meArm.Joystick en

by ted99tw on November 23, 2014

Table of Contents

Author:ted99tw
maker for educational kit since 2014

Intro: MeArm.Joystick en

Origin : Phenoptix's Pocket Sized Robot Arm.

- There are 37 programming lessons, good for beginners and veteran. Students from elementary school can use S4A or Ardublock. There is also lessons available based on C language. One can learn without programming skill.

- Please go through step A, B and C to install necessary software and driver.

- Please go through step D and E to install drag-and-drop style languages. (S4A or Ardublock)

- You can restore factory default any time by uploading meArm.ino any time in Arduino IDE.

- Feel free to download all lessons at a time. (30 lessons.zip , will be available after the end of crowdfunding)

- Restore meArm.Joystick to factory default, please upload meArm.ino in Arduino IDE.

A) Arduino IDE

B) necessary driver

(Windows)

double click Arduino_driver.exe

(MAC)

1) double click ch34xInstall.pkg

2) For Yosemite (OSX 10.10), please open terminal and input below instruction. Reboot after that.

sudo nvram boot-args="kext-dev-mode=1"

(Linux)

Follow the instruction of readme.txt in CH341SER_LINUX.zip

C) setup Arduino IDE

(Windows)

1) System-Device Manager-Ports (CH-340 is is COM3 in this example)

2) Choose "Arduino UNO"

3) Select correct port. (COM3)

4) Select "Arduino as ISP" as programmer

(MAC)

1) Application-Tools-System Info (Plug in meArm.Joystick and ensure "USB2.0-Serial" is there)

2) Choose "Arduino UNO"

3) Select "dev/tty.wchusbserial410"

4) Select "Arduino as ISP" as programmer

D) S4A

1) Download and replace S4A.Image at S4A installation folder.

Windows : C:\Program Files\S4A\S4A.Image

MAC : Macintosh HD/Application/S4A/S4A.Image

2) Open Arduino IDE, upload S4AFirmware15_meArm.ino to meArm.Joystick

E) Ardublock

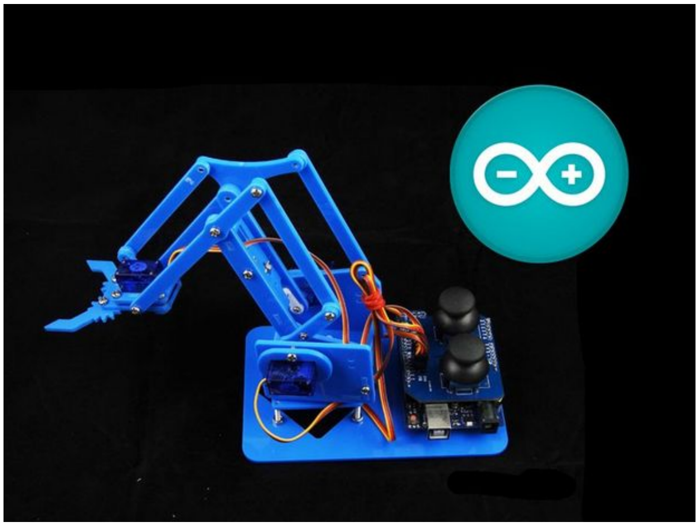
After Arduino installation, please copy "ardublock-beta-20140828.jar" to :

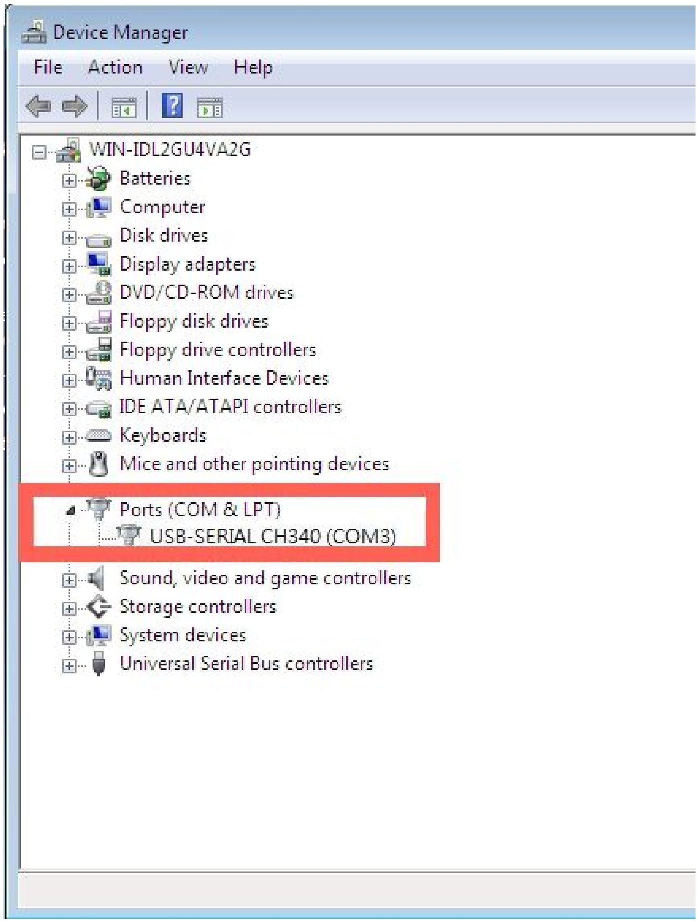
Windows 7 : C:\Users\lienhungcheng\Documents\Arduino\tools\ArduBlockTool\tool\ardublock-beta-20140828.jar

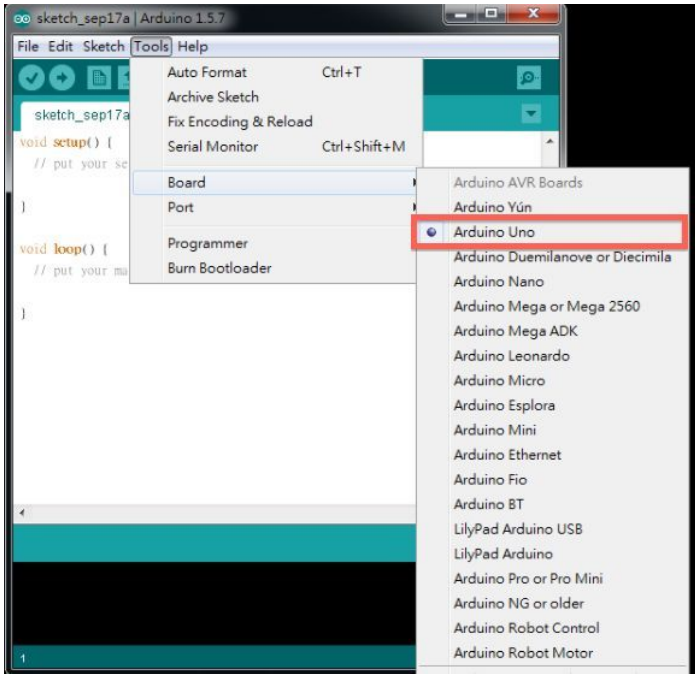
Windows XP : C:\Program Files\Arduino\tools\ArduBlockTool\tool\ardublock-beta-20140828.jar

MAC: /Users/lienhungcheng/Documents/Arduino/tools/ArduBlockTool/tool/ardublock-beta-20140828.jar

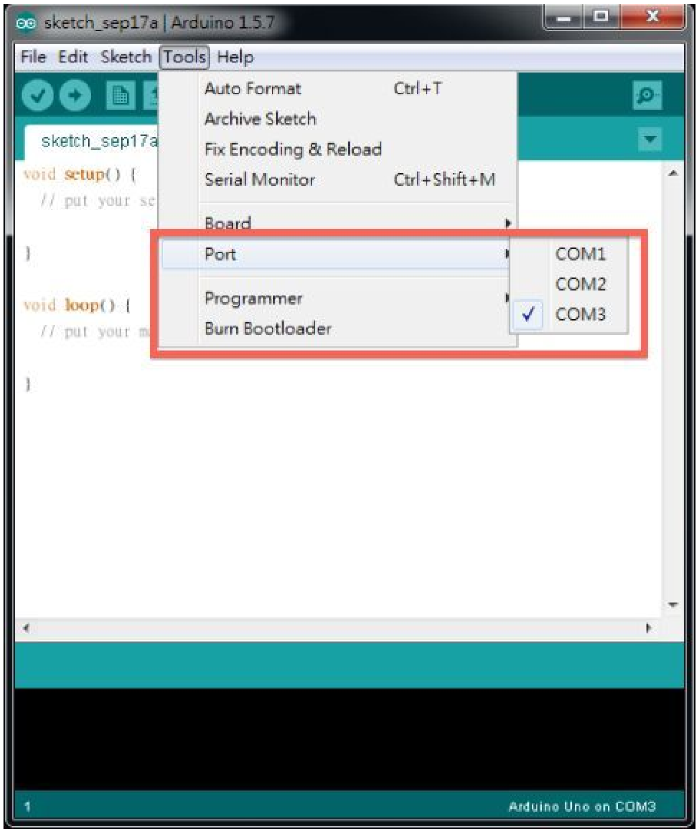
(Please replace "lienhungcheng" with your user name.)





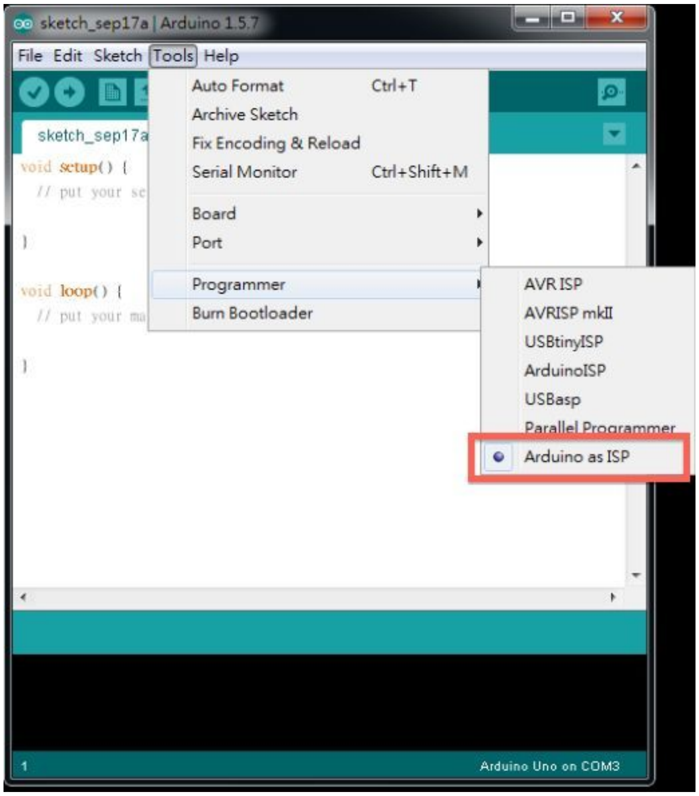


```
void setup() {
  // put your se
}

void loop() {
  // put your ma
}
```

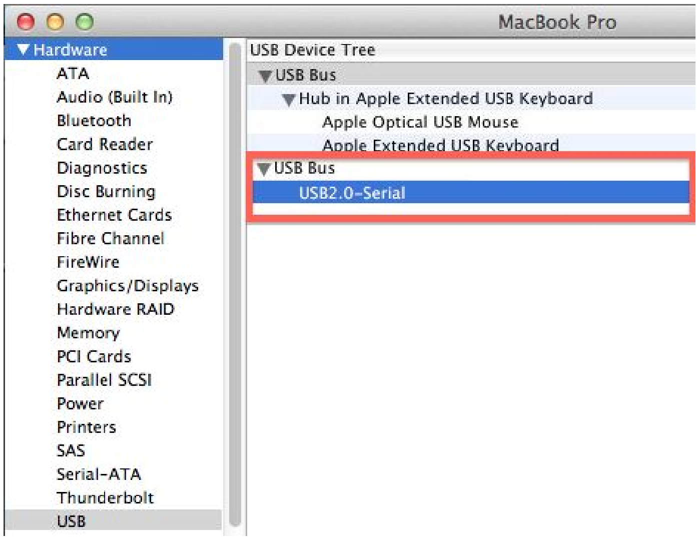


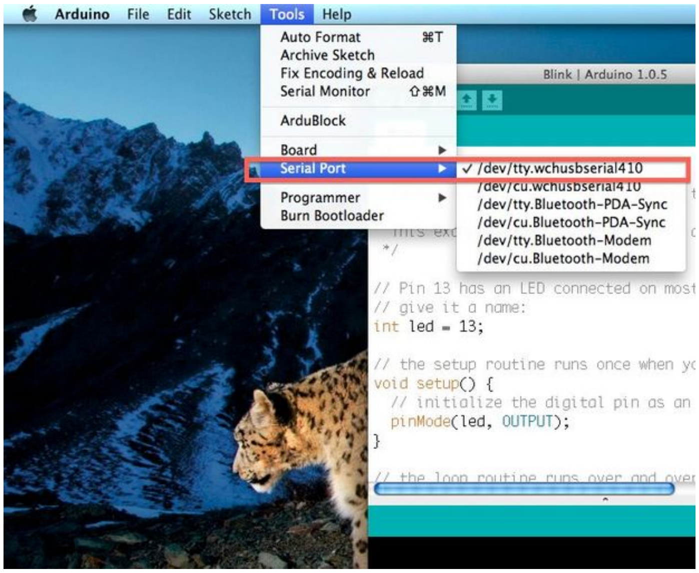
```
void setup() {
  // put your se
}

void loop() {
  // put your m
}
```

Arduino Uno on COM3



```
void setup() {
  // put your se

}

void loop() {
  // put your ma

}
```





```
  This exa
 */

// Pin 13 has an LED connected on most
// give it a name:
int led = 13;

// the setup routine runs once when yo
void setup() {
  // initialize the digital pin as an
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over
```

File Downloads

ardublock.jar (5 MB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'ardublock.jar']

Arduino_driver.exe (227 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'Arduino_driver.exe']

CH341SER_LINUX.zip (8 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'CH341SER_LINUX.zip']

CH341SER_MAC.zip (250 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'CH341SER_MAC.zip']

meArm.ino (2 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'meArm.ino']

S4A.image (7 MB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'S4A.image']

S4AFirmware15_meArm.ino (7 KB)
[NOTE: When saving, if you see .tmp as the file ext, rename it to 'S4AFirmware15_meArm.ino']

Step 1: Turn on the light.

We can use C/S4A/Ardublock to control meArm.Joystick.

A) S4A

(1) Open S4A application.

(2) Drag and drop as the blocks.

(3) Click "Green Flag" and the LED on the bot will light up.

B) Ardublock

(1) Select Tools/Ardublock

(2) Drag and drop as the blocks

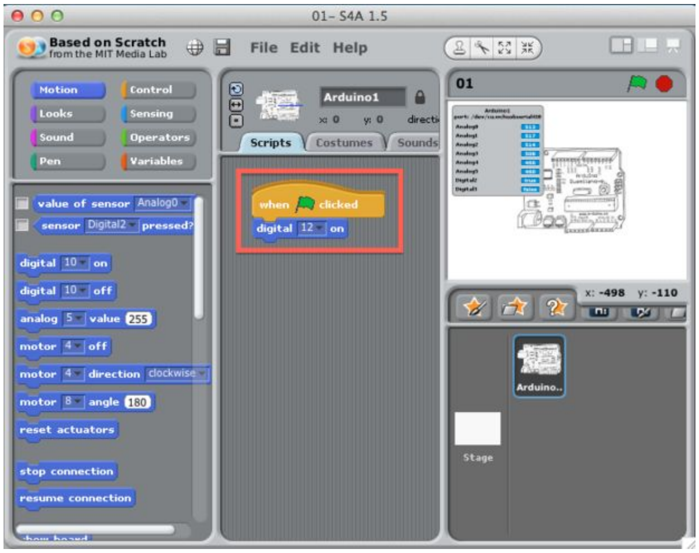
(3) Press "Upload to Arduino"

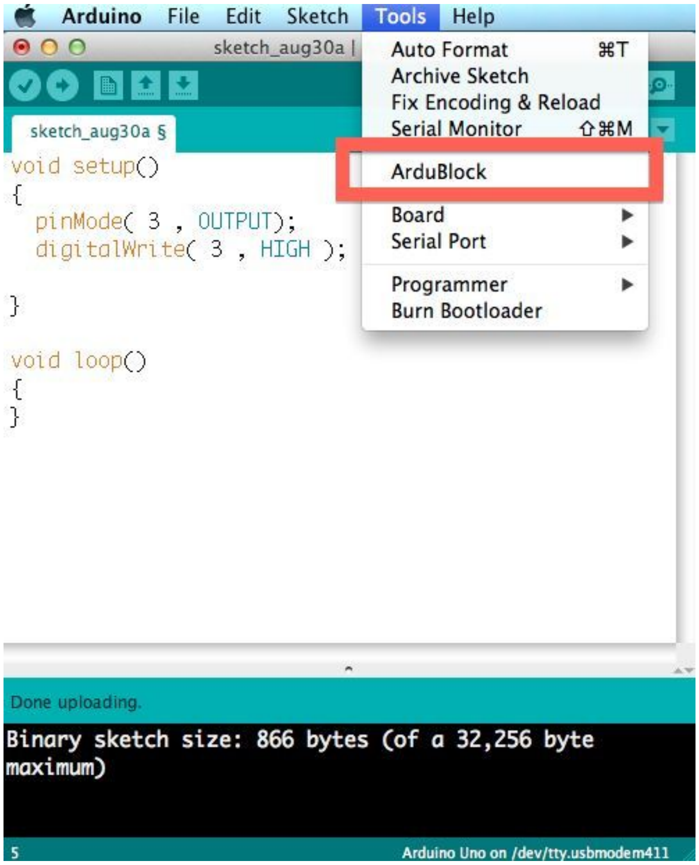
C) C

(1) Open _01.ino

(2) Press "Upload"

PS. The pin of LED is 3 (12 in S4A). It will light up if the value is HIGH. You can open ".sb" in S4A, ".abp" in Ardublock or ".ino" in Arduino IDE to get the same result.

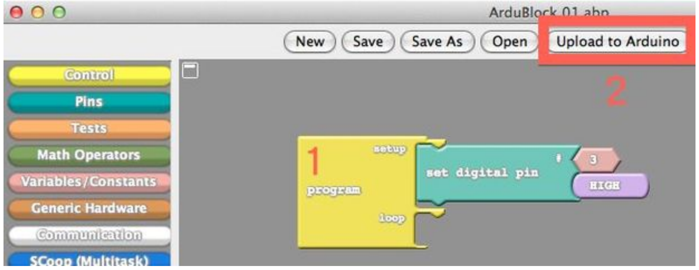




```
void setup()
{
  pinMode( 3 , OUTPUT);
  digitalWrite( 3 , HIGH );

}

void loop()
{
}
```

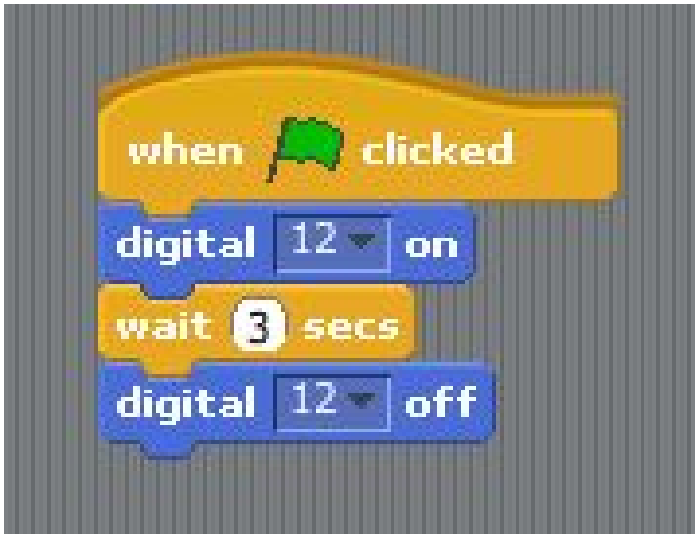
```
Done uploading.
Binary sketch size: 866 bytes (of a 32,256 byte
maximum)
```

5 Arduino Uno on /dev/tty.usbmodem411



Step 2: Turn off the light after 3 seconds

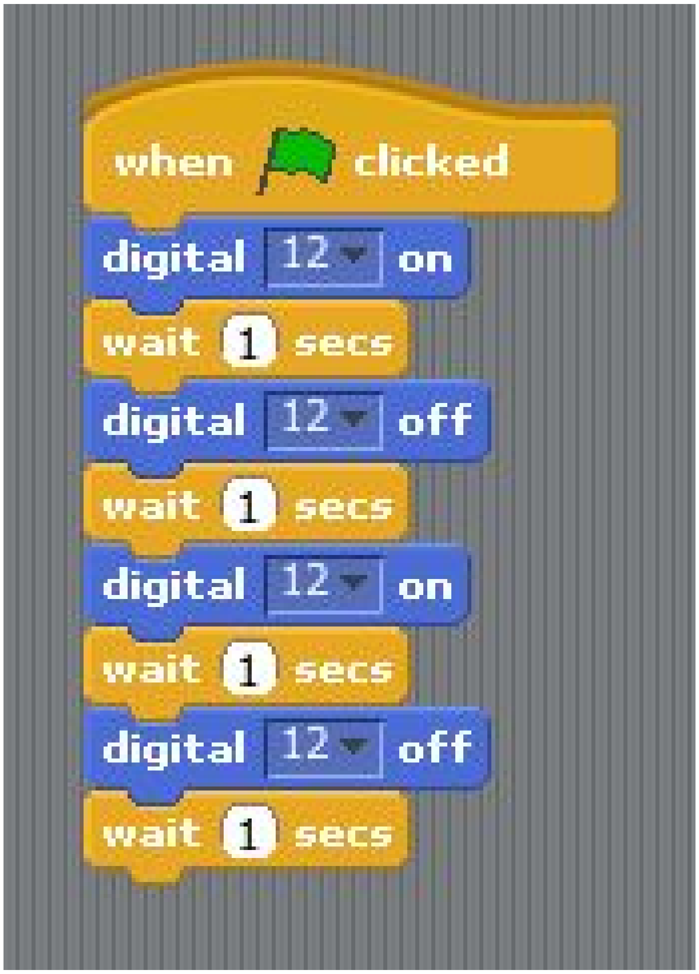
To turn it off, we only have to set the pin to "LOW" or "OFF" after delaying 3000 milli seconds.



```
when clicked
digital 12 on
wait 3 secs
digital 12 off
```

Step 3: Flash it

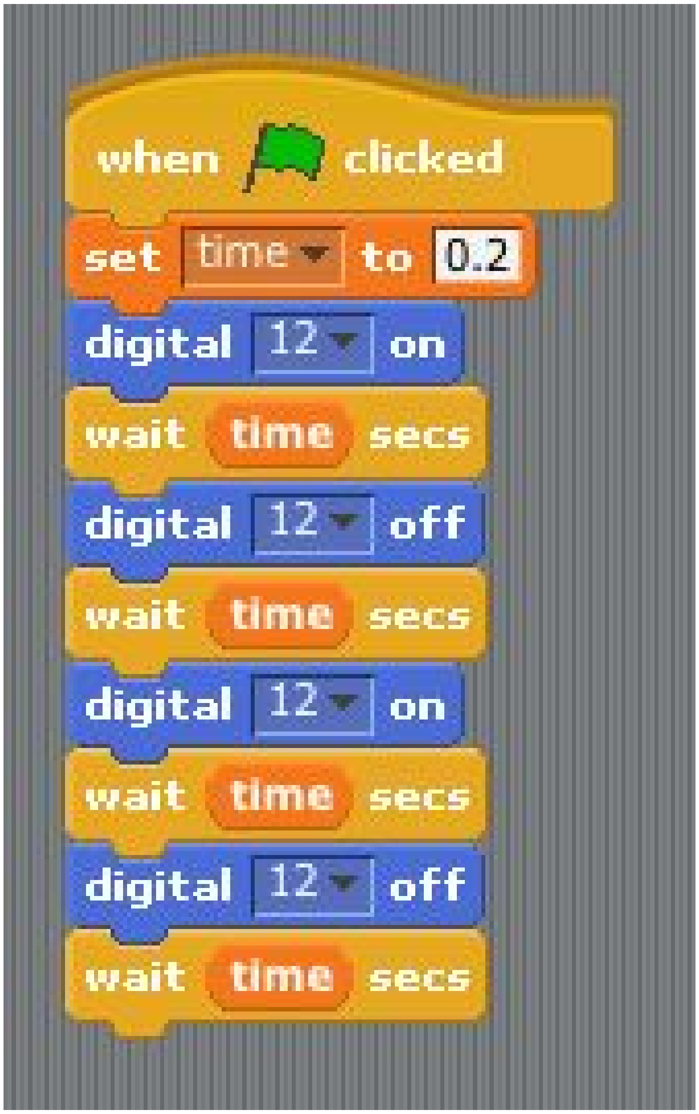
Since you are able to turn the LED on and off, why not flash it? Now let's flash it at every 1 second.



```
when 🏁 clicked
digital 12 on
wait 1 secs
digital 12 off
wait 1 secs
digital 12 on
wait 1 secs
digital 12 off
wait 1 secs
```

Step 4: Variable

Assuming we are changing the flashing frequency from 1 flash per second to 1 flash per 0.2 seconds, we would have to change the delay time many times. If we use Variable, we only have to change Variable one time to get the job done.



```
when [flag] clicked
set time to 0.2
digital 12 on
wait time secs
digital 12 off
wait time secs
digital 12 on
wait time secs
digital 12 off
wait time secs
```

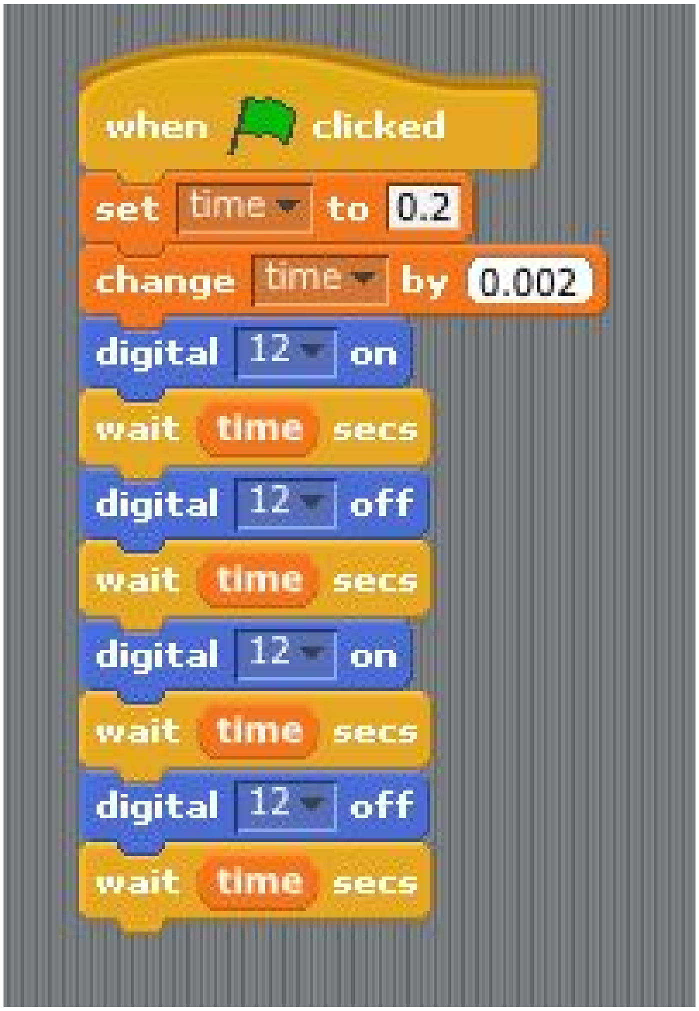
Step 5: Variable (continue)

Since we have variable, why not vary it again? Let's try add/minus/multiply/divide
mytime = mytime / 2;

mytime = mytime * 2;

mytime = mytime + 100;

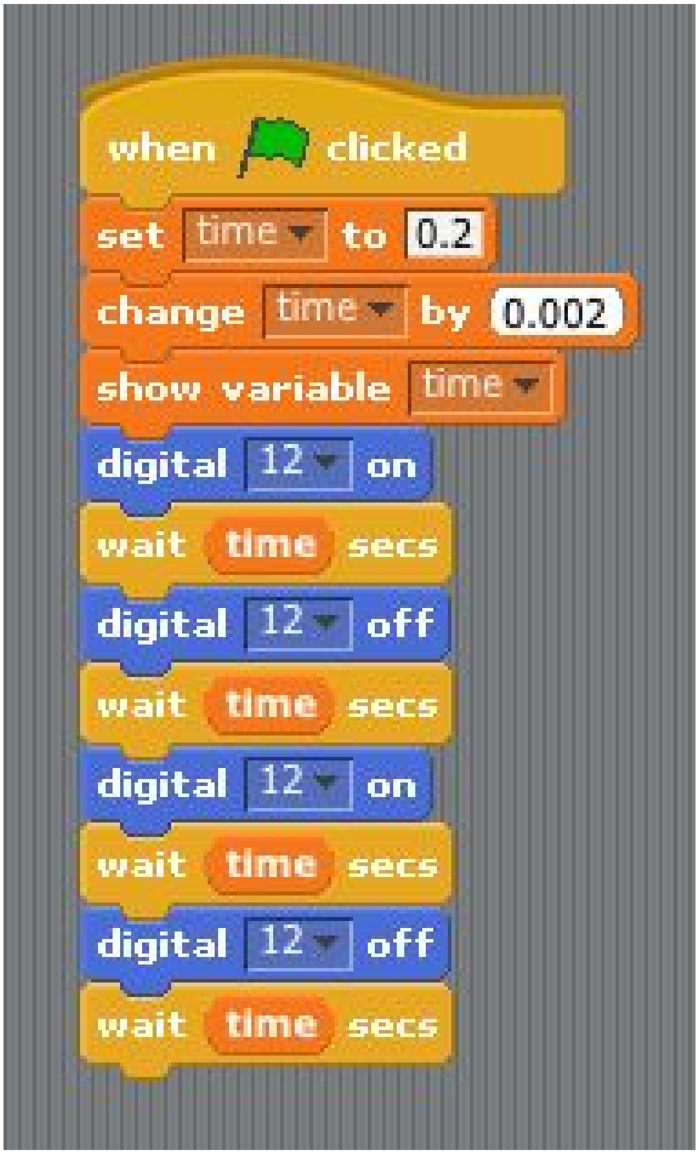
mytime = mytime - 250;



```
when [flag] clicked
set time to 0.2
change time by 0.002
digital 12 on
wait time secs
digital 12 off
wait time secs
digital 12 on
wait time secs
digital 12 off
wait time secs
```

Step 6: Print out variables

Since variable is so efficient, we should handle that as best as possible. We can use below instruction to print it out.

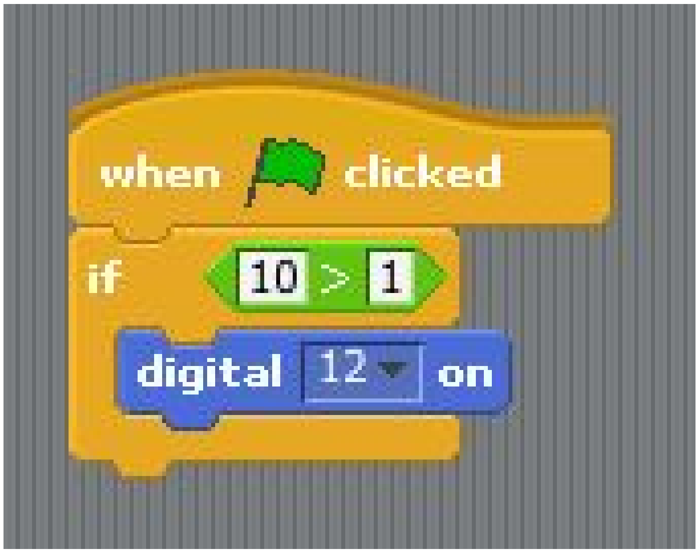


```
when [flag] clicked
set time to 0.2
change time by 0.002
show variable time
digital 12 on
wait time secs
digital 12 off
wait time secs
digital 12 on
wait time secs
digital 12 off
wait time secs
```

Step 7: If I am True

"If" statement is widely used in everyday life. For example, if it rains, bring an umbrella. If it's windy, wear a coat. If the condition is true, do some action.
"If 10 is greater than 1, turn on the LED."

Since 10 is great than 1, so the LED will light up.

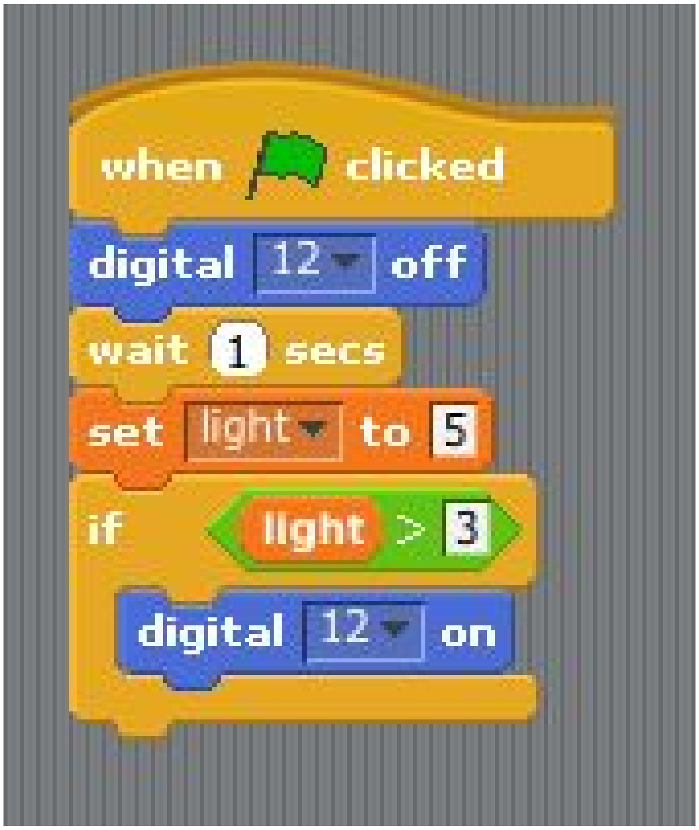


```
when [flag] clicked
if 10 > 1
    digital 12 on
```

Step 8: If + Variable

We normally use Variable with If statement. In this lesson, "light" variable could be current light value from sensor.

```
int light = 5;

if (light < 10)

{

digitalWrite(3, HIGH);

delay(1000);

}
```

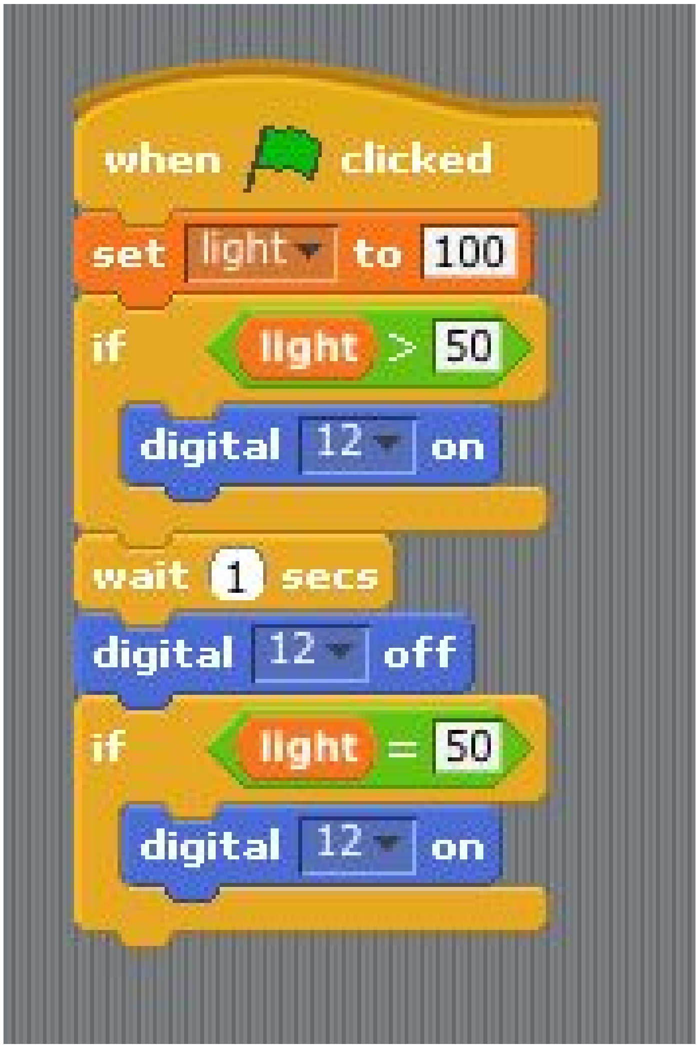


```
when 🏁 clicked
digital 12 off
wait 1 secs
set light to 5
if light > 3
    digital 12 on
```

Step 9: "Boolean" is a friend of "If"

In last lesson, if "(light < 10), we call it "True." If not, we call it "False."

True and False are both boolean. We do certain action if the boolean value is True, and do others if False.

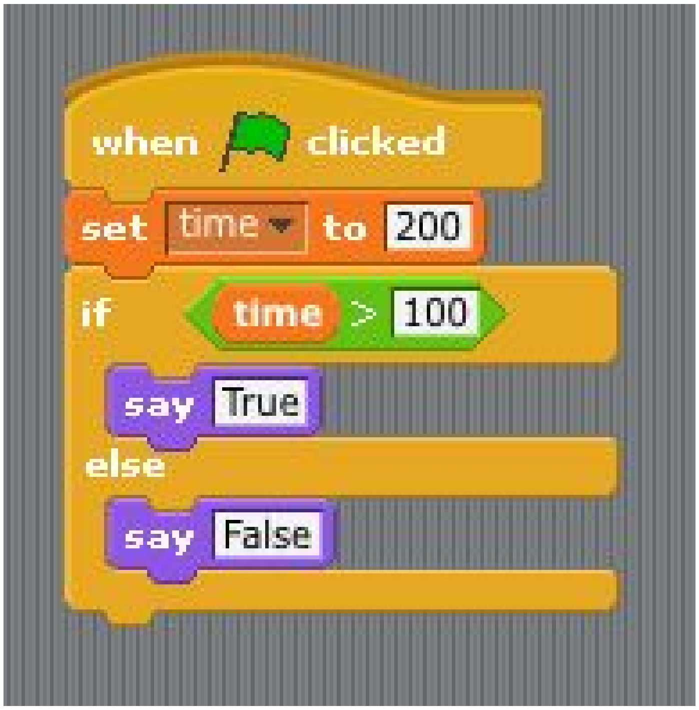


```
when [flag] clicked
set light to 100
if light > 50
    digital 12 on
wait 1 secs
digital 12 off
if light = 50
    digital 12 on
```

Step 10: Left turn or right turn (if-else)

Since there are 2 kinds of boolean value, the statement usually looks like "if-else."

```
if (true)
{
left turn
}
else
{
right turn
}
```



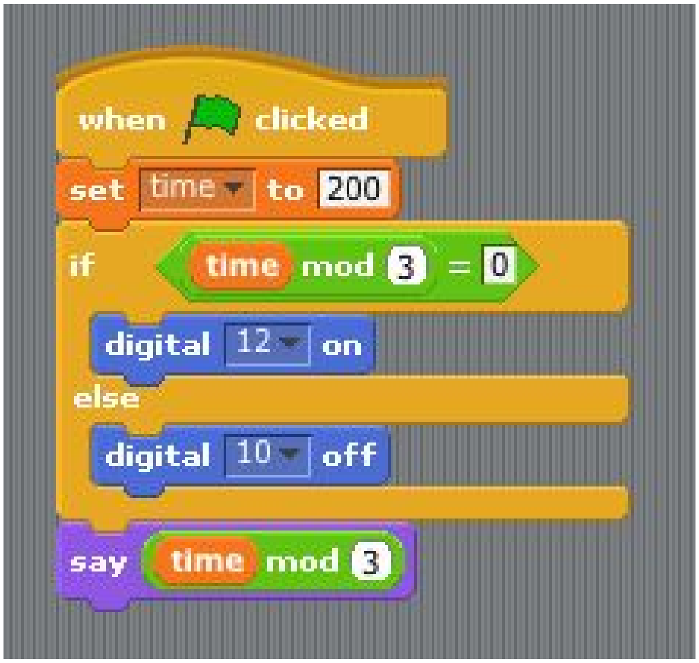
```
when clicked
set time to 200
if time > 100
    say True
else
    say False
```

Step 11: Left turn or right turn (if-else) continue 1

"if-else" statement is frequently used with variable.

For instance, "if variable can be divided by 3, then blabla…"
("|" is modular)

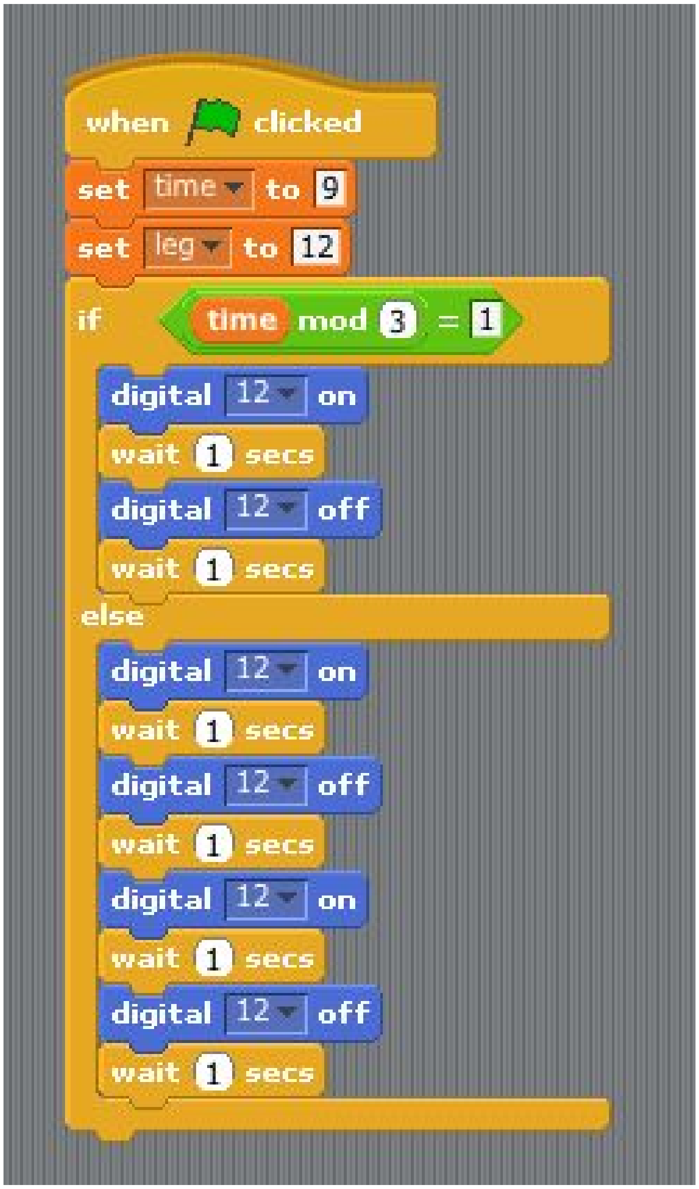
```
if (num ? 3 == 0)

{

digitalWrite(3, HIGH);

} else {

digitalWrite(3, LOW);

}
```



```
when clicked
set time to 200
if time mod 3 = 0
    digital 12 on
else
    digital 10 off
say time mod 3
```

Step 12: Left turn or right turn (if-else) continue 2

Since "if-else" is so important, let's do more practice.
Can you figure out how to write the statement "if odd, flash LED 1 time, else flash 2 times?"

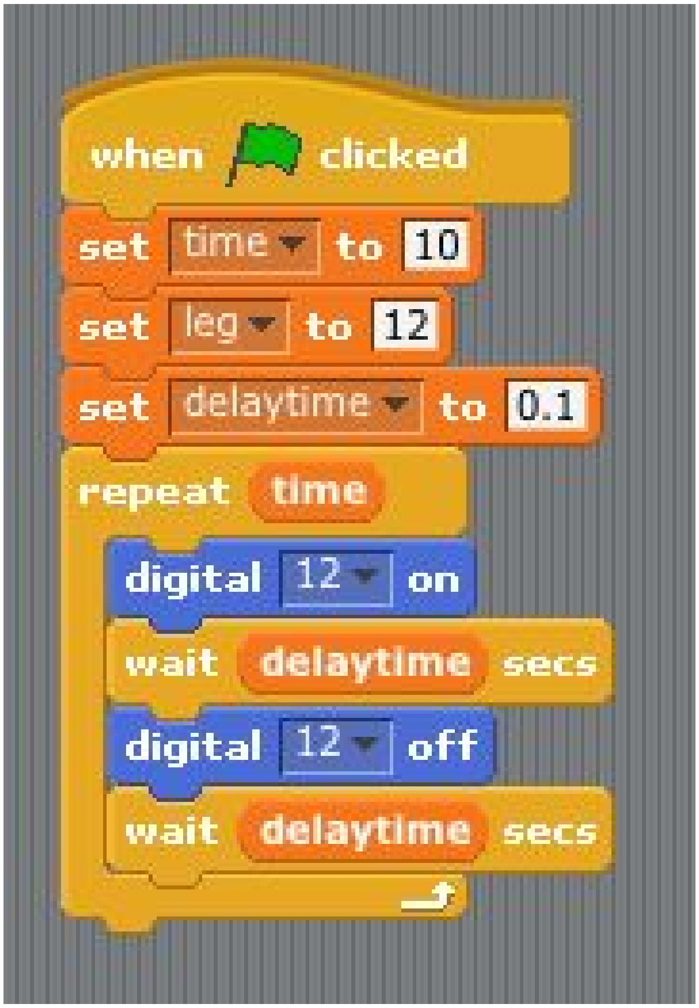


```
when ⚑ clicked
set time to 9
set leg to 12
if time mod 3 = 1
    digital 12 on
    wait 1 secs
    digital 12 off
    wait 1 secs
else
    digital 12 on
    wait 1 secs
    digital 12 off
    wait 1 secs
    digital 12 on
    wait 1 secs
    digital 12 off
    wait 1 secs
```

Step 13: Loop : capable of doing repeated boring things

Besides "if-else", another fantastic work computer can do is "loop." It's capable of doing repeated and boring things.
For example, let's flash LED 5 times.

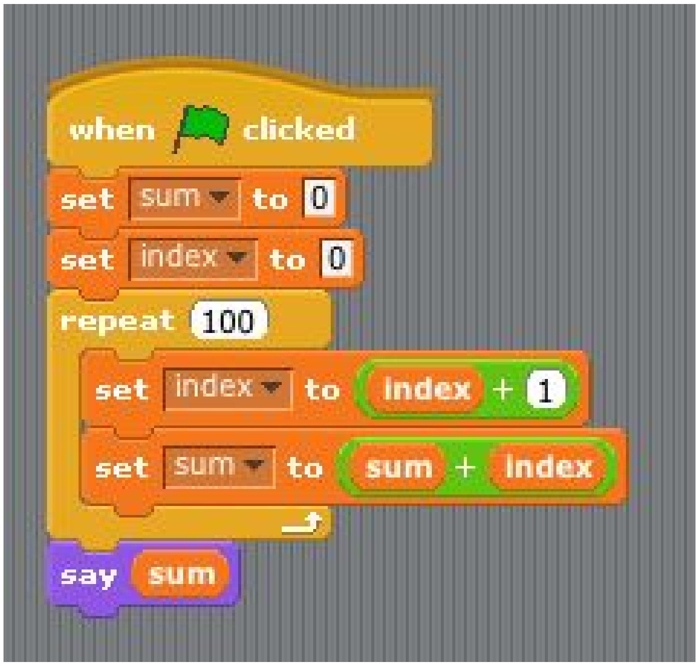
basic format in C language : "for (int i = 0; i < 5; i++)"



```
when [flag] clicked
set time to 10
set leg to 12
set delaytime to 0.1
repeat (time)
    digital 12 on
    wait (delaytime) secs
    digital 12 off
    wait (delaytime) secs
```

Step 14: **Use loop to add from 1 to 100**

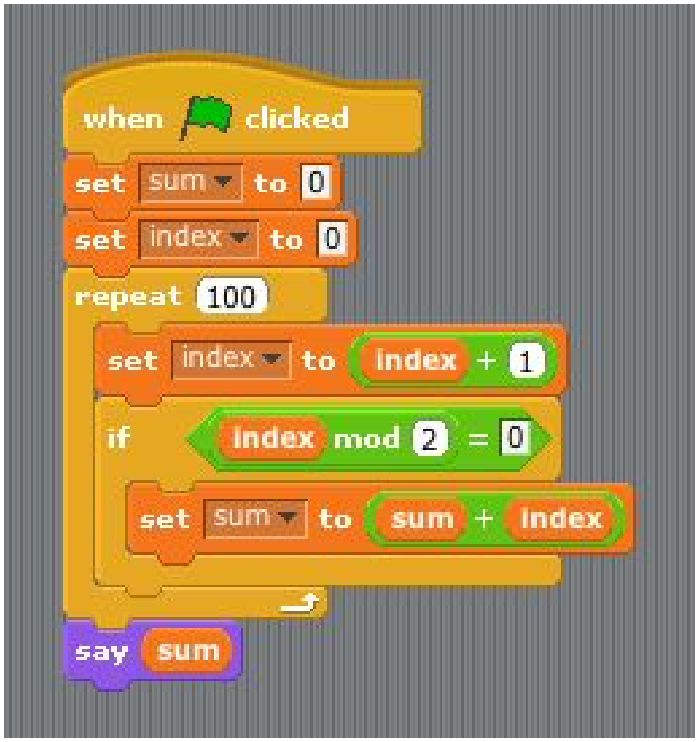
Let's practice how to add from 1 to 100 with loop.



```
when [flag] clicked
set sum to 0
set index to 0
repeat 100
    set index to ((index) + 1)
    set sum to ((sum) + (index))
say (sum)
```

Step 15: We can also only add even number within the loop.

"Loop" is so powerful. Let's try to add only even number within 1 to 100.

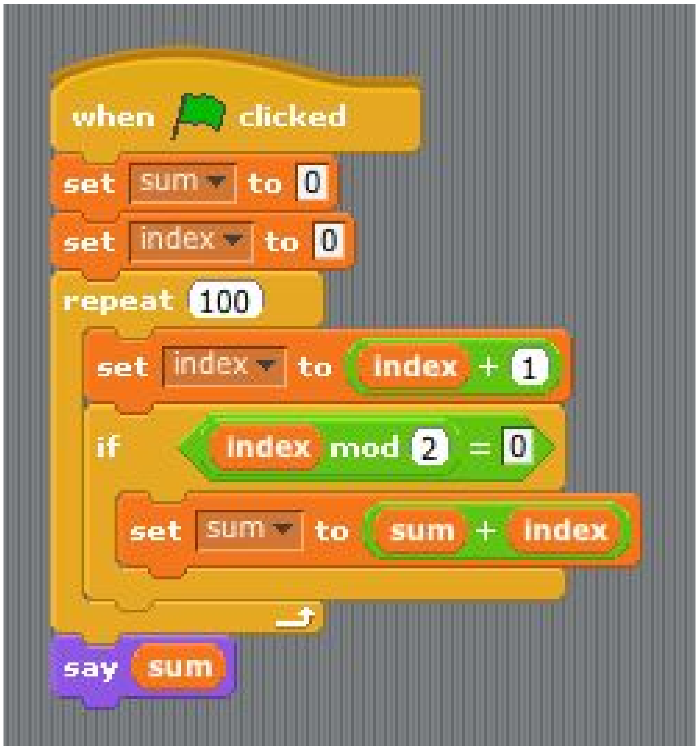


```
when 🏴 clicked
set sum to 0
set index to 0
repeat 100
    set index to (index + 1)
    if <((index) mod (2)) = 0>
        set sum to (sum + index)
say sum
```

Step 16: "step" of loop

The idea of step of the loop is to do action once per "step" times.

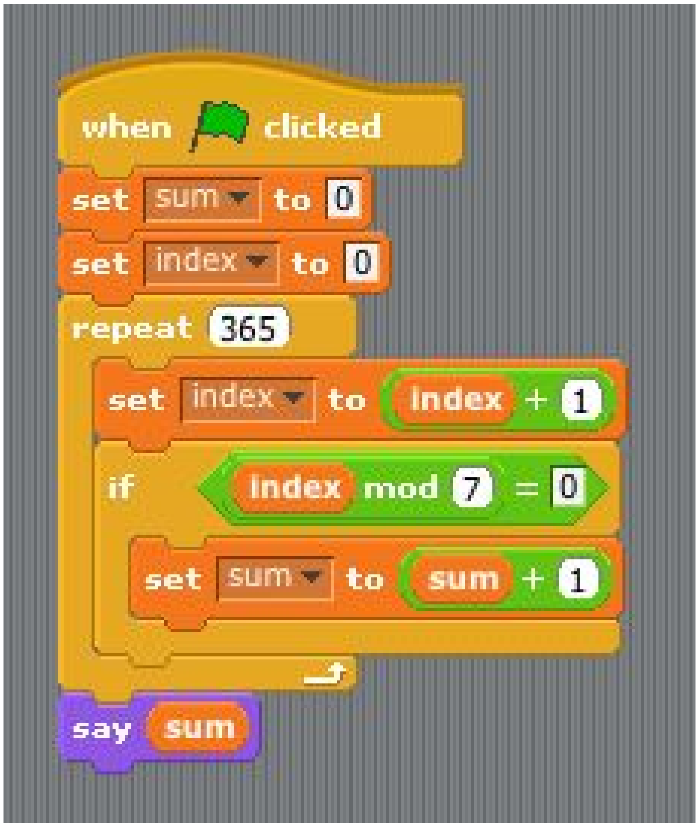
So if we can to add only even number with step 2.



```
when 🏴 clicked
set sum to 0
set index to 0
repeat 100
    set index to (index + 1)
    if <((index) mod (2)) = 0>
        set sum to (sum + index)
say sum
```

Step 17: Can we use "step" to find out total Sundays in a year?

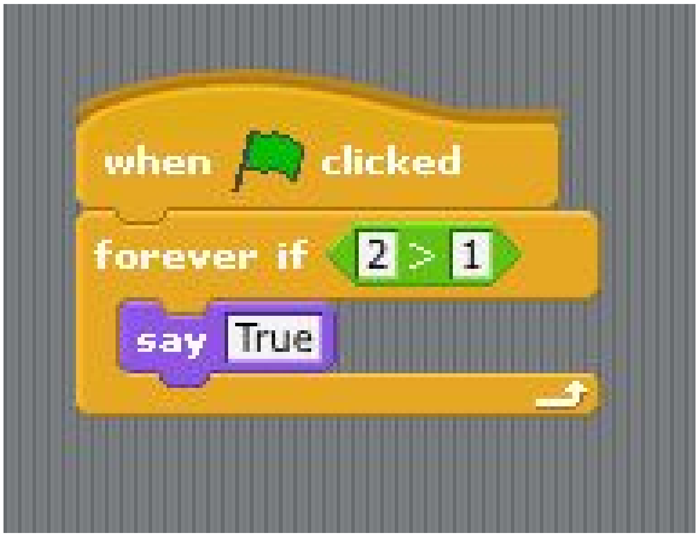
The key of this quiz is to set the "step" value.



```
when [flag] clicked
set sum to 0
set index to 0
repeat 365
    set index to (index + 1)
    if <(index mod 7) = 0>
        set sum to (sum + 1)
say sum
```

Step 18: Loop (continue) : do something forever

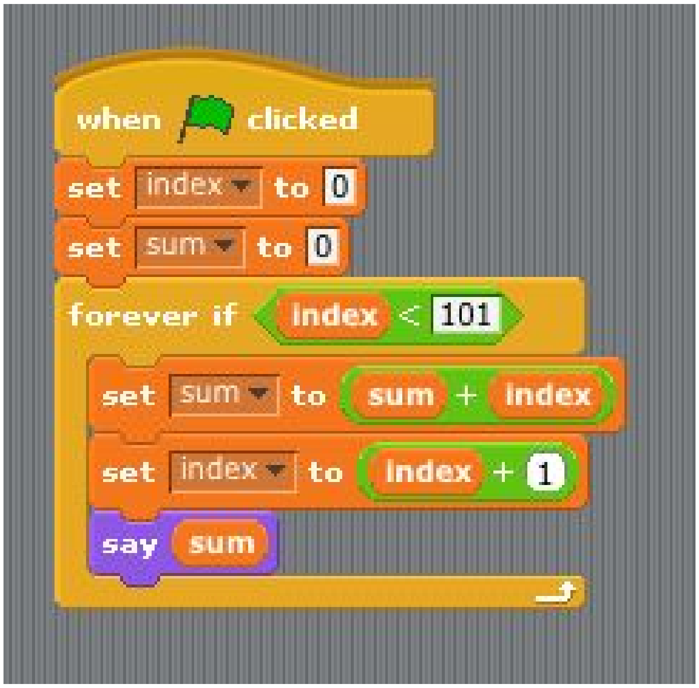
Loop statement is so important in most programming language. So there is another variation of this statement "while." The key of "while" statement is to do forever if the condition is met.



```
when [flag] clicked
forever if <2 > 1>
    say True
```

Step 19: Loop (continue) : standard form

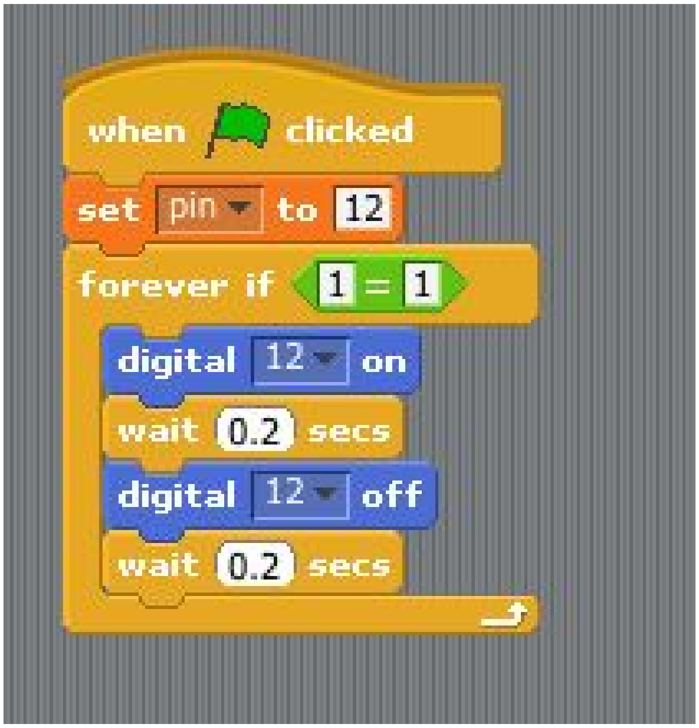
In order not to hang the system, we usually add a condition in the while loop. So the while loop will stop when the condition is False.



```
when [flag] clicked
set index to 0
set sum to 0
forever if <index < 101>
    set sum to (sum + index)
    set index to (index + 1)
    say sum
```

Step 20: Flash the LED forever

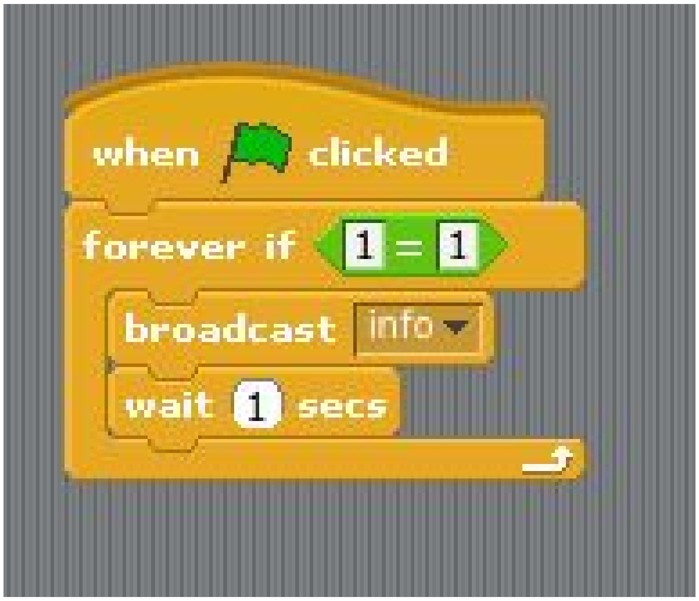
The while loop is best to do something unchanged repeatedly. This lesson is to learn how to flash the LED forever.



```
when [flag] clicked
set pin to 12
forever if <1 = 1>
    digital 12 on
    wait 0.2 secs
    digital 12 off
    wait 0.2 secs
```

Step 21: Function : a set of statements in a blackbox

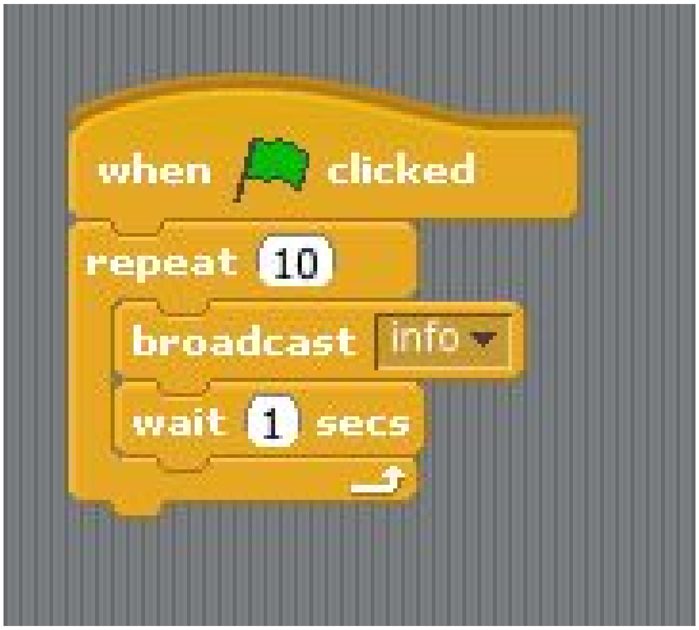
To make the code tidy, programmer used to utilize "Function" to run a set of statement as a blackbox. This blackbox is so-called "Function" or "Subroutine." We will learn how to use Function in this lesson.
ps. Only Arduino C and Ardublock support "Function" capability. S4A doesn't.



```
when [flag] clicked
forever if <(1) = (1)>
    broadcast [info ▾]
    wait (1) secs
```

Step 22: Function + Loop : let's flash the LED 10 times.

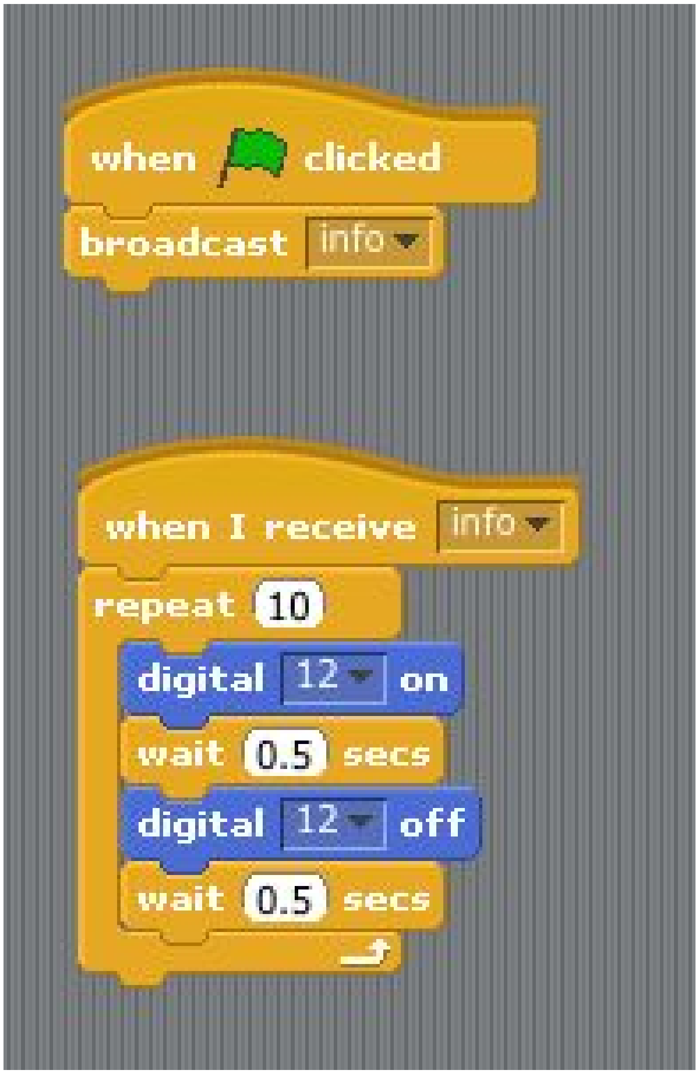
After wraping multiple statements into Function, we can treat this special Function as a statement. Hence we can put the Function into a loop as well. Let's see how to flash LED 10 times by using Function and loop.



```
when [flag] clicked
repeat (10)
    broadcast [info ▾]
    wait (1) secs
```

Step 23: Loop inside the Function

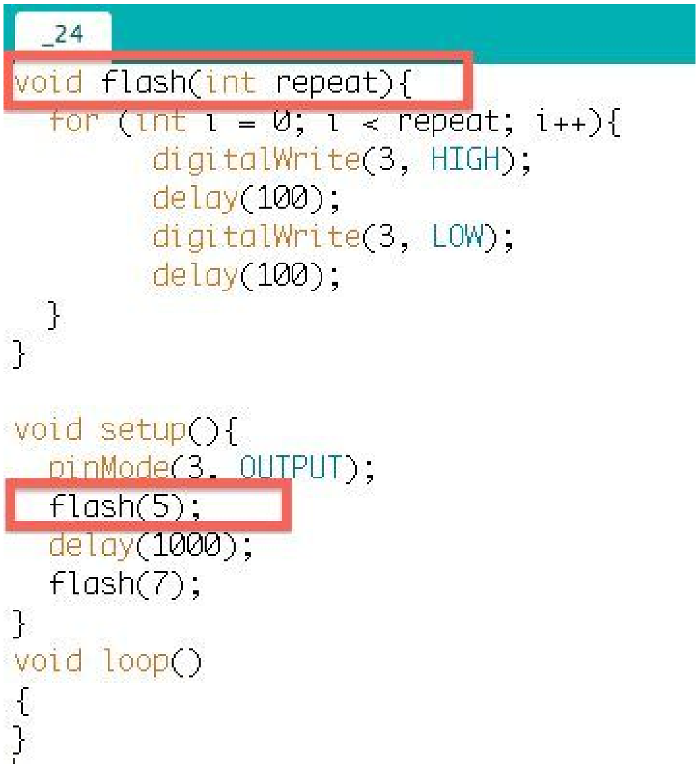
Not only can we run Function within a loop, we can run loop within a Function as well.



```
when [flag] clicked
broadcast info

when I receive info
repeat 10
    digital 12 on
    wait 0.5 secs
    digital 12 off
    wait 0.5 secs
```

Step 24: Parameter of a Function (Only C supports, Ardublock doesn't.)

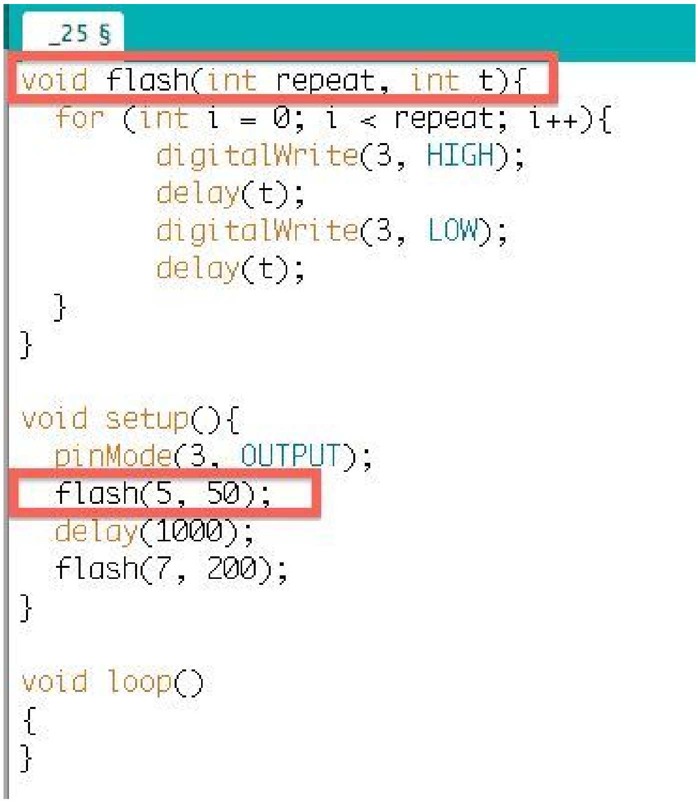
"Function" is so powerful. For instance, we can provide a parameter when run a Function. In this example, we will pass a parameter to a Function to flash the LED X times.



```
void flash(int repeat){
  for (int i = 0; i < repeat; i++){
        digitalWrite(3, HIGH);
        delay(100);
        digitalWrite(3, LOW);
        delay(100);
  }
}

void setup(){
  pinMode(3, OUTPUT);
  flash(5);
  delay(1000);
  flash(7);
}
void loop()
{
}
```

Step 25: Can we pass multiple parameter to a Function?

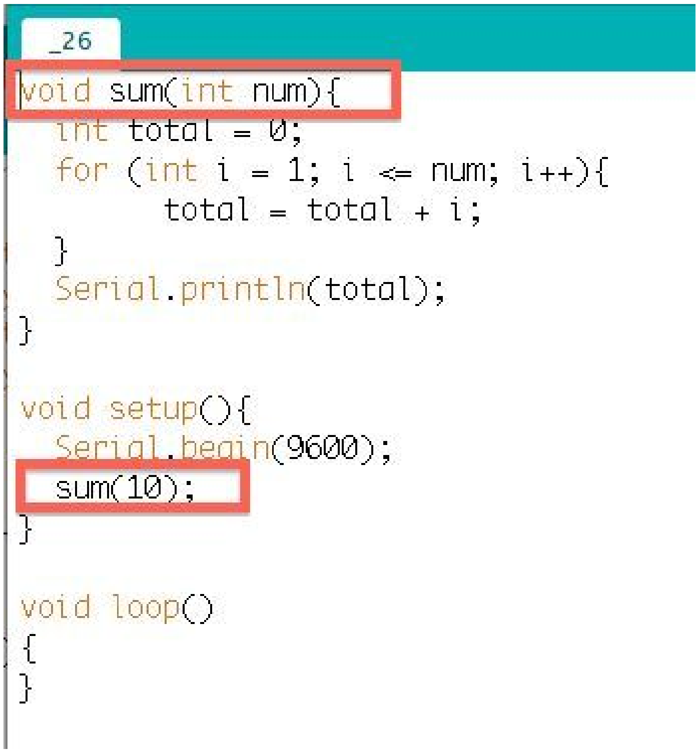
No only can we pass parameter to a Function, we can pass multiple ones as well. For example, we can pass 2 parameters. One is for the times of flash while the other as the frequency.



```
void flash(int repeat, int t){
  for (int i = 0; i < repeat; i++){
        digitalWrite(3, HIGH);
        delay(t);
        digitalWrite(3, LOW);
        delay(t);
  }
}

void setup(){
  pinMode(3, OUTPUT);
  flash(5, 50);
  delay(1000);
  flash(7, 200);
}

void loop()
{
}
```

Step 26: More challenging

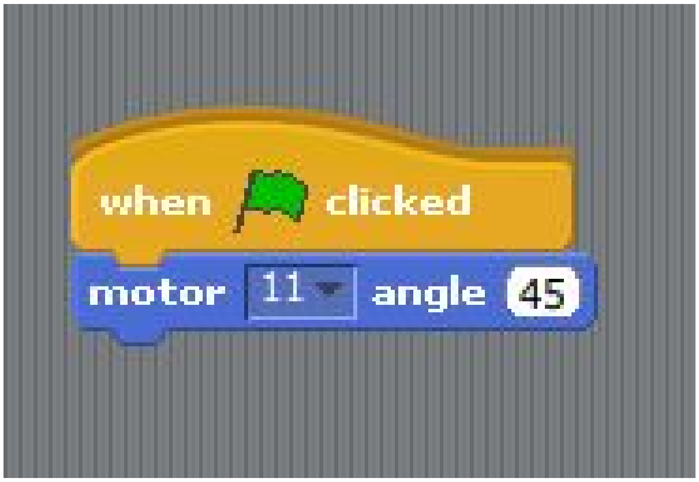
A Function with parameter is so powerful. So give a try and see if you can work out a Function to add from 1 to number y. So if we want to add from 1 to 100, we can just call "Sum(100)."



```
void sum(int num){
  int total = 0;
  for (int i = 1; i <= num; i++){
        total = total + i;
  }
  Serial.println(total);
}

void setup(){
  Serial.begin(9600);
  sum(10);
}

void loop()
{
}
```

Step 27: Servo at a glance

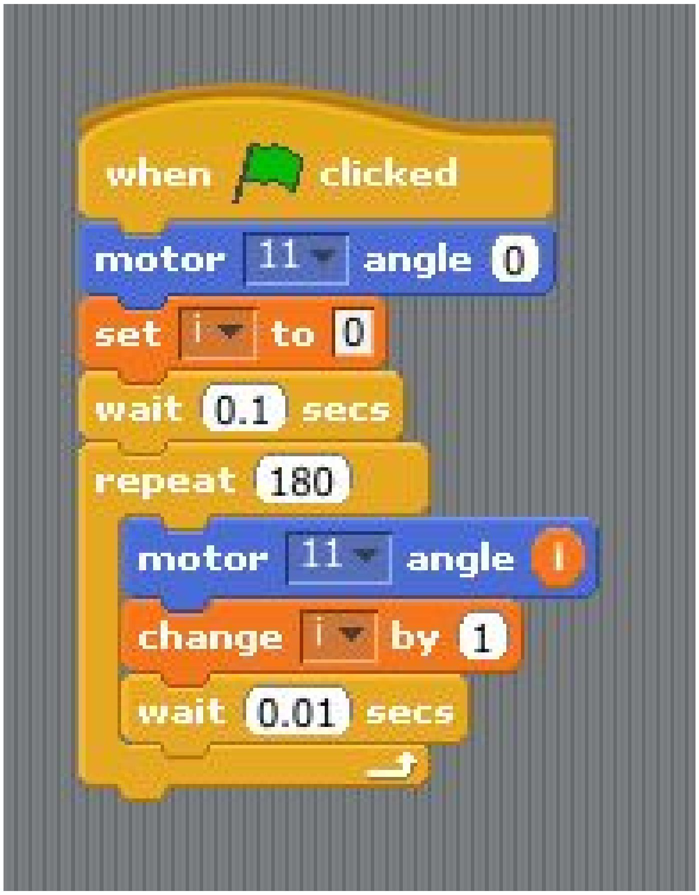
From this lesson, we are going to learn how to control meArm.Joystick. Since you have run through previous lesson, you can get the idea how to control the bot quick. The mechanic arm is driven by 4 servos. We only have to set the pin number and angle of the servo, and Arduino will do the others for you. The angle is ranged from 1 to 179. Please be careful that don't turn the servo to an angle out of range. This will harm the system and servo. Please pay special attention that don't drive the screws too tight. As it will be harmful to the system if the servo can't turn to a specific angle due to this reason.



```
when clicked
motor 11 angle 45
```

Step 28: Turn the servo slowly from 1 to 179 degree

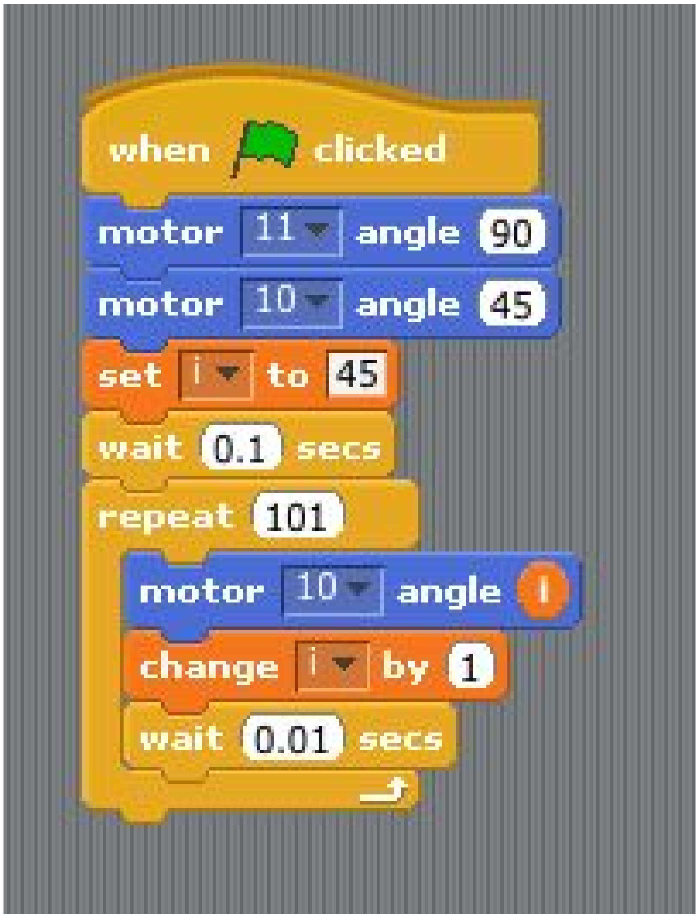
Since the range of the servo is 1~179, let's rotate the bot degree by degree.



```
when 🟩 clicked
motor 11 angle 0
set i to 0
wait 0.1 secs
repeat 180
    motor 11 angle i
    change i by 1
    wait 0.01 secs
```

Step 29: Let's try the 2nd servo with angle from 45 to 145 degree.

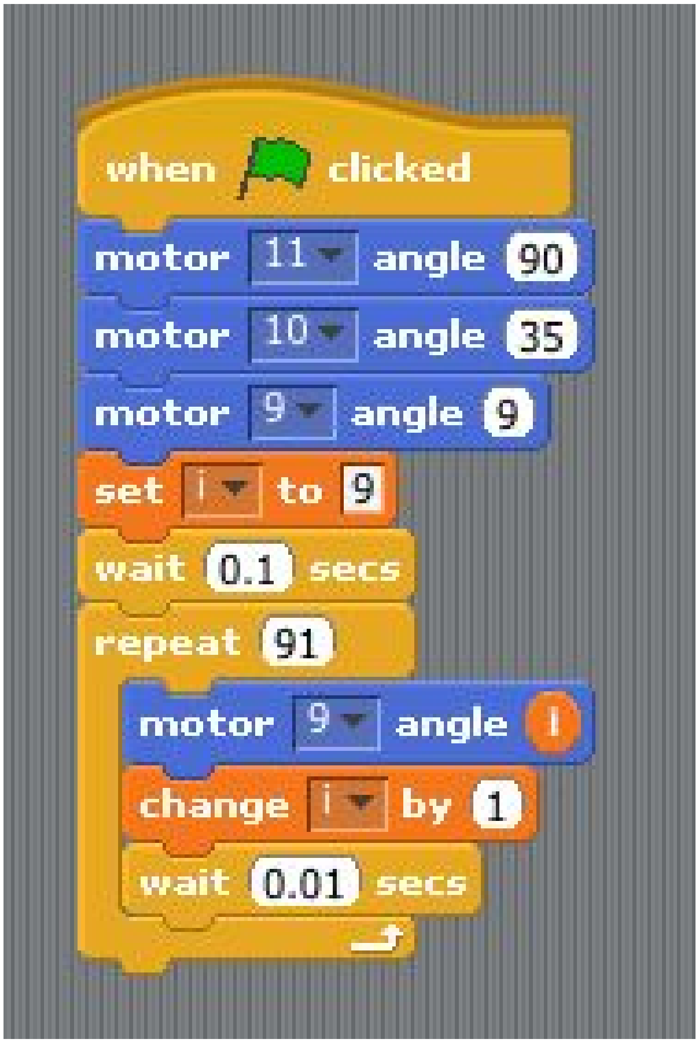
The 2nd servo is to control meArm.Joystick forward and backward. Let's rotate it from 45 to 145 degree.



```
when 🟩 clicked
motor 11 angle 90
motor 10 angle 45
set i to 45
wait 0.1 secs
repeat 101
    motor 10 angle i
    change i by 1
    wait 0.01 secs
```

Step 30: Let's try the 3rd servo with angle from 90 to 179 degree.

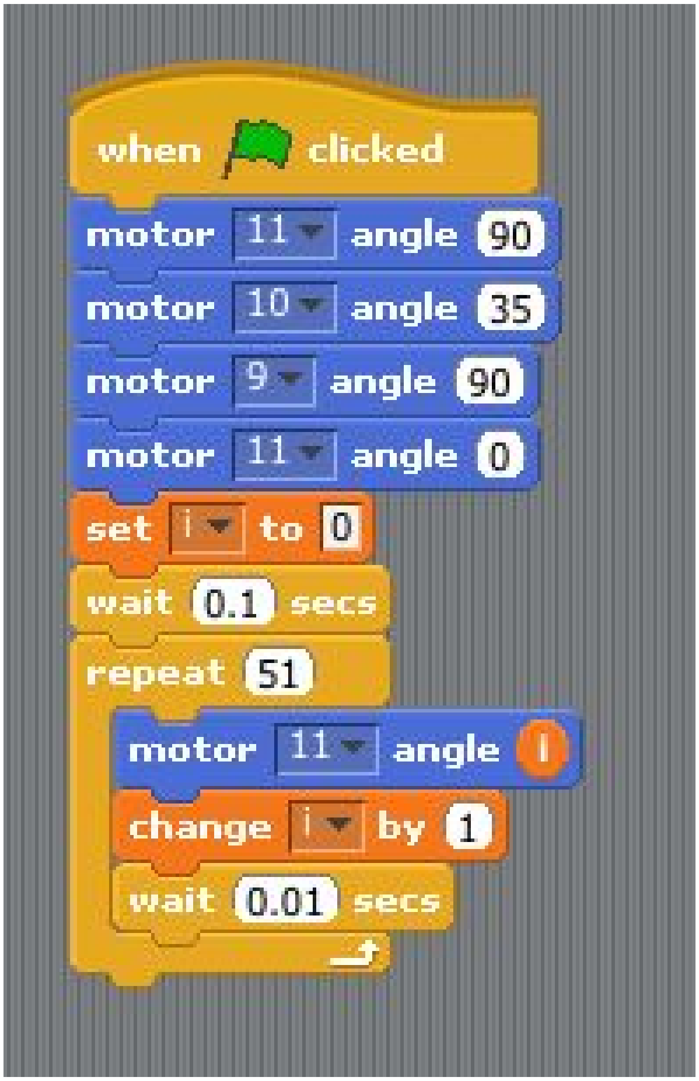
The 3rd servo is to control the direction of up and down. Let's rotate it from 90 to 179 degree.



```
when ⚑ clicked
motor 11 angle 90
motor 10 angle 35
motor 9 angle 9
set i to 9
wait 0.1 secs
repeat 91
    motor 9 angle i
    change i by 1
    wait 0.01 secs
```

Step 31: Let's try the 4th servo with angle from 0 to 50 degree.

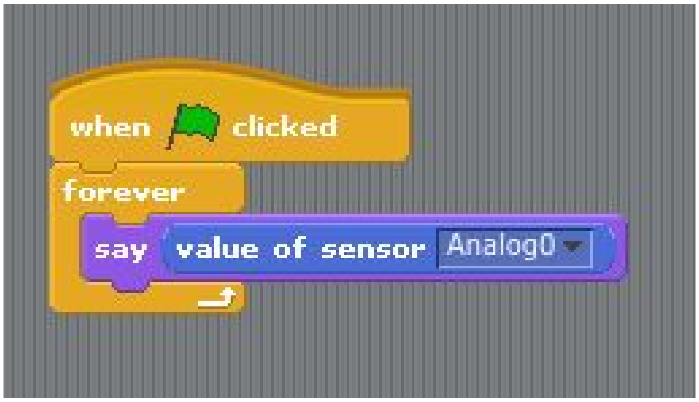
The 4th servo is to control the claw. Let's rotate it from 1 to 50 degree.



```
when ⚑ clicked
motor 11 angle 90
motor 10 angle 35
motor 9 angle 90
motor 11 angle 0
set i to 0
wait 0.1 secs
repeat 51
    motor 11 angle i
    change i by 1
    wait 0.01 secs
```

Step 32: Read the value of joystick

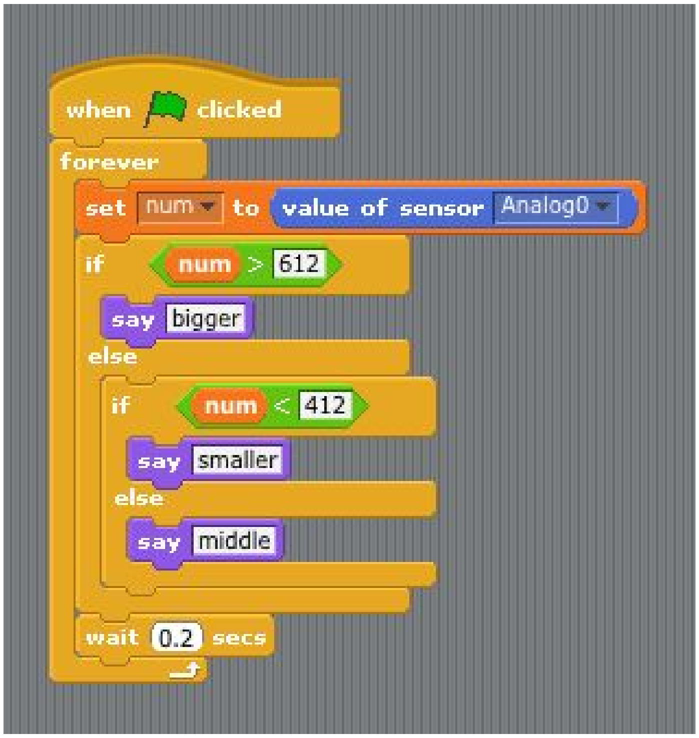
There are 2 joysticks with 2 directions(left-right, forward-backward) of each joystick. So we need to monitor 4 directions through analog pins A0, A1, A2 and A3. The value of each analog pin is from 0 through 1023. In this lesson, we read value from A0 and print it out through serial port.



```
when ⚑ clicked
forever
    say value of sensor Analog0
```

Step 33: Let's divide the value of 0~1023 into 3 parts

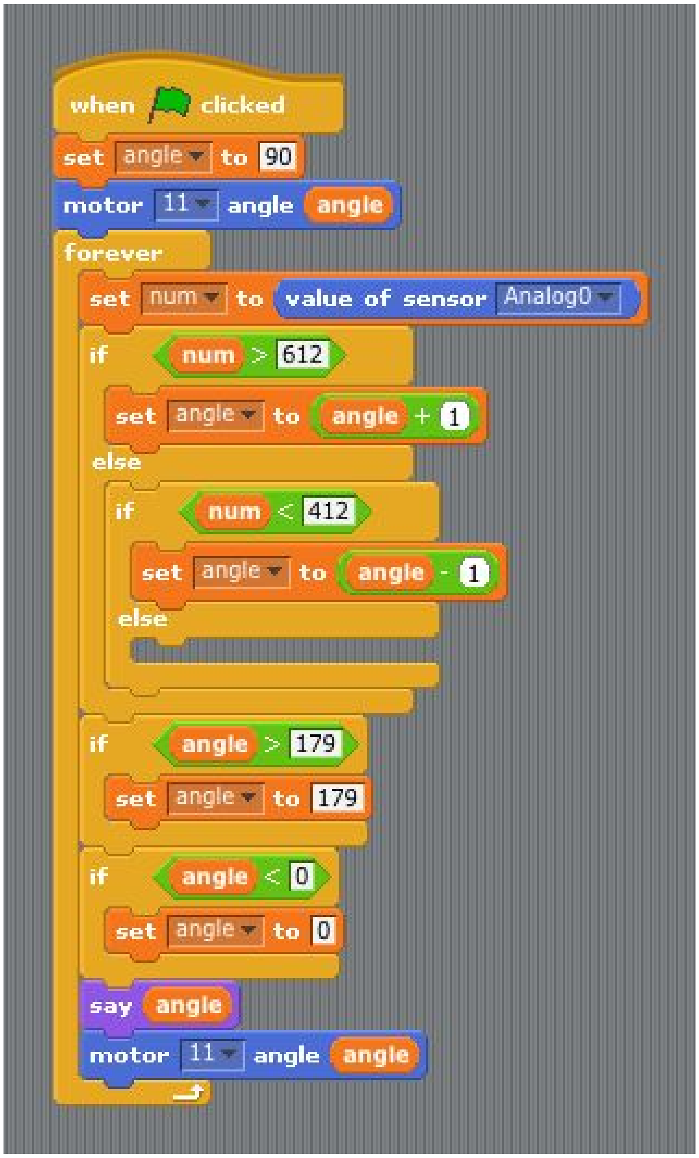
The reason is to detect the user operation of the joystick. If the value is greater than 612, it says "bigger", if less than 412 it says "smaller", otherwise says "middle."



```
when clicked
forever
    set num to value of sensor Analog0
    if num > 612
        say bigger
    else
        if num < 412
            say smaller
        else
            say middle
    wait 0.2 secs
```

Step 34: Combine joystick and servo

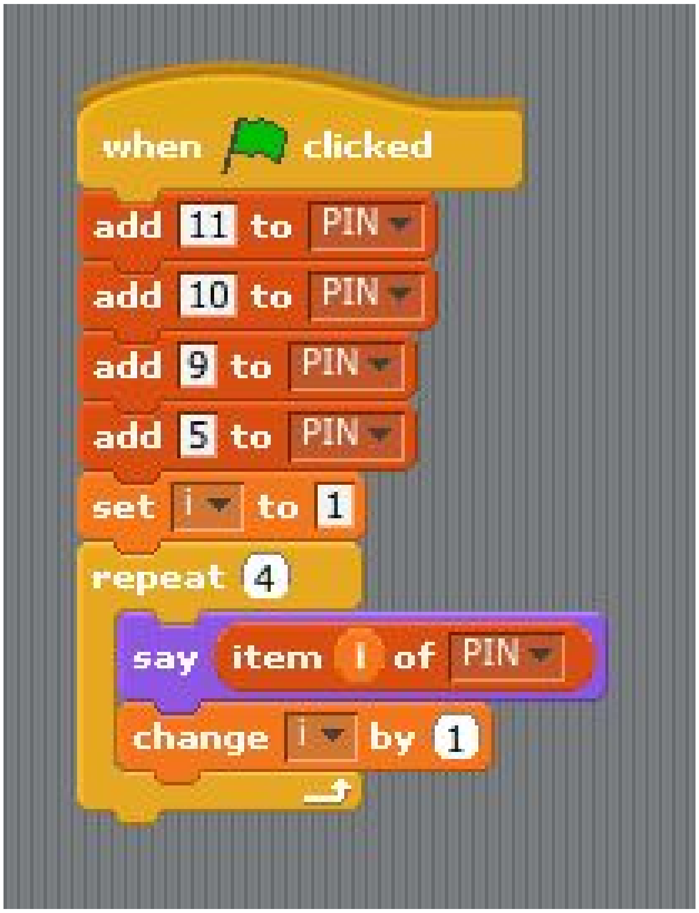
Let's combine the joystick and servo by adding or minusing the servo degree based on joystick direction.



```
when ⚑ clicked
set angle to 90
motor 11 angle angle
forever
    set num to value of sensor Analog0
    if num > 612
        set angle to angle + 1
    else
        if num < 412
            set angle to angle - 1
        else
    if angle > 179
        set angle to 179
    if angle < 0
        set angle to 0
    say angle
    motor 11 angle angle
```

Step 35: Array : collect similar things together

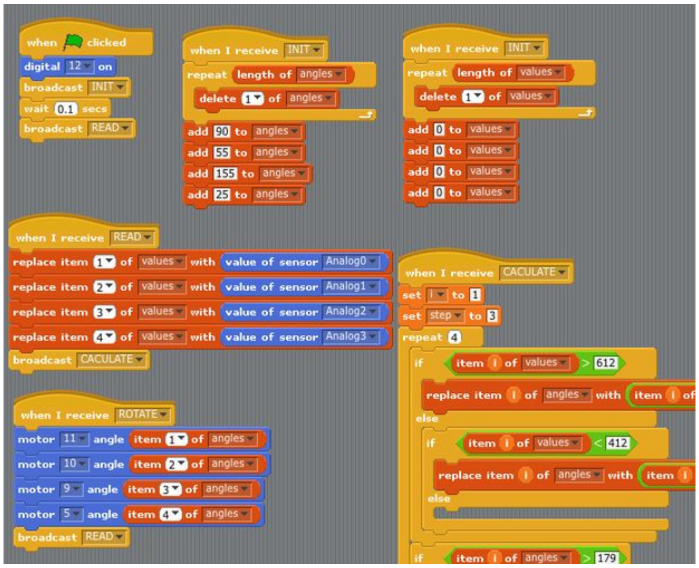
Since we are reading value from 4 joysticks and control the degrees of 4 servos, we can use Array to store similar values in an "Array" variable. So the code will look tidy.



```
when [flag] clicked
add 11 to PIN
add 10 to PIN
add 9 to PIN
add 5 to PIN
set i to 1
repeat 4
    say item i of PIN
    change i by 1
```

Step 36: Combine everything together and the bot will be under your control

To fully control the bot, we need to use all the programming skills we learned from previous lessons. We won't go through each line of code. Try to study the code and you can understand it after short period of time.



```
when [flag] clicked
digital 12 on
broadcast INIT
wait 0.1 secs
broadcast READ
```

```
when I receive INIT
repeat length of angles
    delete 1 of angles
add 90 to angles
add 55 to angles
add 155 to angles
add 25 to angles
```

```
when I receive INIT
repeat length of values
    delete 1 of values
add 0 to values
add 0 to values
add 0 to values
add 0 to values
```

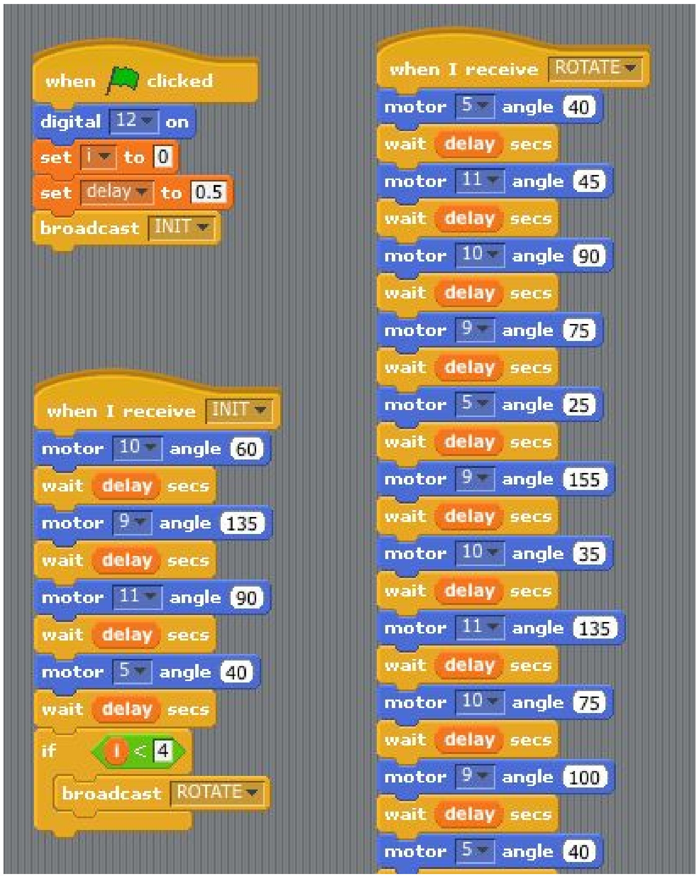
```
when I receive READ
replace item 1 of values with value of sensor Analog0
replace item 2 of values with value of sensor Analog1
replace item 3 of values with value of sensor Analog2
replace item 4 of values with value of sensor Analog3
broadcast CACULATE
```

```
when I receive ROTATE
motor 11 angle item 1 of angles
motor 10 angle item 2 of angles
motor 9 angle item 3 of angles
motor 5 angle item 4 of angles
broadcast READ
```

```
when I receive CACULATE
set i to 1
set step to 3
repeat 4
    if item i of values > 612
        replace item i of angles with item i of
    else
        if item i of values < 412
            replace item i of angles with item i
        else
    if item i of angles > 179
```

Step 37: Auto mode

Since we knew how to control servo, we can make the bot to automatically do something if we can set the angle of each servo in advance.



```
when ⚑ clicked
digital 12 on
set i to 0
set delay to 0.5
broadcast INIT

when I receive INIT
motor 10 angle 60
wait delay secs
motor 9 angle 135
wait delay secs
motor 11 angle 90
wait delay secs
motor 5 angle 40
wait delay secs
if i < 4
    broadcast ROTATE

when I receive ROTATE
motor 5 angle 40
wait delay secs
motor 11 angle 45
wait delay secs
motor 10 angle 90
wait delay secs
motor 9 angle 75
wait delay secs
motor 5 angle 25
wait delay secs
motor 9 angle 155
wait delay secs
motor 10 angle 35
wait delay secs
motor 11 angle 135
wait delay secs
motor 10 angle 75
wait delay secs
motor 9 angle 100
wait delay secs
motor 5 angle 40
```

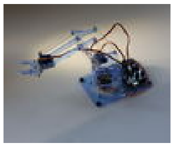
Step 38: Restore the factory default

You can restore the bot to factory default by upload meArm.ino in Arduino IDE any time. If you have any problem, please add me friend "Lien Ted" in Facebook or email me at ted99.tw@gmail.com. Or join the forum in joyarm.weebly.com.

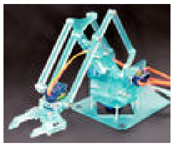
Related Instructables

meArm with joystick by ted99tw

Giant Robot Arm "meArm" Remix by Jazzmyn



MeArm - Build a Small Hackable Robot Arm by phenoptix



Pocket Sized Robot Arm #meArm V0.4 by phenoptix

Build your own ultimate open-source computerized sliding timelapse rig (video) by ritwika

DIY Facebook notifier with Arduino - Arduino Facebook Notifier (v.0.1.1) - Open Source project (video) by Apolikamixitos

Comments

1 comments **Add Comment**

Fission Chips says: Nov 23, 2014. 7:16 PM **REPLY**

This is so amazing! Your code explanation is great, the design looks so modern and innovative... you deserve a high five for your awesome work! This looks like such a great educational kit, and seems so cool I want to get it! Totally nice job.