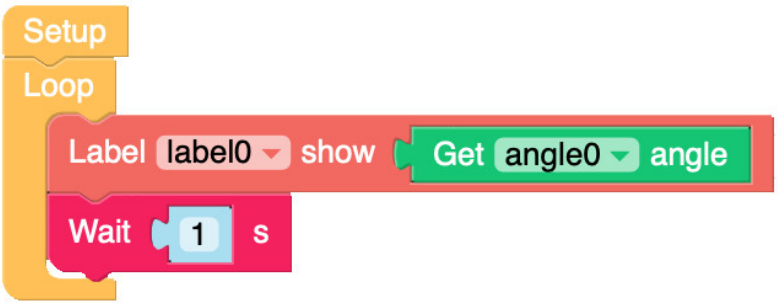
Environmental,



The environmental sensor uses the DHT12 and BMP280 sensors to record temperature, air pressure and, humidity. Because the sensor communicates over I2C it should work with both the M5Stack and M5Stick.

Angle Sensor.

The Angle sensor uses a variable resistor or potentiometer to send an analogue signal to the M5 units which the M5's can translate into an angle of rotation. In UIFlow we use this code to get the value and display it on screen.



However, this does not work on the M5Stack Core because it only has an I2C port. to use it on the core we need to use some extra wires to connect it to the base plate.

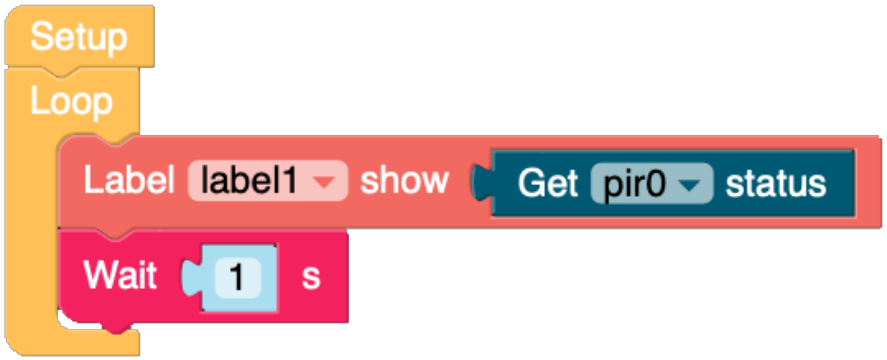
And then by replacing "Get angle0 angle" with analog read pin 36, we can get the angle value.

On the M5Go and M5Fire we connect it to port B and set port B in the drop down box displayed above.

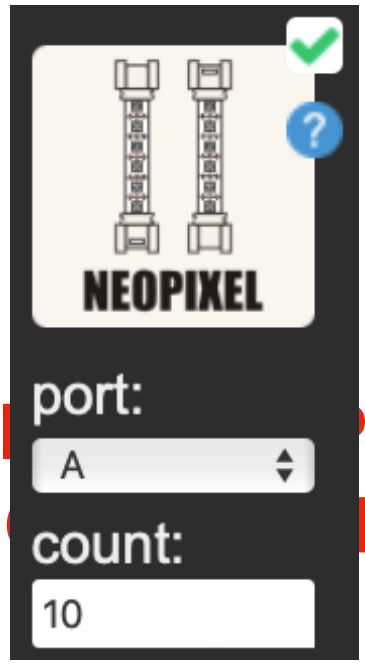
PIR,

The PIR or Passive Infrared Sensor is a sensor that detects invisible Infrared light that is all around us. All living things emit Infrared light and the PIR sensor can be used to detect if a person is in range.
Our UI design is simple in that it uses two labels just like the angle sensor.

Our code is also almost identical to that of the Angle sensor.



Neopixel,



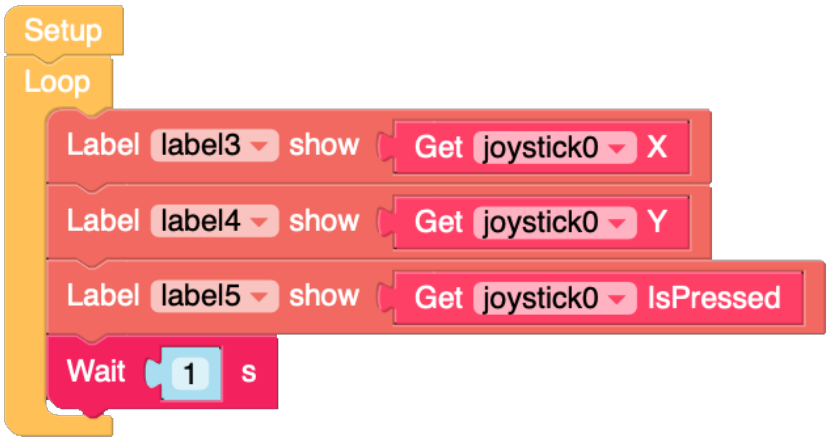
The Neopixel block is used to control WS2128 RGB's know as Neopixels . This unit gives access to the blocks used to control the Neopixel stip hexagon, CatEar, Neoflash, and can be used to control any none M5Stack Neopixel or WS2128 based product. In my Youtube video I used an Adafruit Neopixel Jewel as my strip and hexagon hadn't arrived yet.

The Neohex also has an additional power connector for when there are lots of Neopixels that need power.

Joystick,

The joystick works in a similar way to the Angle sensor but instead of having a single variable resistor sending analogue signals to the M5Stack, the joystick as two variable resistors and if you press down, a third button press is sent. A ATMega328 reads these signals and communicates with the M5Stack over I2C. Our code that we need to read and display these values is almost identical to what we have seen before.

Our code is almost identical except that now we have three values to show.

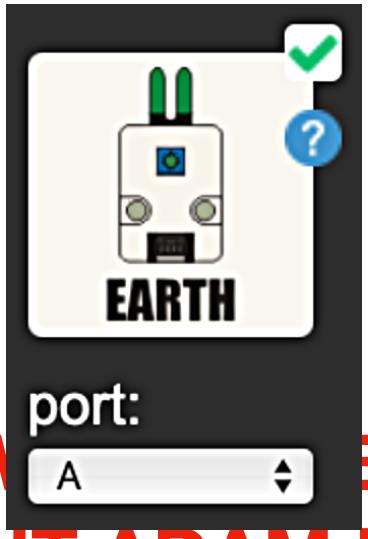


Light,

The light sensor is used to measure light intensity. it has a potentiometer to adjust the detection threshold of the señor which then sends a high or 1 when triggered or Low 0 when not triggered.
To use the Light sensor first need to add the Light Sensor unit below the UI designer and then we create labels in the UI designer, **Analog Value:** and **Digital Value:.**
We then create two place holder labels that will be used to show the sensor values.

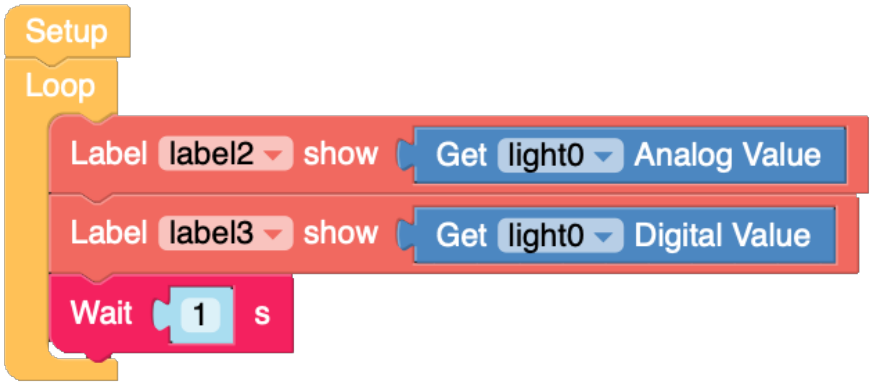
Then we create a loop that reads the values from the sensor and changes the two "Text" labels to show the values.

Earth,

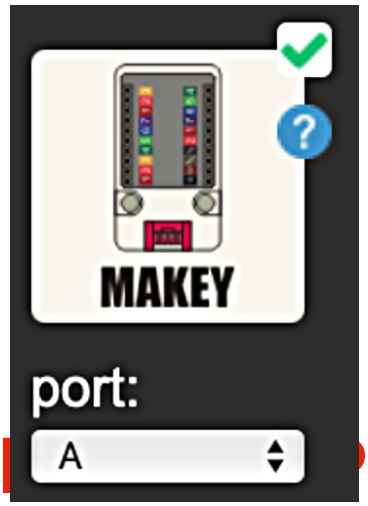


The earth sensor is designed like the light sensor in that it shares the same core electronics. The difference between the Light and Earth sensor is that instead of a Light Dependent resistor, the earth uses water levels between the two probes to create a resistance.

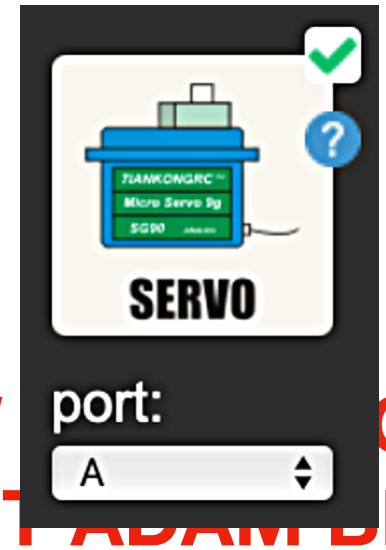
The code that we used for the light sensor can be reused here including the sam UI layout.



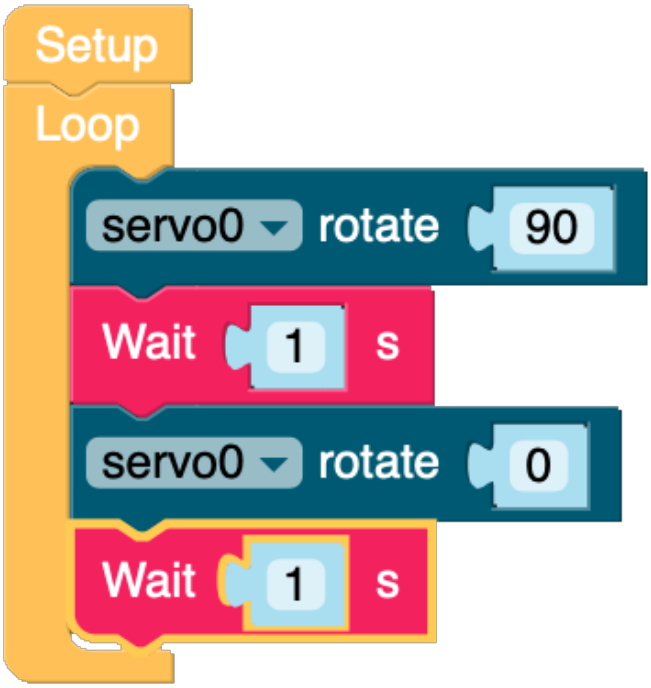
Makey,



Servo,



The Servo unit is not like the other units. Instead of the white box the other units come packaged in, the servo is directed connected to the M5Stacks Grove Connection.
The Servo has two functions available to it, Servo Rotation and Servo Milliseconds.
The following simple loop sets the servo to ninety degrees, waits one second and then sets it back to zero degrees.



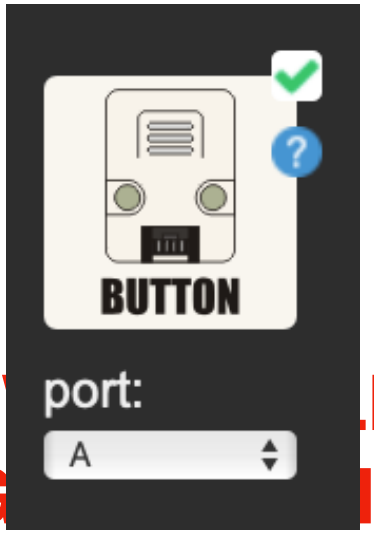
Weight,

The weight sensor is an adapter that allows HX711 weight sensors to be connected to the M5Stack.

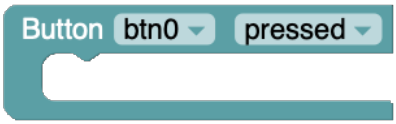
Tracker,

Button

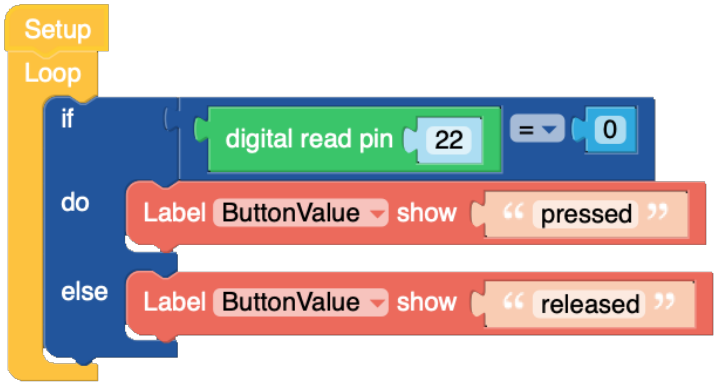
Not Compatible with M5Stick and M5Camera.



The Button Unit provides code just for interfacing with the single button module. this contains a single normally open push button.
There are two code blocks available to the button, one is a loop that monitors the status of the button and the other is an action that is used in other functions.



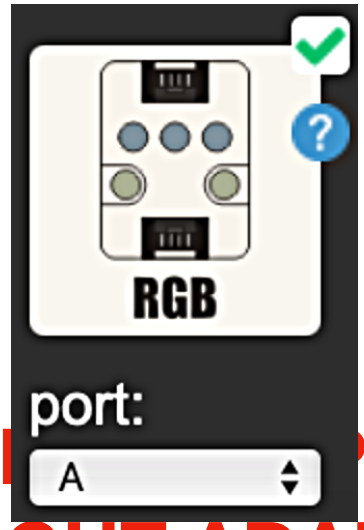
The following code example doesn't use the button blocks due to issues with the grove port on the M5stack Core.



Duel Button,

Ware as the single button only uses one data pin on the "Grove" connector, the duel button uses both data pins of the connector.

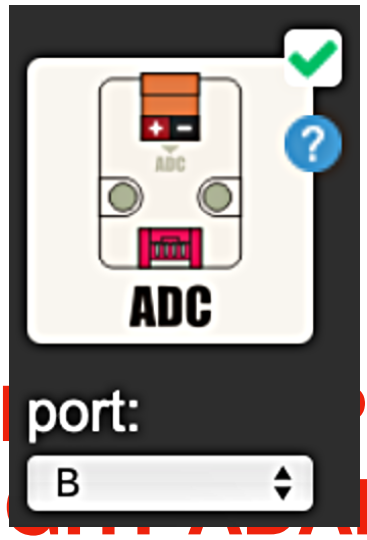
RGB,



The RGB port uses the same WS2812 Neopixel but only has three of them. The port on the top allows you to connect more RGB and Neopixel units together.

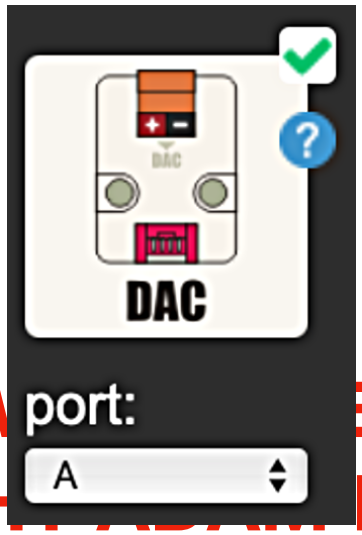
Relay,

Analogue to Digital Converter
ADC



I2C Address = 0x48

Digital to Analogue Converter
DAC

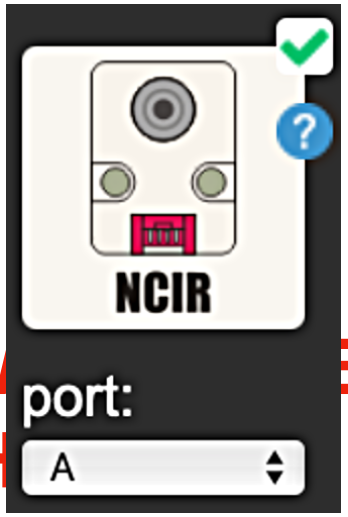


I2C Address = 0x60

Infrared Sensor
IR

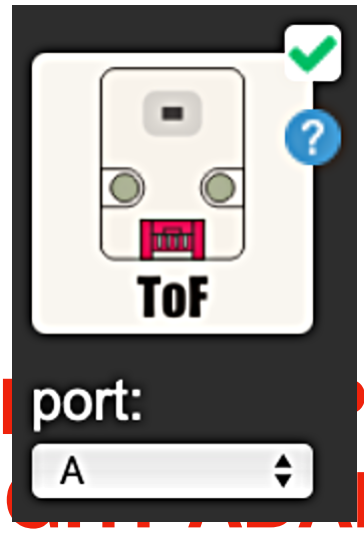
The IR unit uses a pair of IR LED's to send and receive information or act as an avoidance sensor for robots.

Non Contact InfraRed Sensor.
N.C.I.R



Non Contact InfraRed Sensor or N.C.I.R is an infrared sensor for non contact temperature measurement.
I2C Address = 0x5A

Time of Flight Sensor
T.O.F,



The T.O.F sensor or time of flight sensor is a high precision sensor that uses 950nm lasers to measure distance.

I2C Address = 0x29

GPS Unit

The GPS unit is built around the AT6558 receiver which is capable of receiving GNSS signals from up to six satellites at a time.

Fingerprint Unit.

The fingerprint unit is built around the FPC1020A capacitive fingerprint reader and recognition chip.

RFID Reader.

The RFID or Radio Frequency Identification is a contactless communication that works on 13.56MHz using the built in MFRC522 Read/Write device to communicate with the M5Stack over I2c.

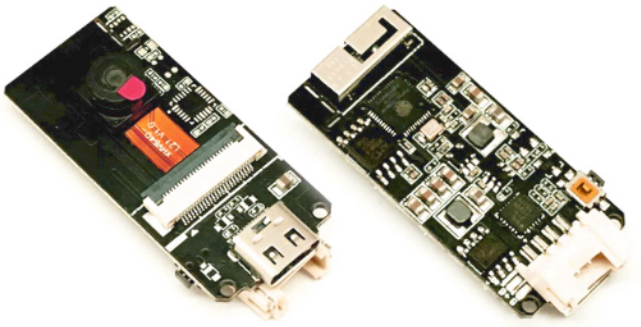
I2C Address = 0x28

Card Keyboard Unit.

The card Keyboard unit is based around the ATMEGA 328
I2C Address = 0x5F

M5 Camera

The M5Camera units consist of two cameras. The Model A camera is available without a case and no PSRam while the Model B has a Black case and uses a PSRam based version of the ESP32.



2019

At present there are no code blocks for these units as they act as stand alone devices.

Going Further.

We are not just restricted to the remade module produced by M5Stack, with some planning and experimenting we can make our own units. We have already seen that the same code can be used with the Light and Earth sensor and so we could misappropriate the function for our own sensors. The Grove system used by M5Stack is identical to the Seeedstudio Grove system and so we can make use of that system to expand our creativity.

To help with inventing our own units we have the following "expansion" units.

Hub unit,
3.96 Unit,
Ext I/O Unit,
A selection of Grove leads and adapters,
Internal expansion connections and a prototype module.

We are not just restricted to these expansions, we can get remade leads that have one end terminated in single Dupont connectors or if we have a soldering iron, we can make our own.

Grove to 4pin Dupont connection.

On thingiverse we can find 3D printable models of the various stackable module cases allowing us to further customise the M5Stack by developing our own internal stackable modules. In the code examples and experiments section I show how to make a version of the angle sensor.

M5Stack's Stackable Modules.

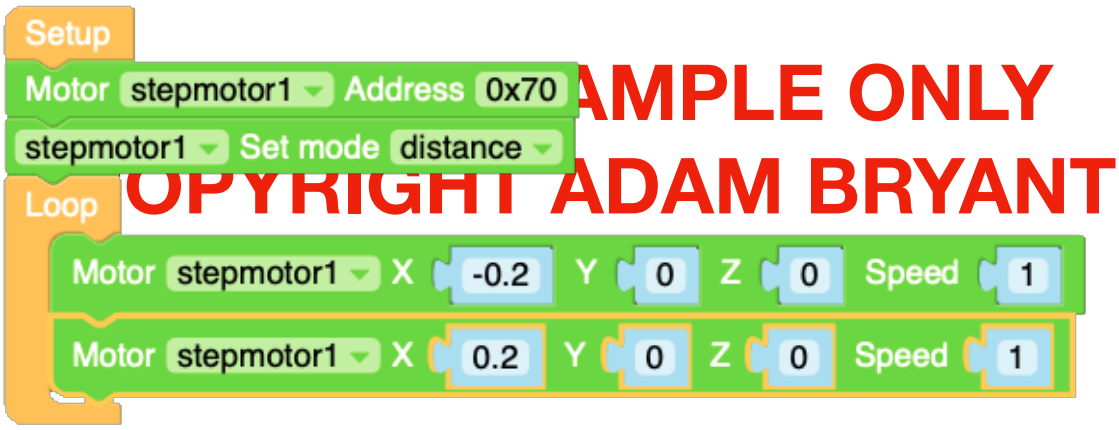
M5Stack has produced a range of prebuilt modules that allow us to do more with the M5Stack. Unlike the units that connect to the M5Stack via the Grove port, the Modules plug into the bottom of the M5Stack core units via the internal 2 row by 15 column internal connector bus.
These stackable Modules are as follows.

Step Motor Module,

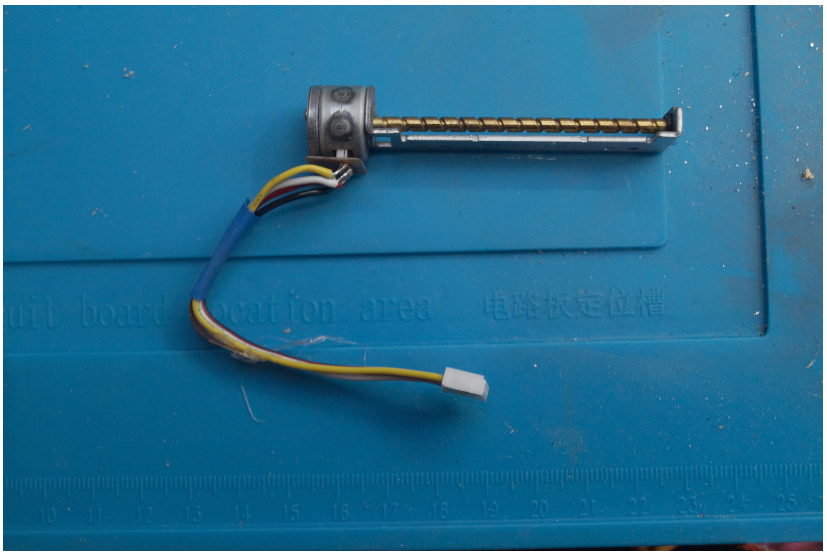
The step motor consist of two modules, the Driver and the cooling fan. The driver modules is built around an ATMega 328 running GRBL to drive three DRV8825 controlled stepper motor outputs.

I2C Address = 0x70

The code below is just a simple program that moves the stepper motor backwards and forwards.



The stepper motor I have been using to test my code is taken from an optical CD/DVD drive but, Nema 17 stepper motors can be run on this module.



Servo Module,

Bala Module,

The Bala module consist of a base unit and two motors for building a self balancing robot. It communicates over I2C with the M5Stack thanks to its ATMEGA 328 which also controls the hardware in the base.

I2C Address = 0x56

Proto Module,

The Proto Module is an unpopulated PCB (With the exception of the internal bus connections,) designed to allow the creation of custom internal stackable modules.

Lego Module,

Commu Module,

Upgraded Battery Module,

The Battery module allows you to easily add an additional 850mAH battery to the M5Stack modules.

FLIR Module,

Plus Base Module,

The Plus Base Module adds an additional 500mAH battery, a rotary encoder, an IR transmitter and an optional PDM mike controlled by a built in ATMEGA 328.
The module also brakes out Ports B and C found on the M5Go base module.

GPS Module,

Lora Module,

Lora Wan Module,

PLC Module,

USB Module,

RFID Go Base,

M5 Bottom Plate

Found on the M5Stack Core Black and Gray, the M5 Bottom Plate contains a 150mAH battery along with the internal bus connector that is connected to Male and Female dupont connectors around the circumference of the base.

Faces Modules

Along with the Units and Modules, the M5Stack system also has a third system called faces. The difference with the Faces system over the rest of the modules is their more finished industrial looking style.

The core of the Faces consist of the M5Core Gray, Faces Base and, the Faces charger The modules available to the faces system so far consist of:

Gameboy Face,
Calculator Face,
Keyboard Face,
Encoder Face,
Joystick Face,

Variables,

Create Variable,
Set Variable,
Change Variable,
Variable,

Maths,

Value,
Calculation,
Pie,
Remainder of,
Condition is even,
Sum of list,
Random Fraction,
Random Integra,
Round,
Square Root,
Sin,
Convert to int,
Convert to Float,

Loops,

Repeat,
Repeat while Condition,
For Each,
Count with,
Break out,

Logic,

If - Do,
If - Else - Do,
Condition True,
Condition Equals Condition,
Not,
Null,
Condition and Condition,
Test - True/False

Graphic,

LCD Clear,
LCD Fill Colour,
LCD Print Text Font,
LCD Pixel Colour,
LCD Line,
LCD Rectangle,
LCD Triangle,
LCD Circle,
LCD Ellipse,
LCD Arc,
LCD Polygon,

Emoji,

Emoji Map,
Set Line Colour,
Change Background Image,

Timer,

Wait,

Function,

Do Something,
Do Something and Return Condition,
If Condition Return Condition,

Text,

Lists,

Advanced functions.

The blocks that appear under the Advanced section are more complicated and contain more lower level functions than we have experienced so far. In this section I will show you how to directly access the pins and lower level functions.

The function in this group are as follows.

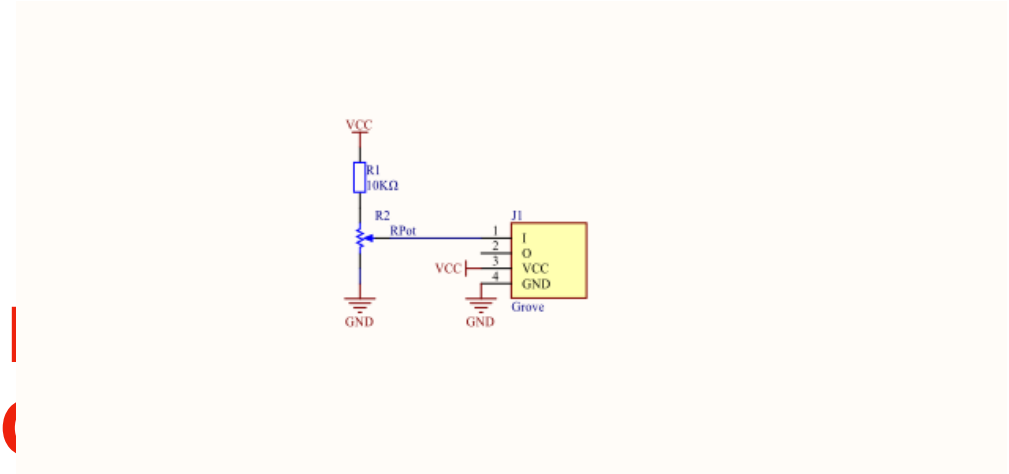
Pin,
  Analog Pin Write,
  Analog Pin Read,
  Digital Pin Read,
  Digital Pin Write,
  Set Pin Mode,
  Map Pin,
GPIO,
  Set Pin Out of Pin,
  Set Pin In,
  Set Pin Out,
  Set Val to Pin In Value
PWM,
  Set PWM of Pin,
  Set PWM Frequency,
  Set PWM Duty Cycle,
ADC,
  Set ADC Pin to Value,
  Set Item,
  Set ADC Pin Bit Width,
  Read ADC Pin,
DAC,
  Set Dac,
  Write Value to Dac,
UART,
  Set Uart,
  Read Uart,
  Read Uart (0) Characters,
  Read a Line of Uart,
  Read Uart and Write to Buffer,
  Read Uart Status?,
  Write Value to Uart,
I2C,
  Set I2C in Port,
  I2C Scan Available,
  I2C Address (0) is Ready,
  I2C Read Address (0) count (0),
  I2C Read Address (0) Reg (0) Count (0),
  I2C Read Address (0) One Byte,
  I2C Read Address (0) Reg (0) One Byte,
Execute,
Execute Code,
Network,
WIFI Connect,
MQTT,
  MQTT Setup,
  MQTT Subscribe,
  MQTT Start,
  Get Topic Data,
  Publish Topic,
Remote,

Remote Qr Code Show,
Add Remote Switch,
Add Remote Button,
Add Remote Slider,
Add Remote Other,

Code Examples and experiments.

Home Made Angle sensor.

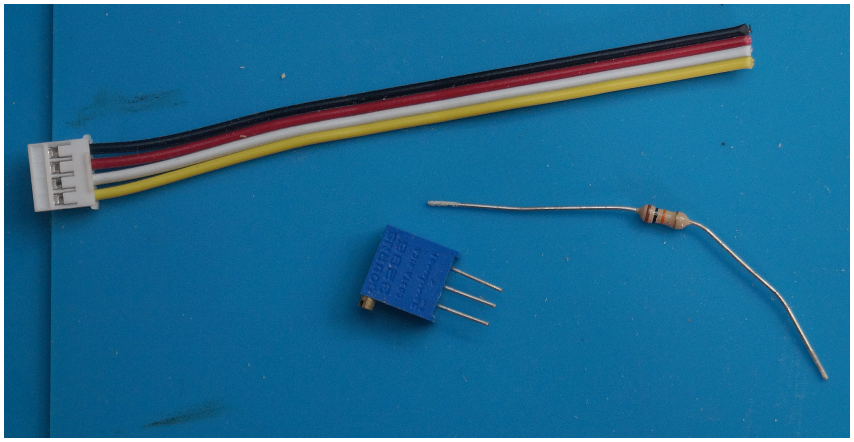
In this simple project I will show you how to make a simple angle sensor based on M5Stacks own sensor as its a simple circuit to clone.
M5Stack provide the following schematic diagram in their documentation of the sensor.



The rectangle R1 is a fixed value resistor of 10K ohm.
R2, the variable resistor is not given a value. To work out a value that may be appropriate I looked at the Button Units document and found that it used a 20K ohm for R2 between D and GND. Looking through my parts bins, I found a 20K ohm resistor (the blue thing with three wires,) and soldered it all together using the following guide.

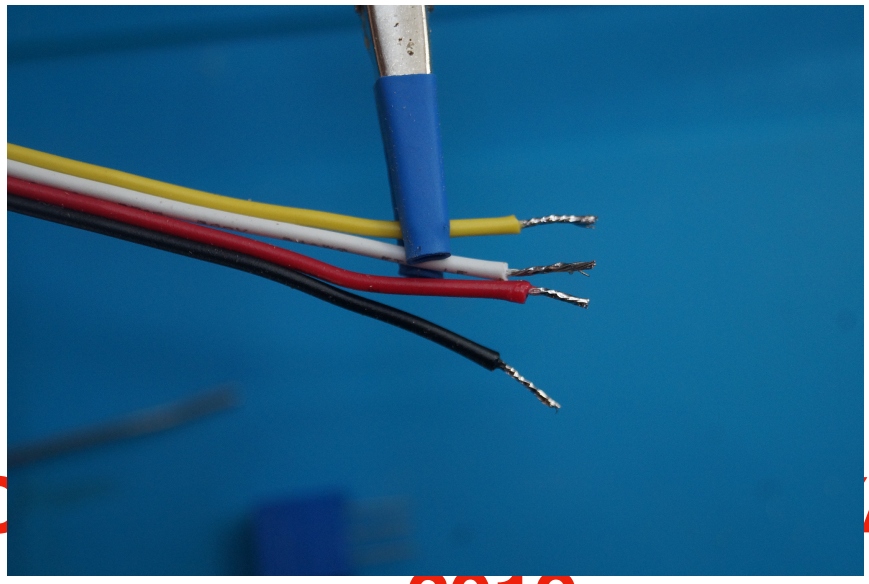
For this project you will need the following.

A Grove lead,
A piece of PCB or breadboard (not shown),
1 X 10K Resistor,
1 X 20K Variable Resistor,

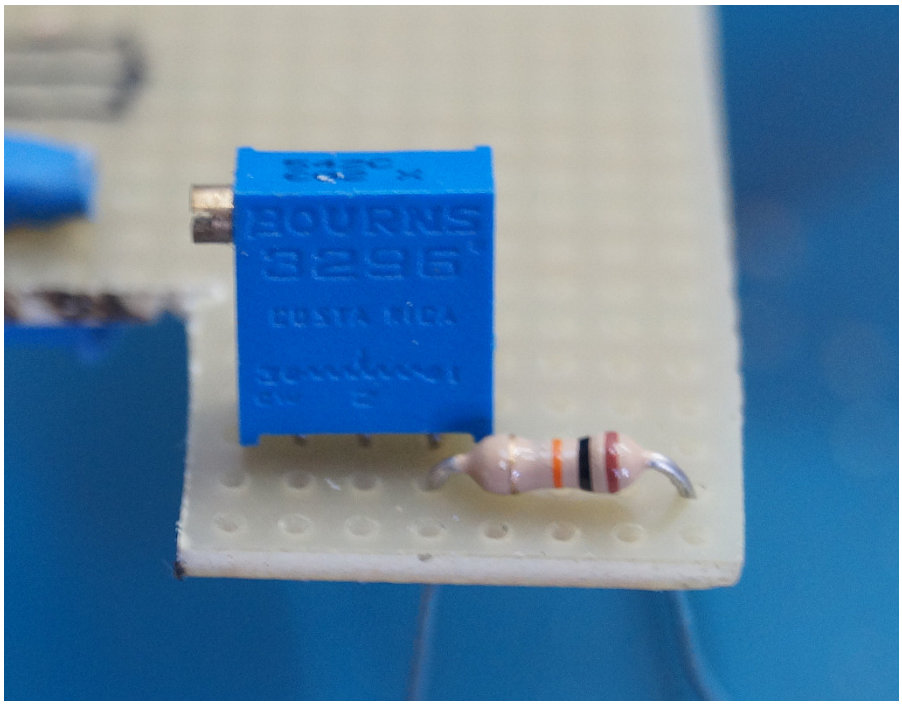


Step 1
Take the grove lead and chop in half, but don't remove the white connectors.

Step 2
Strip the ends and tin the conductors with solder.



Step 3
Lay out the parts on the PCB so that one end of the resistor is connected to one end of the potentiometer.

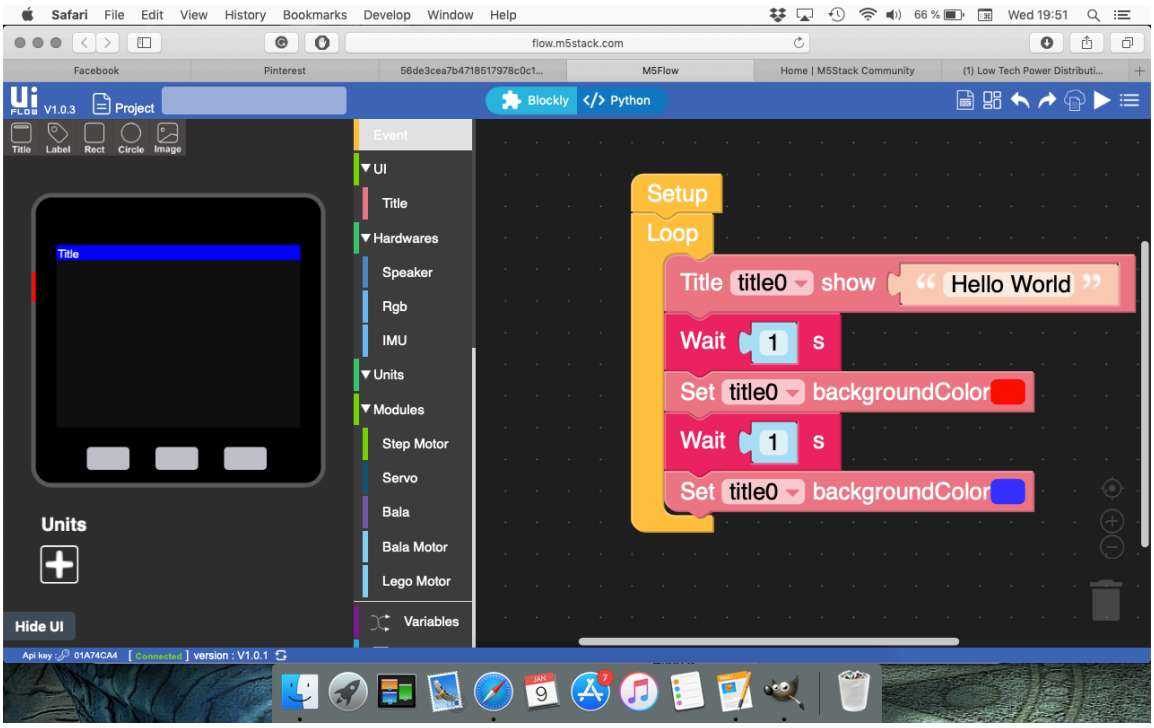


Step 4
Solder the components and lead to the PCB

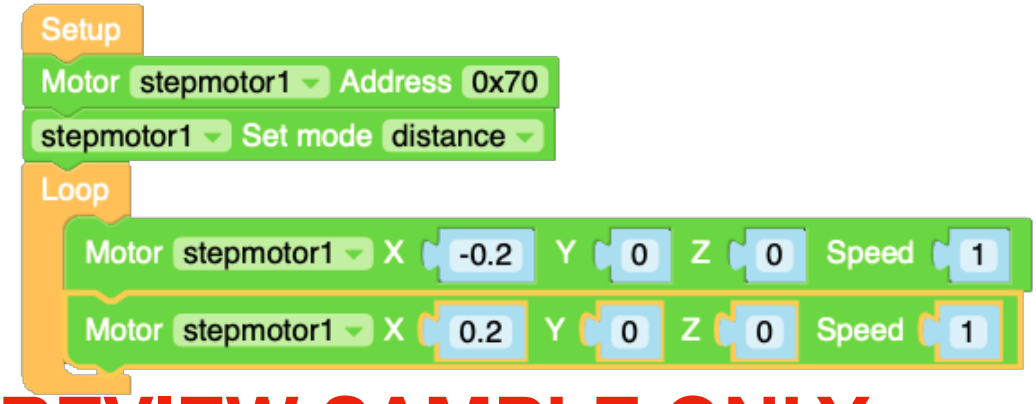
Hello World

Below is a simple script that prints "Hello World" on the screen and flashes the title bars background between red and blue.

Of course this doesn't work on its own, First we need to add a title bar element to the virtual M5Stack and then the Title blocks we become available in the UI list.



Stepper Motor test code.



This test code uses the Stepper Module with an optical driver stepper motor connected to the "X" port.

For this you will need the following,

M5 Core Module,
Stepper Motor Module,
Cooling fan Module,
Optical Drive Stepper motor,
Sensor cable.

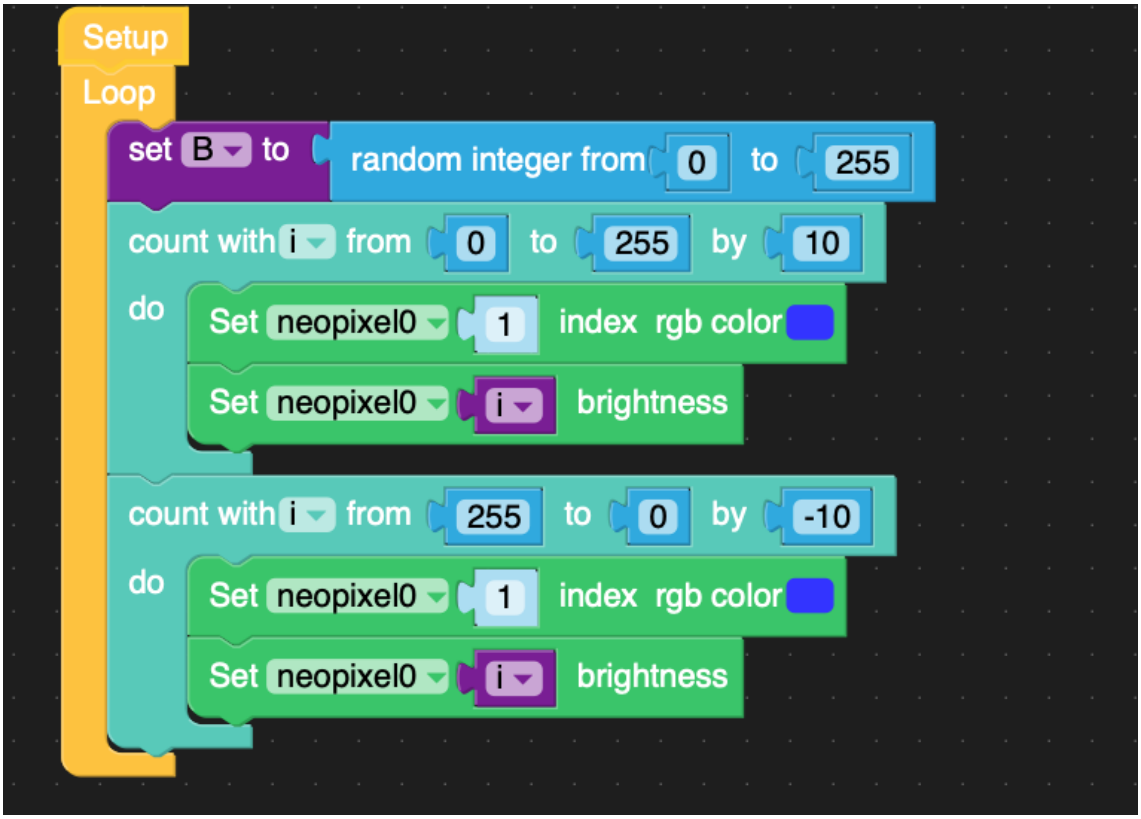
Step 1
Strip the optical drive until you have just the metal carrier, the stepper motor and the laser head carriage.

Step 2
Check the motors pin out! - Chop the 4 pin sensor lead halfway between the connection to make two four wire leads. **DO NOT** chop off the connectors!

Step 3
Strip the four wires and carefully solder to each of the four motor connections, being careful of the motor coils pins. In the picture below I have removed the stepper motor from the movement tray in an attempt to get a better view.

Step 4
Connect the modules together so that the fan module is between the baseplate and the stepper module.
Step 5
You need to make a two wire lead with the provided XT30 connection on one end to plug into the Stepper Module and the other terminated to fit your power source (min had a barrel jack to connect to my solar charge controller.
Step 6
Connect the stepper motor to to the "X" motor port.
Step 7
Power on the M5Stack and connect it to UIFlow.
Step 8
Build the above code in UIFlow and Download to the M5Stack (while it can be run from the cloud, you may have problems if you haven't got a wifi connection.)

Breathing Neopixel test.



This example is based on the Breathing LED example from the built in programs and uses an Adafruit Neopixel Jewel plugged into the grove connection (port A on devices with multiple grove ports), however the M5 Neopixel Strip also works. This example also show an error which is ignored without stopping the program from running.

For this you will need the following,

M5 Core Module,
Grove lead,
Neopixel strip (hexagon will not work in this example)

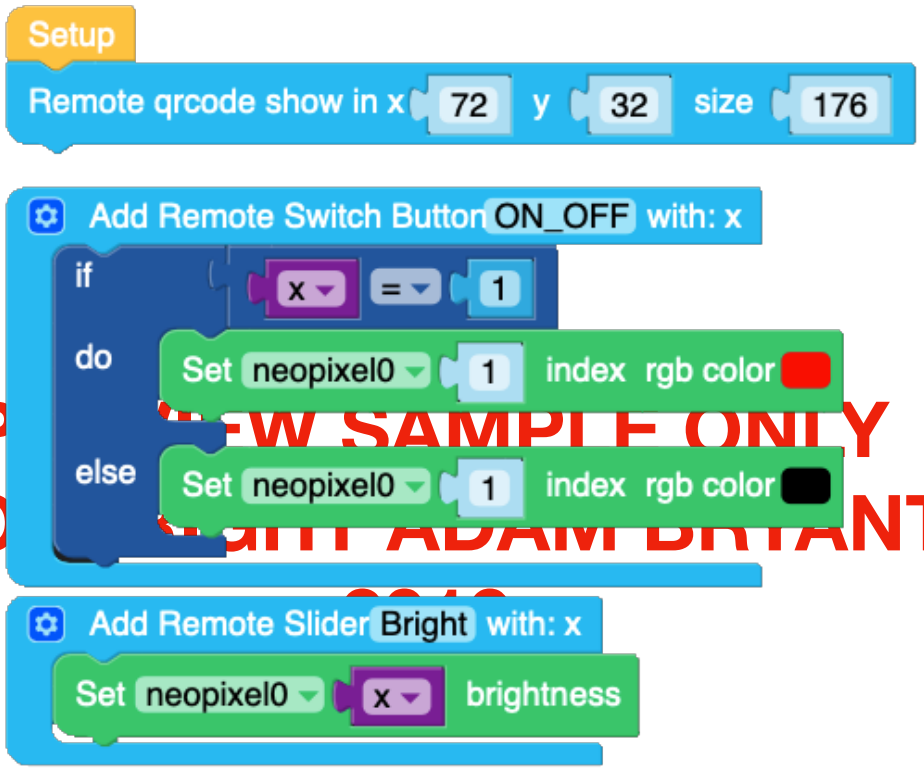
Step 1
Connect one end of the Grove lead to the Neopixel strip,
Step 2
Plug the other end of the grove lead into the M5Core.
Step 3
Power on the M5Stack and connect it to UIFlow.
Step 4
Build the above code in UIFlow and Download to the M5Stack (while it can be run from the cloud, you may have problems if you haven't got a wifi connection.)

How the code works

First we need to add the Neopixel unit. Set the port to A and the No of Pixel to the number on the strip then hit OK to add it to the project. We add the yellow "Loop" block and then we create a variable B and assign it a random integer from 0 to 255.
Next we use a "Count Loop" to increase our variable integer from o to 255 each time the count loop cycles. We then need to use a second "Count Loop" to dim the brightness back

to 0. We drag the Neopixel index block out next and add that to both of of the count loops and set the colour to blue. After each Neopixel index block we add a Neopixel Brightness block and add our variable to the value condition space. When run, first count loop increases the brightness and colour of the Neopixel in the blue channel until it reaches 255, when it hits 255, it finishes the first count loop and steps into the second loop which works in the opposit direction, decreasing the brightness. Once it reaches 0 the main loop takes over and restarts the counting up.

Remote controlled Neopixel

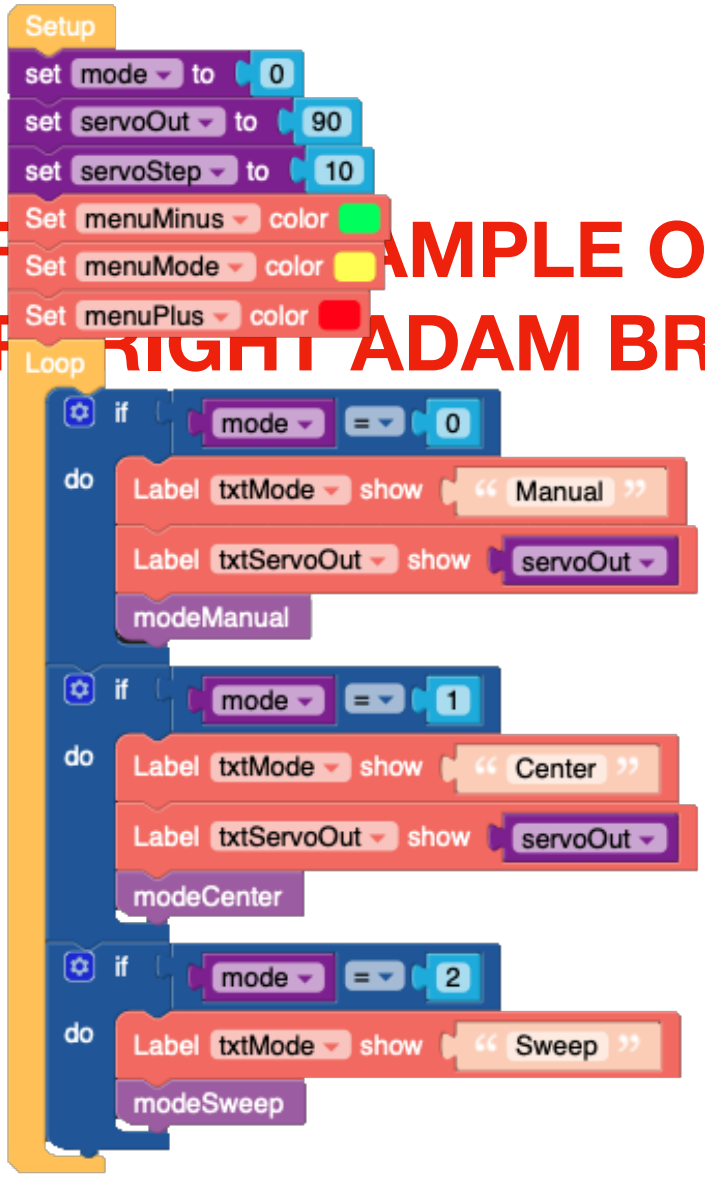


This example is based upon the example program in UIFlow to control the M5Fire's RGB LED strips.
I have replaced the code blocks that operated the RGB LEDs and replaced them with Neopixel code blocks.

Pattiuak's Servo Tester.

This code was created by Forum member Pattiuak and is reproduced here with his permission.

The Main program

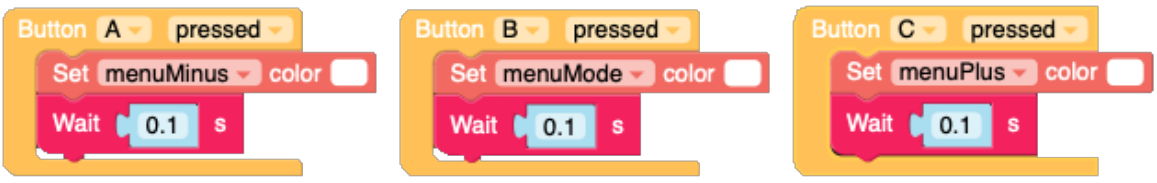


The main program creates three variables **mode**, **servoOut** and **servoStep**. It then sets the **mode** variable a value of 0, the **servoOut** variable a value of 90 and the **servoStep** a value of 10. Once done, the code moves on to set the colours of the label **menuMinus** to green, **menuMode** to yellow and **menuPlus** to red before moving onto the main loop.
The main loop contains three if do loops which are used to create three operating mode options and tell the main program to move to the smaller program loops of **modeManual**, **modeCentre** and **modeSweep**.

The ModeSweep Loop

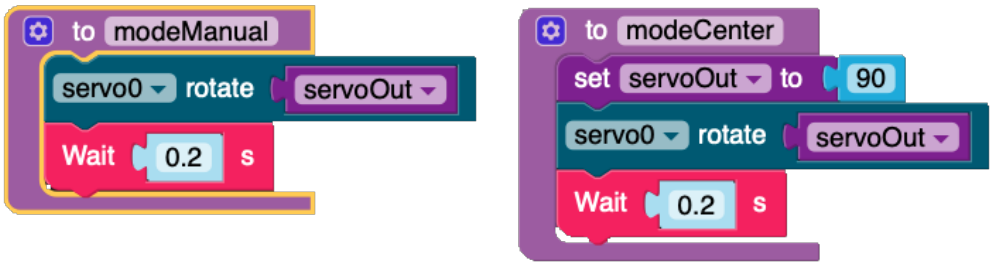
The modeSweep moves the servo to ninety degrees, one hundred and eighty degrees, then back to ninety degrees and then back to zero degrees while changing the txtServoOut table to show where the servo is moving to.

The Button Press Loops.



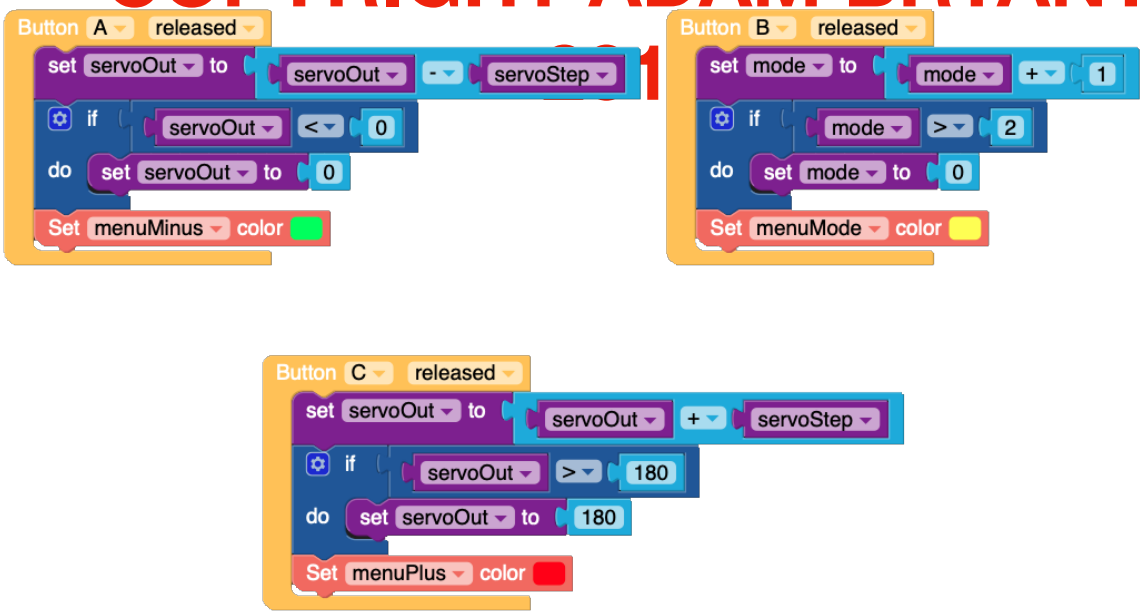
The Button Press Loops all do the same thing, they wait for one of the three buttons to be pressed, change the menu mode colour to white and then wait for 0.1 seconds before returning back to the waiting for a button press.

Mode Loop and Mode Centre Loop



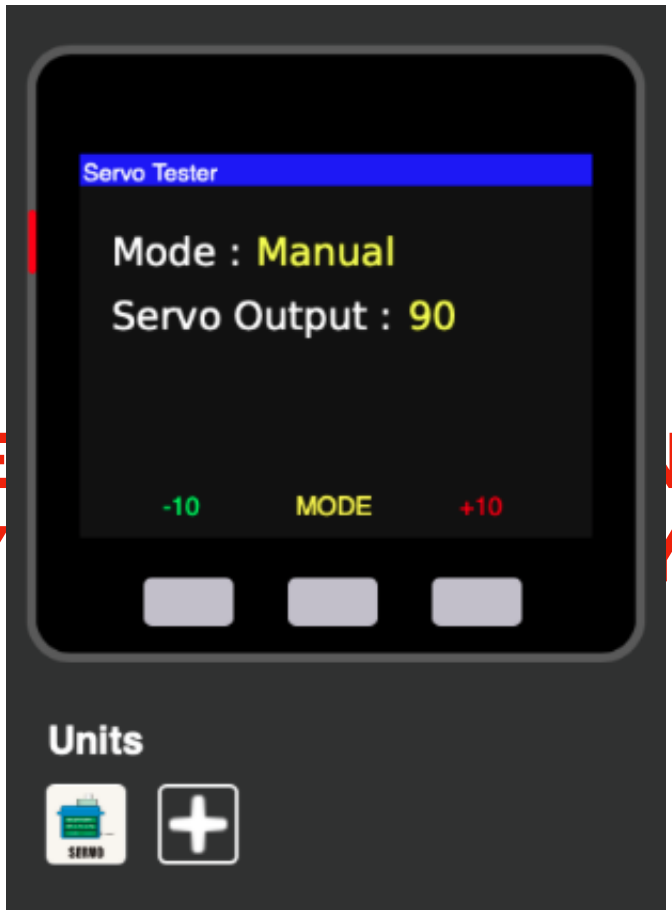
These two loops are almost the same except for mode centre starts by setting the servo to ninety degrees which is the centre of rotation for our servo.

The Button Loops.



Our last bit of code contains the loops for the buttons. Button A decrease the servo position and Button C increase the servo rotation. Our last button, button B is used to change between the different movement modes in our program.

The UI Design.



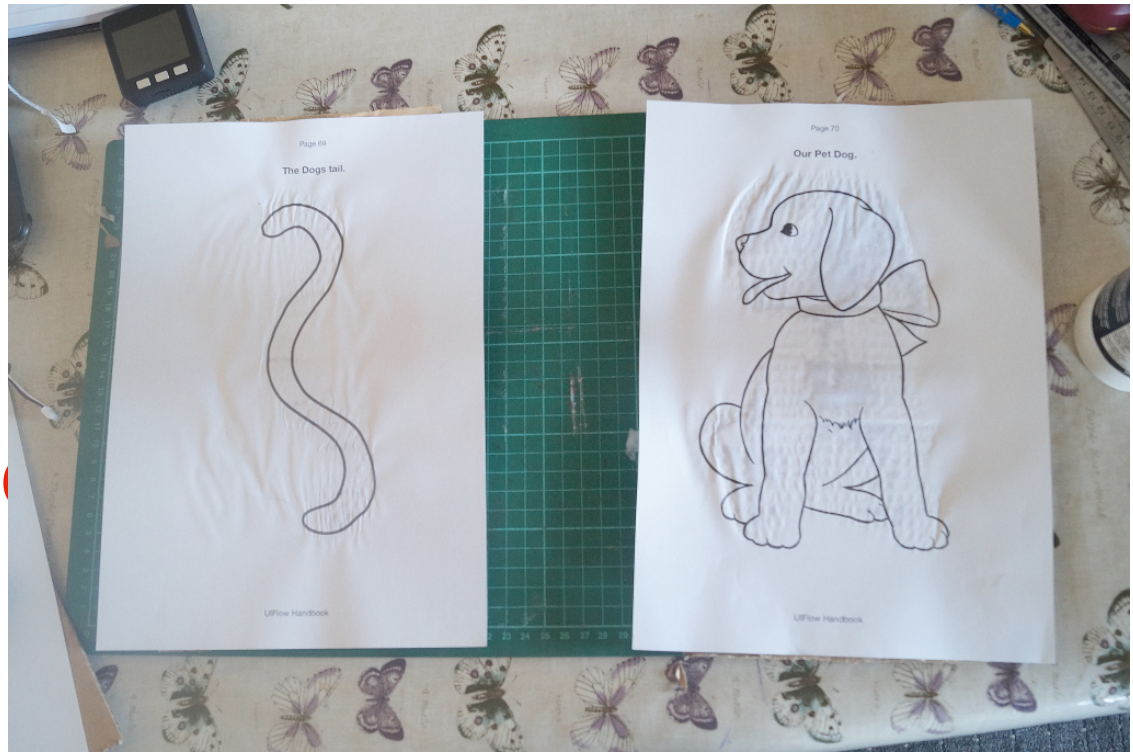
The User Interface design is quite simple and contains a title bar and labels.

A Dog with a Wagging Tail

In this example we will make a simple picture of a dog with a wagging tail.

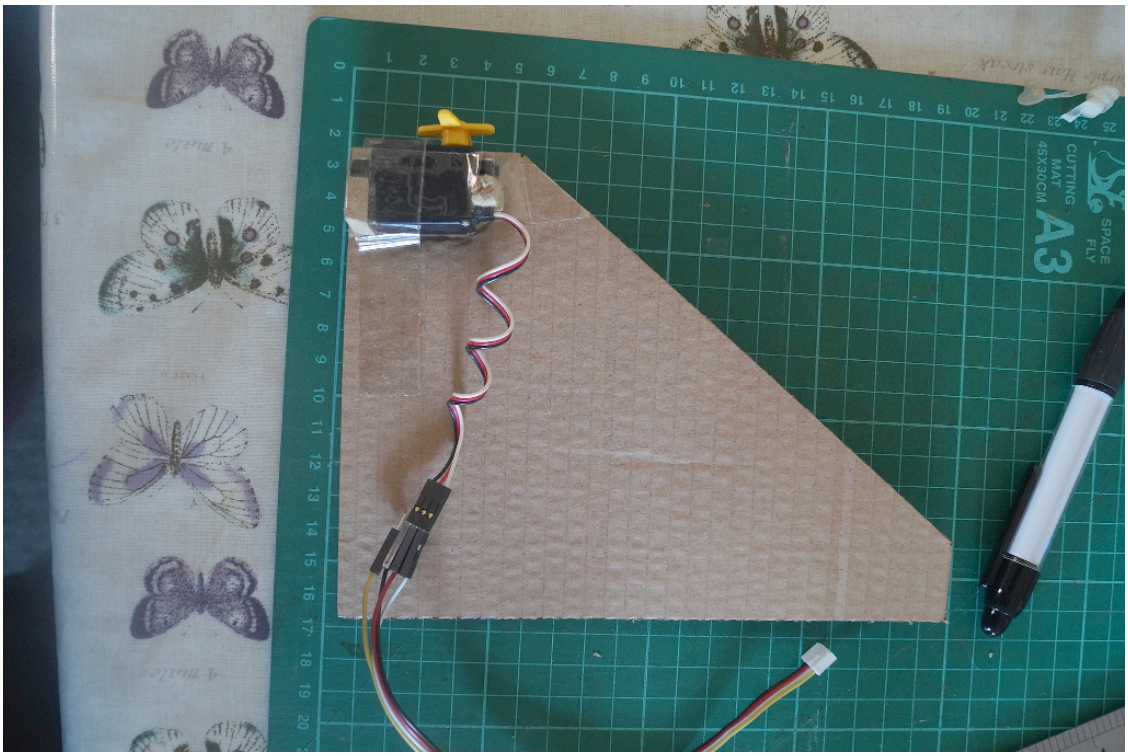
For this we will need.
M5Stack,
A servo,
Thick Cardboard,
Glue,
Scissors,
Print outs of the included dog pictures.

Step 1
Print out our pet dog and his tail and glue to separate pieces of card and cut out our dog and his tail when the glue has dried.



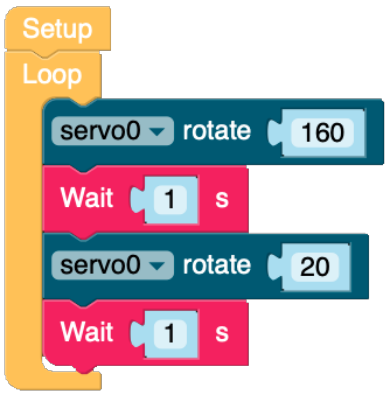
Step 2
Colour in our dog if desired.

Step 3
Cut out a rough triangle of card to act as a stand and to support a servo and glue or tape the Servo to the stand with the horn and cable towards the top.



Step 4
Glue or tape the stand to the back of our pet dog.
Step 5
Glue or tape the tail to the servo horn.

Step 6
Connect the servo to the M5Stack.
Step 7
Build the code below and download to the M5Stack.

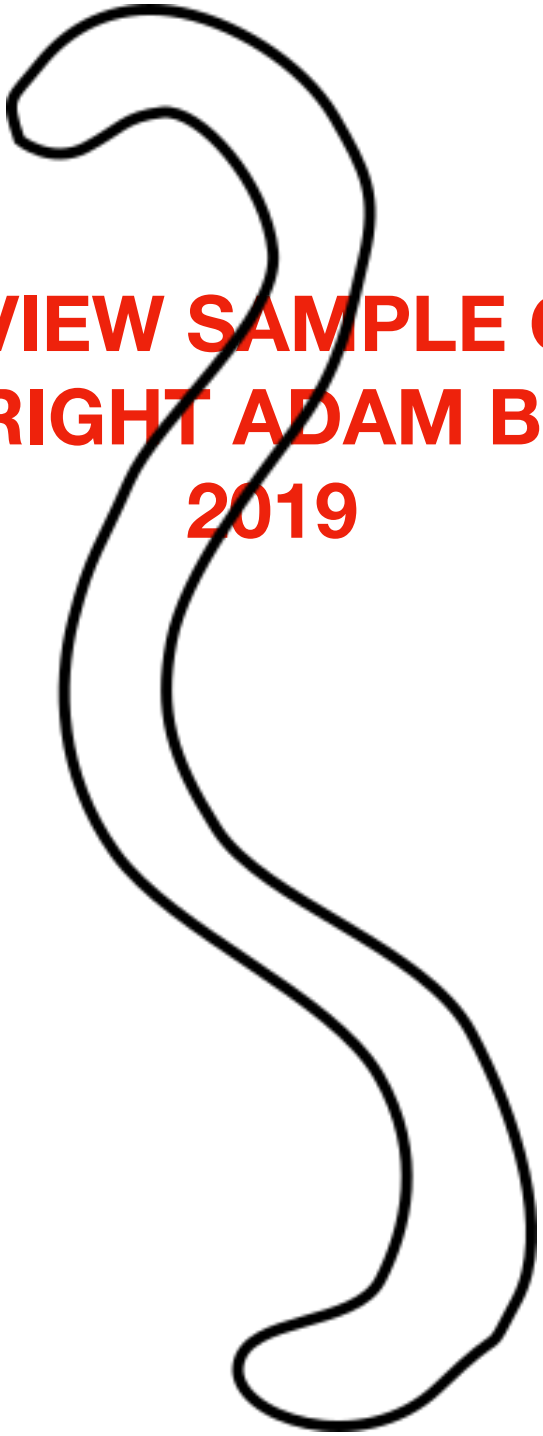


Step 9
Sit back and watch our pet dog wag his tail.

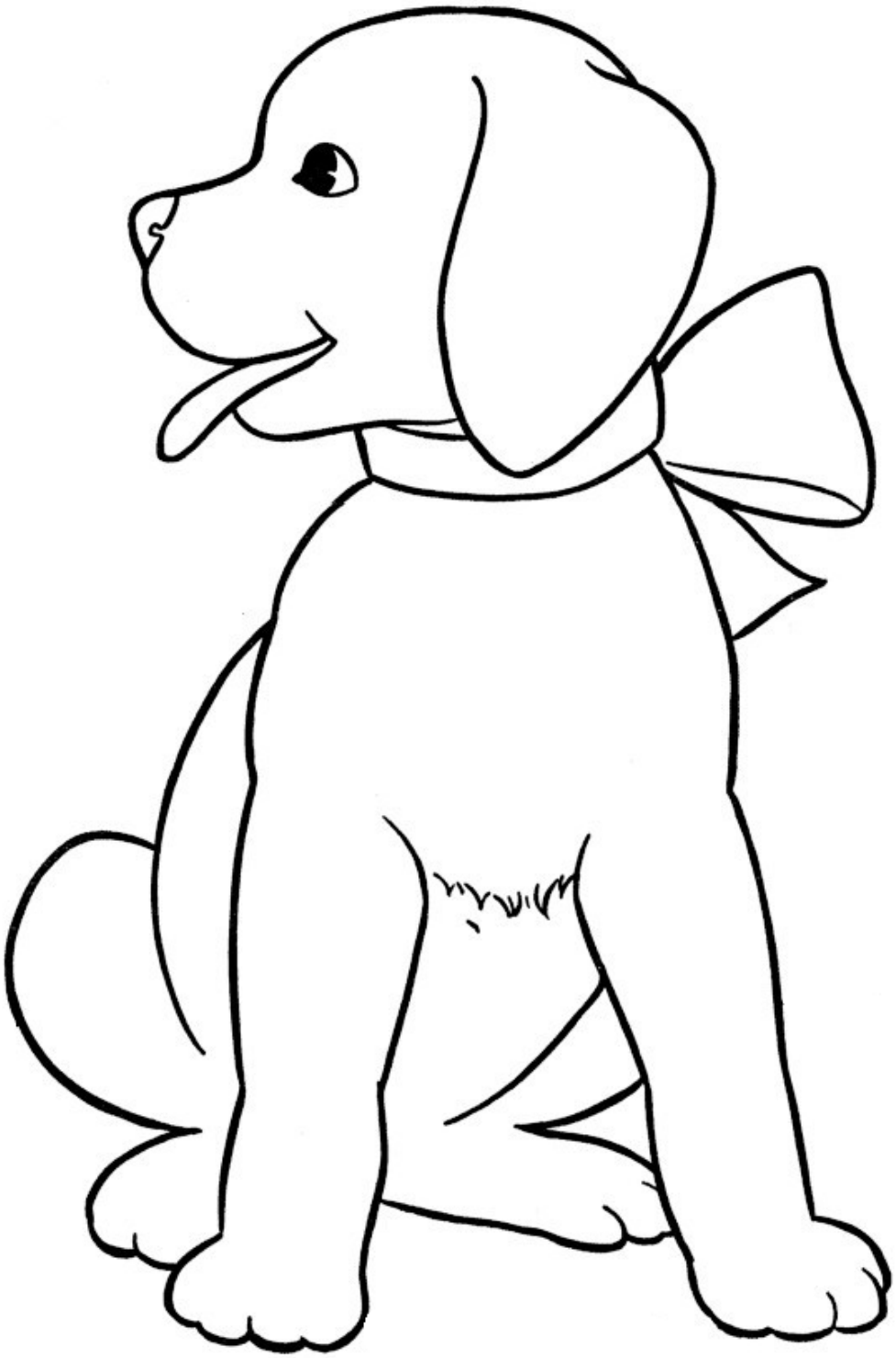
How the code works.

The code has be designed to be very simple to understand. All it contains is a main loop which moves the servo between zero and one hundred and eighty degrees, waits a second and then moves it back.
To speed up the tail wagging we just decrease the **wait** time.
To change the position of the wagging, we just change the **Servo Rotate** values.

The Dogs tail.



Our Pet Dog.



I2C Device Quick Reference list.

A/D Converter	= 0x48
Bala Base	= 0x56
Card keyboard	= 0x5F
D/A Converter	= 0x60
Lego	= 0x56
N.C.I.R	= 0x5A
Plus Unit	= 0x62
RFID Unit	= 0x28
Servo	= 0x53
Step Motor	= 0x70
Time of Flight	= 0x29

Index and Roadmap of Functions.

Introduction,
Colour Setting,
Setup,
Download or Test,
Exploring the Blocks.
Event,
  Loop,
  Button,
  Obtain Button,
U.I. (user Interface)

Hardwares,
  Speaker,
    Speaker Beep,
    Speaker Volume,
    Play Tone.
  RGB,
    Set RGB Bar Colour,
    Set RGB Bar Colour R,G,B,
    Set (Side) RGB Bar Colour,
    Set (Side) RGB Bar Colour R,G,B,
    Set (individual) RGB Bar Colour,
    Set (individual) RGB Bar Colour R,G,B,
    Set RGB Brightness,
  IMU,
    Get X,
    Get Y,
    Get Z,
    Level,
    Stand,
    Left Tilt,
    Right Tilt,
    Other Side,
Units,
  Environmental,
    Get Pressure,
    Get Temperature,
    Get Humidity,
  Angle,
    Get Angle,
  PIR,
    Get PIR Status,
  Neopixel,
    Set individual Neopixel to colour,
    Set Neopixel Range to colour,
    Set Neopixel Range to RGB colour,
    Set all Neopixel Colour,
    Set Neopoxel Brightness,
    Set Neopixel Hex Colour,
    Set Neopixel Hex (Variable) Colour,
  Joystick,
    Get X,
    Get Y,
    Is Pressed,
  Light,
    Get Light Analogue Value,
    Get Light Digital Value,
  Earth,
    Get Earth Analogue Value,
    Get Earth Digital Value,
Makey,
  Get Value,
  Get all Value,
  Start Piano Mode,
Servo,
  Servo Rotate,
  Servo Speed,
Weight,
  Zero Weight,
  Get Weight,
Tracker,
  Get Analogue Value,
  Get Digital Value,
  Constrain limits,
Button,
  Button Loop,
  Button Action,
Duel Button,
  Button Loop,
  Button Action,
RGB,
  Set RGB Bar Colour,
  Set RGB Bar Colour R,G,B,
  Set RGB all Colour,
  Set RGB Brightness,
Relay,
  Set Relay On,
  Set Relay Off,
Heart Unavailable,
ADC,
  ADC Read,
Colour Unavailable,
DAC,
  Set DAC,
IR,
  Get IR State,
  Set IR On,
  Set IR Off,
NCIR,
  NCIR Read,
Thermal Unavailable,
TOF,
  Get Distance,
EXT I/O,
  Set I/O Mode,
  Digital Write Variable,

Digital Write State,
Digital Read,
Digital Read Port,

Modules,
Stepper Motor,
Stepper Motor Module Address,
Stepper Motor Position,
Run "G" Code,
Stepper Motor Movement Mode,
Lock Stepper Motor,
Unlock Stepper Motor,
Servo,
Set Servo Position,
Set Servo Speed,
Bala,
Bala Motor,
Lego Motor,

Faces Modules
Gameboy Face,
Calculator Face,
Keyboard Face,
Encoder Face,
Joystick Face,

Variables,
Create Variable,
Set Variable,
Change Variable,
Variable,
Maths,
Value,
Calculation,
Pie,
Remainder of,
Condition is even,
Sum of list,
Random Fraction,
Random Integra,
Round,
Square Root,
Sin,
Convert to int,
Convert to Float,
Loops,
Repeat,
Repeat while Condition,
For Each,
Count with,
Break out,
Logic,
If - Do,
If - Else - Do,
Condition True,
Condition Equals Condition,
Not,
Null,
Condition and Condition,
Test - True/False

Graphic,
LCD Clear,
LCD Fill Colour,
LCD Print Text,
Font,
LCD Pixel Colour,
LCD Line,
LCD Rectangle,
LCD Triangle,
LCD Circle,
LCD Ellipse,
LCD Arc,
LCD Polygon,
Emoji,
Emoji Map,
Set Line Colour,
Change Background Image,
Timer,
Wait,
Function,
Do Something,
Do Something and Return Condition,
If Condition Return Condition,
Text,
Lists,
Advanced,
Pin,
Analog Pin Write,
Analog Pin Read,
Digital Pin Read,
Digital Pin Write,
Set Pin Mode,
Map Pin,
GPIO,
Set Pin Out of Pin,
Set Pin In,
Set Pin Out,
Set Val to Pin In Value
PWM,
Set PWM of Pin,
Set PWM Frequency,
Set PWM Duty Cycle,
ADC,
Set ADC Pin to Value,
Set Item,
Set ADC Pin Bit Width,
Read ADC Pin,
DAC,
Set Dac,
Write Value to Dac,
UART,
Set Uart,
Read Uart,
Read Uart (0) Characters,
Read a Line of Uart,
Read Uart and Write to Buffer,
Read Uart Status?,
Write Value to Uart,
I2C,
Set I2C in Port,

I2C Scan Available,
I2C Address (0) is Ready,
I2C Read Address (0) count (0),
I2C Read Address (0) Reg (0) Count (0),
I2C Read Address (0) One Byte,
I2C Read Address (0) Reg (0) One Byte,
Execute,
Execute Code,
Network,
WIFI Connect,
MQTT,
MQTT Setup,
MQTT Subscribe,
MQTT Start,
Get Topic Data,
Publish Topic,

Remote,
Remote Qr Code Show,
Add Remote Switch,
Add Remote Button,
Add Remote Slider,
Add Remote Other,

Experiments and Sample Code,
Home Made Angle Sensor
Breathing Neopixel,
Remote Controlled Neopixel,
Neopixel Photography Light,
Optical Drive Stepper Motor Test.
Dog with Wagging Tail.

I2C Device Address Quick Reference

Index