Motor Driver for micro:bit

This is a motor driver module designed for the BBC micro:bit. Can drive 2 DC motots and 3 steering gears

Resources

- Schematic

Datasheet

- RT9193
- TB6612FNG

Demo Code

- Demo Code (Python)

Software

- makecode website
- mu

Features

- micro:bit edge connector, directly pluggable
- Onboard connectors for two DC motors and three steering gears
- 5V regulator, supports 6V~12V input (VIN terminal). Output current up to 3A
- Onboard connectors for DC motor and standard steering gear.
- Comes with developing resources and munual

Specification

- Power input range:6V~12V
- Gear voltage: 5V
- Logic voltage: 3.3V
- Driver: TB6612FNG
- Dimension: 65mm x 36mm
- Holes size: 3.0mm

Pinouts

PIN	Description

Vcc	Power (6~12V)
GND	Ground
A1	positive pole of motor A
A2	negative pole of motor A
B1	positive pole of motor B
B2	negative pole of motor B
P0	control pin P0 of steering gear
P1	control pin P1 of steering gear
P2	control pin P2 of steering gear

Programming Guide

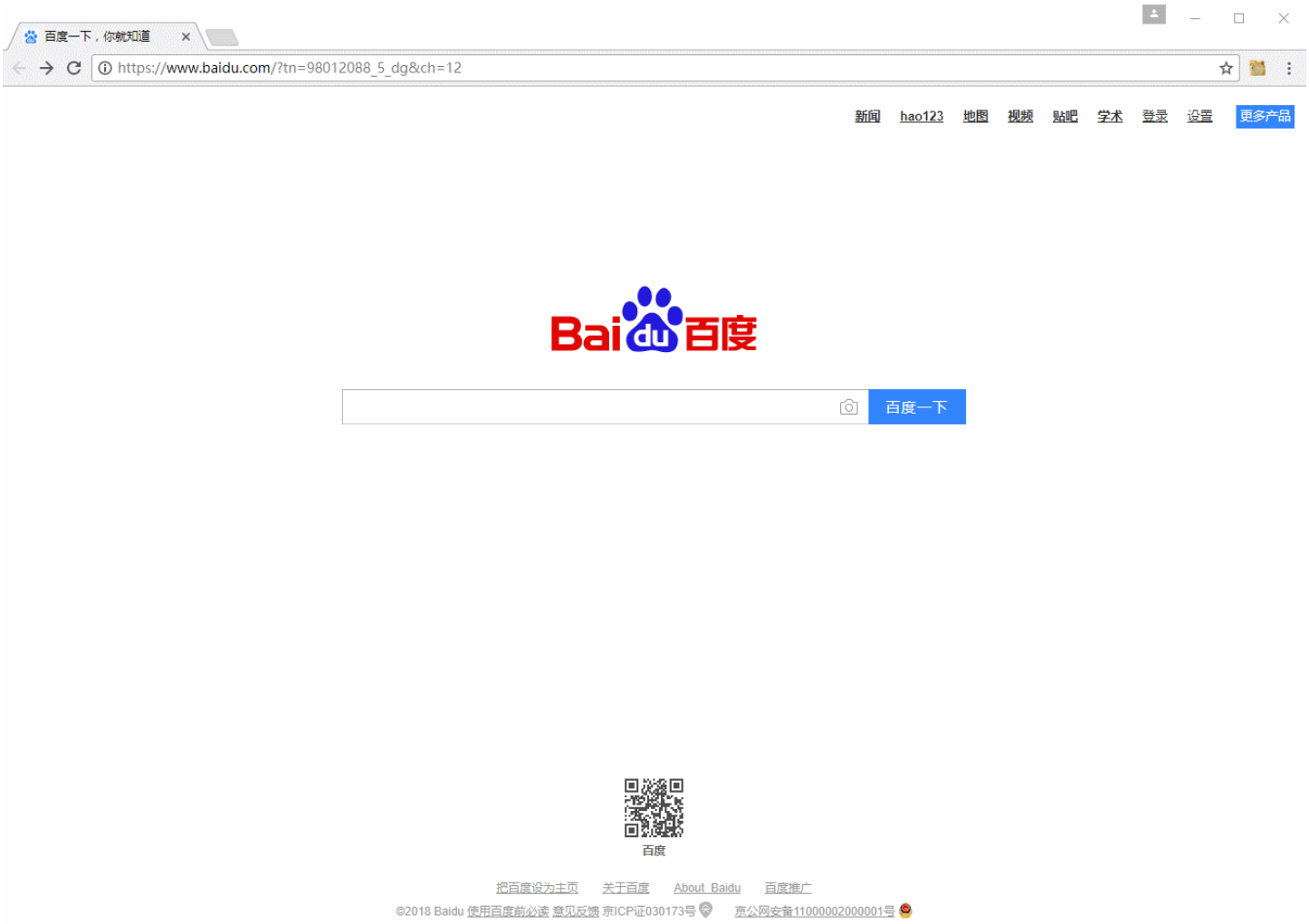
Micor:bit has variety of programming methods like mbed, micropython, typescript and other programming methods, as well as code online websites which are abundant.

The official recommendation are two programming methods: typescript and micropython. Typescript is the graphical programming language.

Typescript

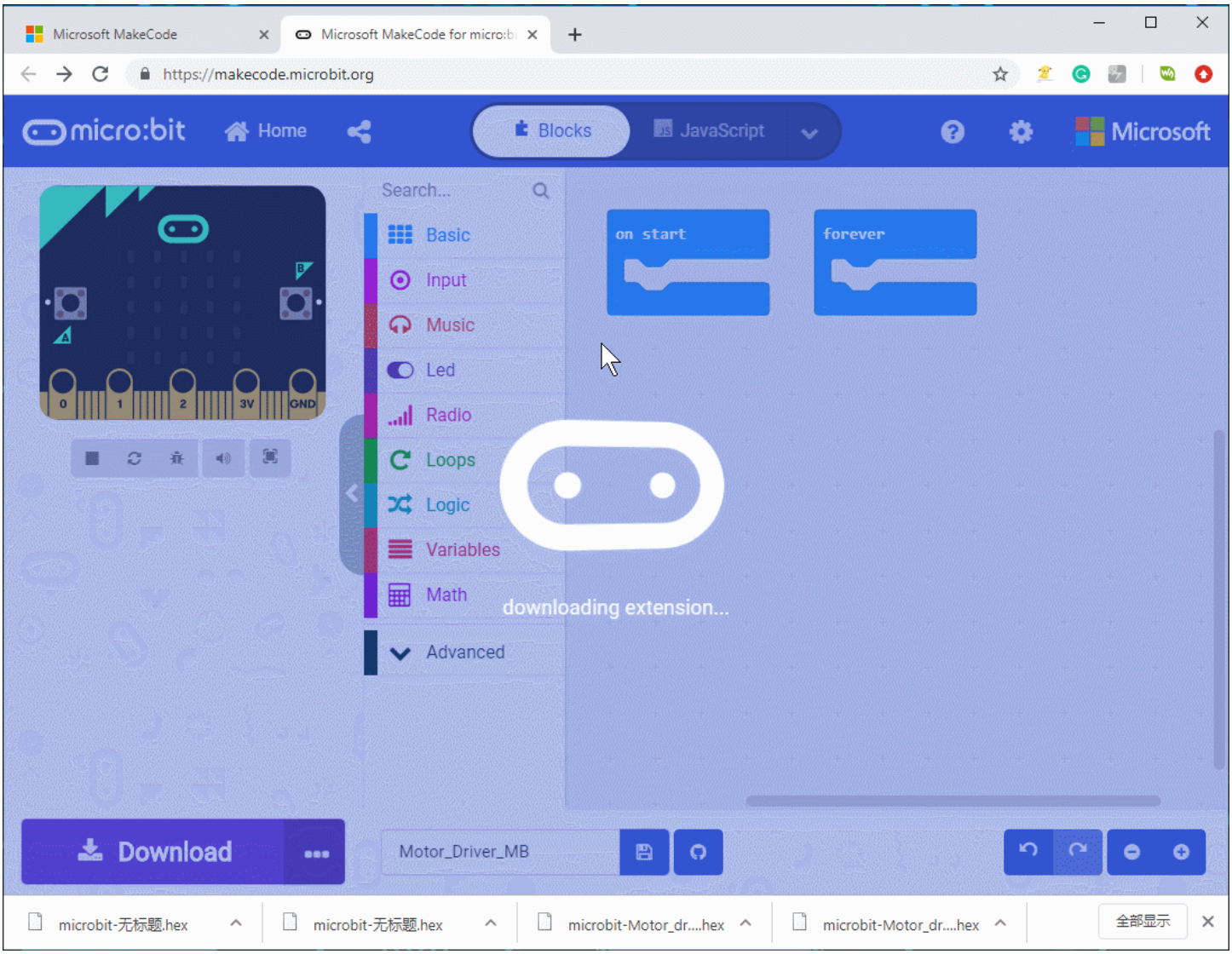
Typescript is a kind of graphical programming, its website is that: https://makecode.microbit.org/#

- Open a browser and input the next URL



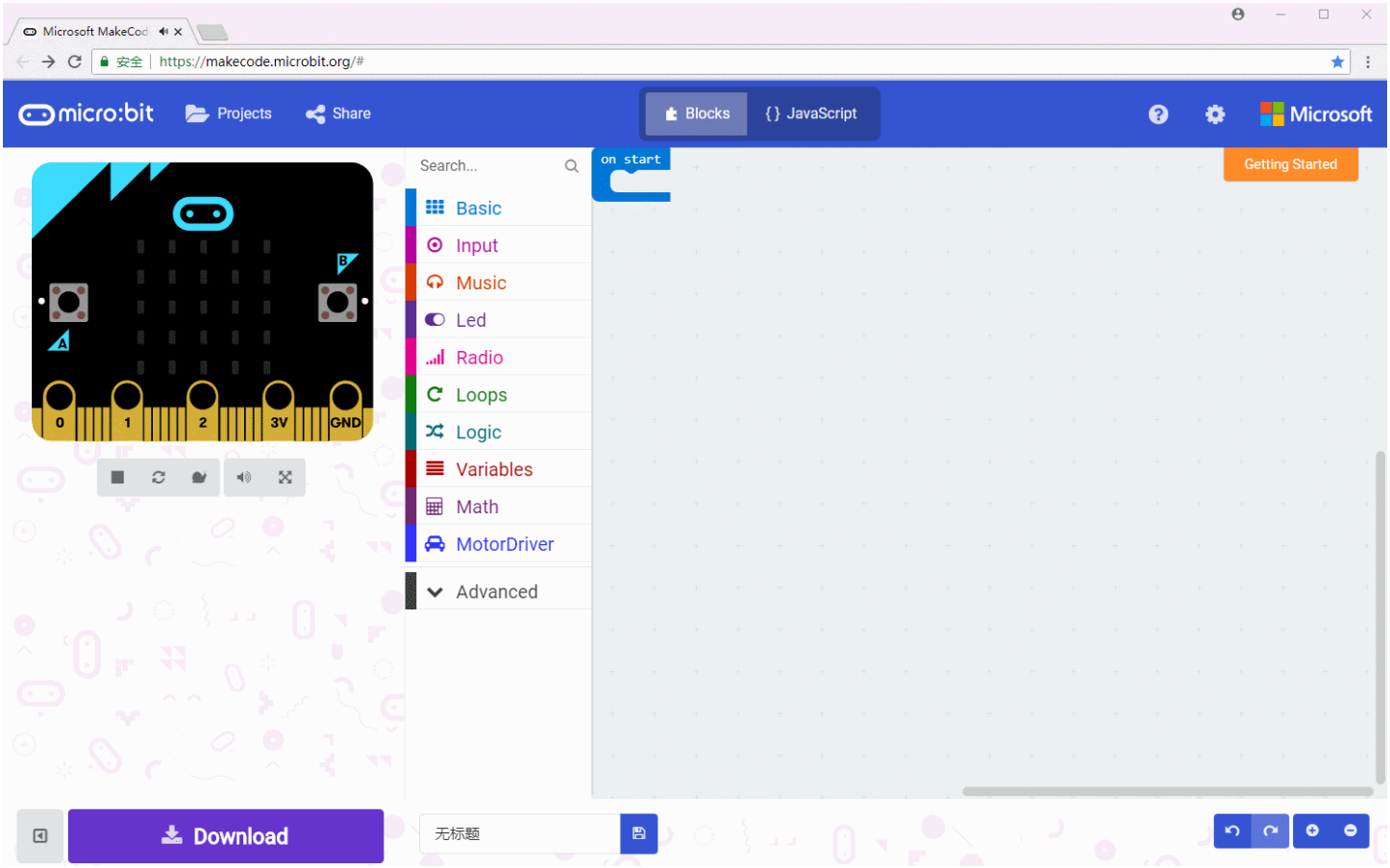
- Add Packeage

You can clone the package of Motor Driver HAT for micro;bit from github. Click More..->Add Package, then copy the URL to the Edittext. https://github.com/waveshare/pxt-Motor

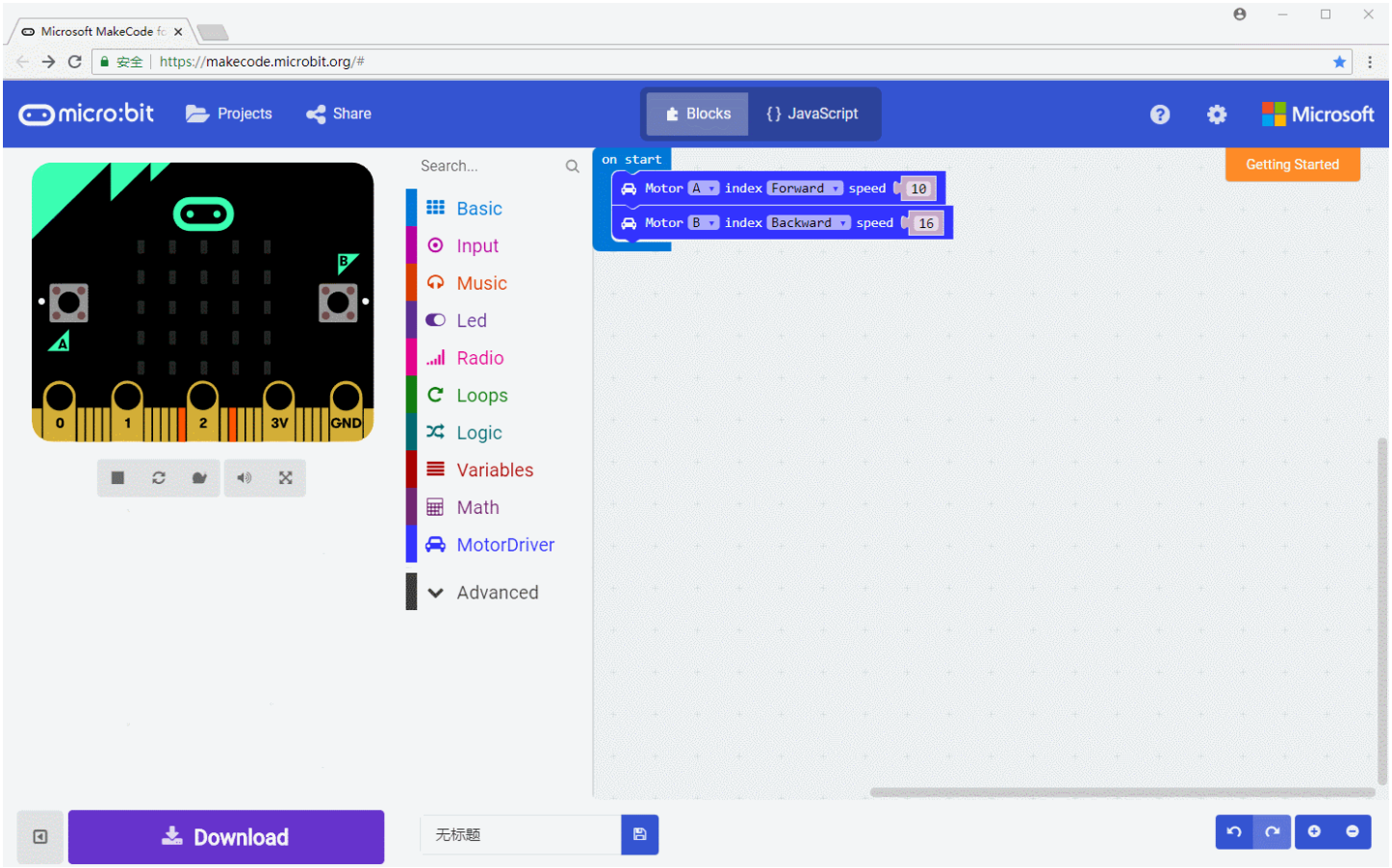


Control Motor Rotation

Control two motors at the same time, and set their rotating direction and speed

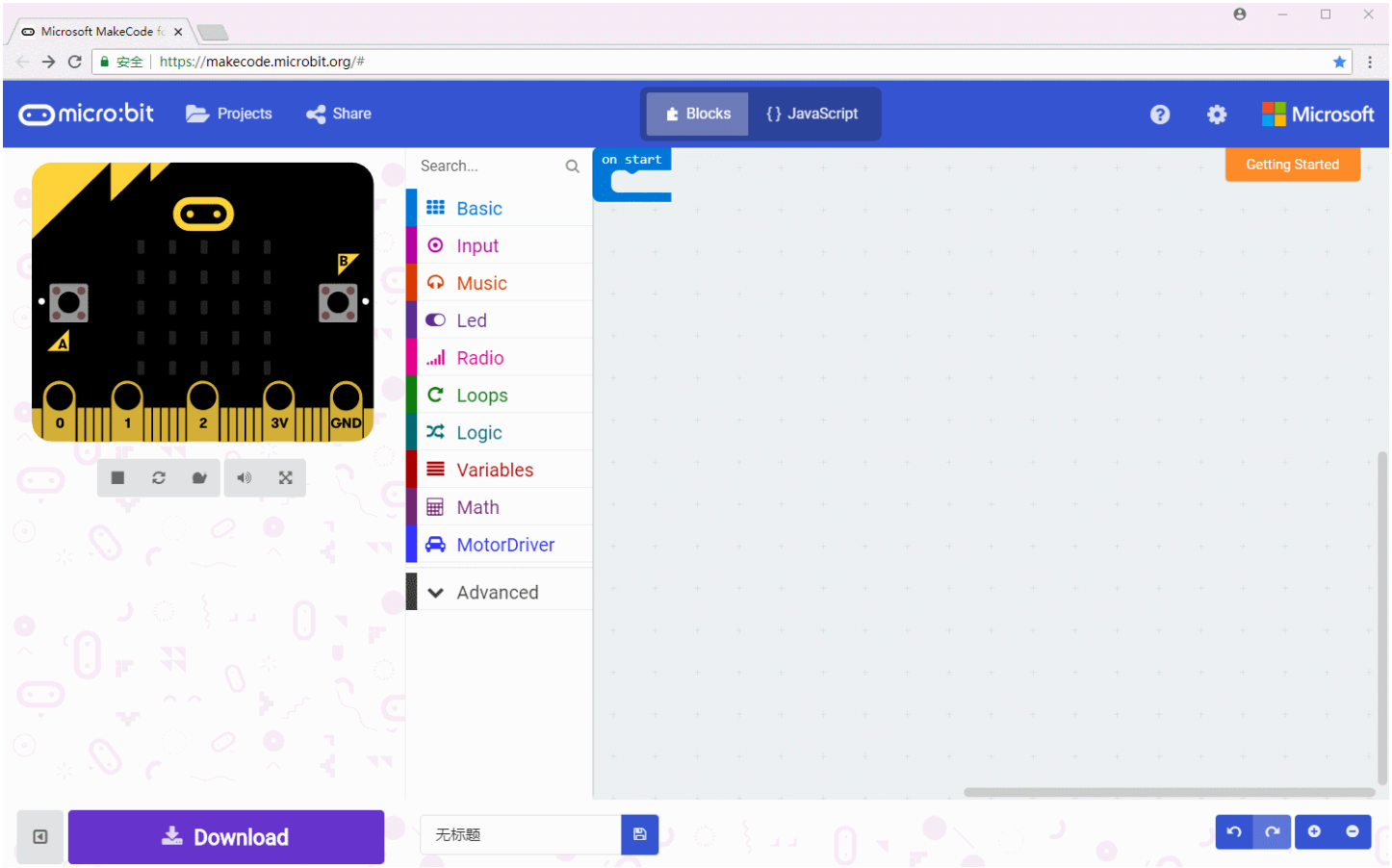


Stop motor when button A is pressed



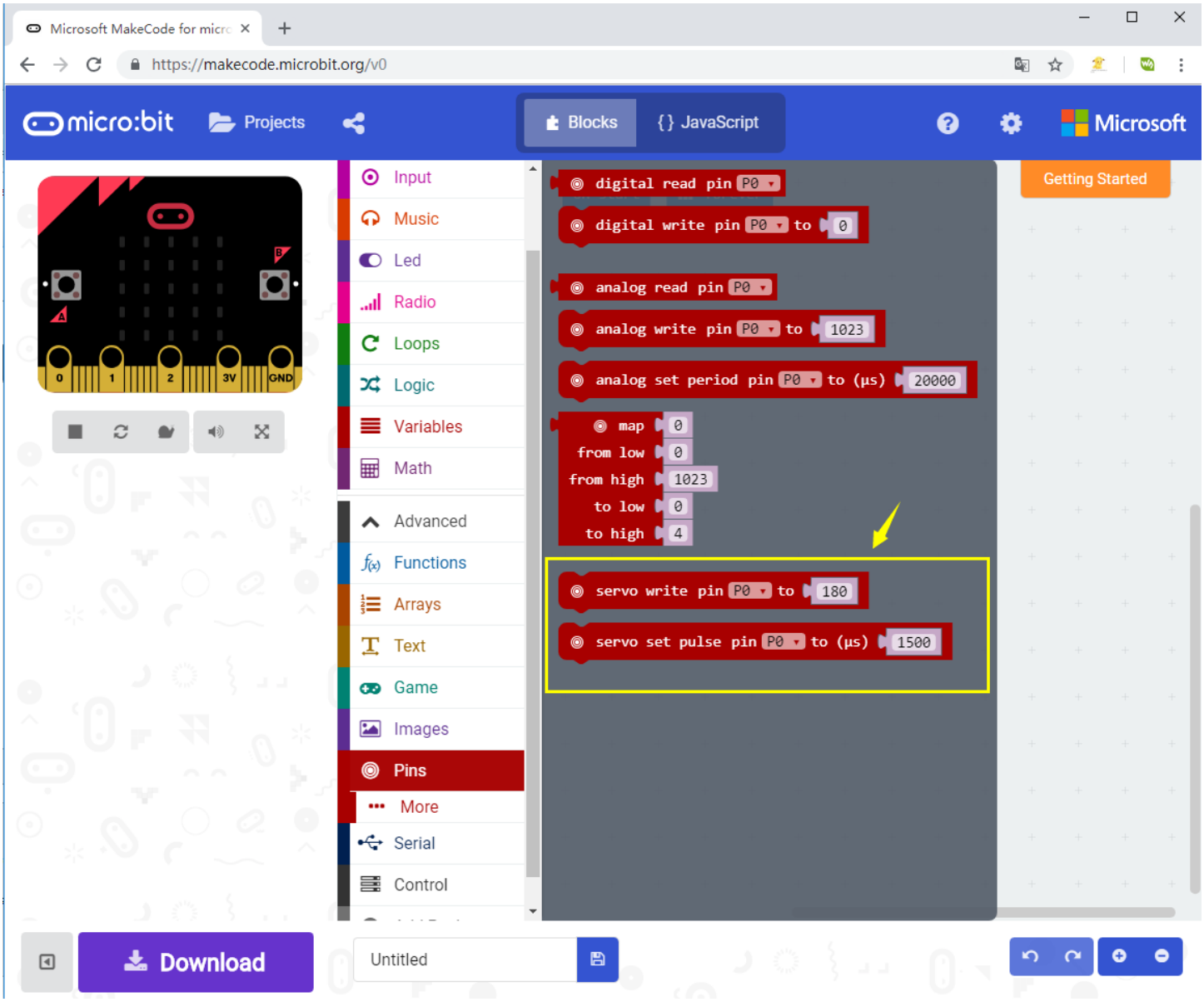
Control Servo

Control servo to rotate in certain angle.



Other blocks

There are also some block in Avandced->More which can be used to control servo



Python

You can get more information about python programming of micro:bit on this web page: https://microbit-micropython.readthedocs.io/en/latest/index.html

We provide python codes for this module, you can download and have a try.

Unzip the demo code you download and open the file motor.py which is under the python directory with software mu.

Initiailing

```
class MotorDriver():
    def __init__(self):
        self.PWMA = pin8
        self.AIN1 = pin13
        self.AIN2 = pin12
        self.PWMB = pin16
        self.BIN1 = pin14
        self.BIN2 = pin15
        self.S0 = pin0
        self.S1 = pin1
        self.S2 = pin2
        self.S0.set_analog_period(20)
        self.S1.set_analog_period(20)
        self.S2.set_analog_period(20)
```

Control Motor

```
def MotorRun(self, motor, index, speed):
    if(speed > 16):
        return
    speed = speed * 64 - 1

    if(motor == 0):
        self.PWMA.write_analog(speed)
        if(index == Dir[0]):
            self.AIN1.write_digital(0)
            self.AIN2.write_digital(1)
        else:
            self.AIN1.write_digital(1)
            self.AIN2.write_digital(0)
    else:
        self.PWMB.write_analog(speed)
        if(index == Dir[0]):
            self.BIN1.write_digital(0)
            self.BIN2.write_digital(1)
        else:
            self.BIN1.write_digital(1)
            self.BIN2.write_digital(0)
```

Stop

```
def MotorStop(self, motor):
    if (motor == 0):
        self.PWMA.write_analog(0)
    else:
        self.PWMB.write_analog(0)
```

Cotnrol Servo

```
  def ServosTurnZero(self, servo):
      if(servo == 0):
          self.S0.write_analog(25)
      elif(servo == 1):
          self.S1.write_analog(25)
      else:
          self.S2.write_analog(25)

  def ServosTurnFull(self, servo):
      if(servo == 0):
          self.S0.write_analog(128)
      elif(servo == 1):
          self.S1.write_analog(128)
      else:
          self.S2.write_analog(128)

  def ServosStop(self, servo):
      if(servo == 0):
          self.S0.write_analog(0)
      elif(servo == 1):
          self.S1.write_analog(0)
      else:
          self.S2.write_analog(0)
```

Control servo to rotate in certain angle

```
 def ServoTurn(self, servo, angle):
     if(angle > 180):
         return
     temp = angle / 2 + 25
     if(servo == 0):
         self.S0.write_analog(temp)
     elif(servo == 1):
         self.S1.write_analog(temp)
     else:
         self.S2.write_analog(temp)
```

Temp=angle/2+25 is the duty ratio, its range is 0~1023.

Example:

```
Motor = MotorDriver()

# control 2 motor
Motor.MotorRun(0, 'forward', 16)
Motor.MotorRun(1, 'backward', 16)

# control 3 servo
Motor.ServoTurn(0, 50)
Motor.ServoTurn(1, 150)
Motor.ServosStop(2)
```

Motor1 moves forward, Motor2 moves backward, servo1 rotate 50 degrees, servo2 rotate 150 degrees.