BBC Micro:bit Overview

แนะนำบอร์ด BBC Micro:bit ในเบื้องต้น และซอฟต์แวร์สำหรับฝึกเขียนโปรแกรม

บอร์ดไมโครบิต

Python (ไพธอน) เป็นภาษาคอมพิวเตอร์หนึ่งในหลายภาษาที่ได้รับความนิยมอย่างมาก ได้เริ่มเผยแพร่มาตั้งแต่ราวปีค.ศ. **1991** ถูกพัฒนาขึ้นครั้งแรกโดย **Guido Van Rossum** และหลายคนก็คิดว่า เป็นภาษาคอมพิวเตอร์ที่ง่ายต่อการเรียนรู้สำหรับผู้เริ่มต้น

ไพธอนเป็นซอฟต์แวร์ประเภท **Open Source** มีการแบ่งออกเป็นสองเวอร์ชันคือ **Python 2** และ **Python 3** (แต่สำหรับ **Python 2** ไม่มีการพัฒนาต่อไป ตั้งแต่ 1 มกราคม ค.ศ. **2020**)

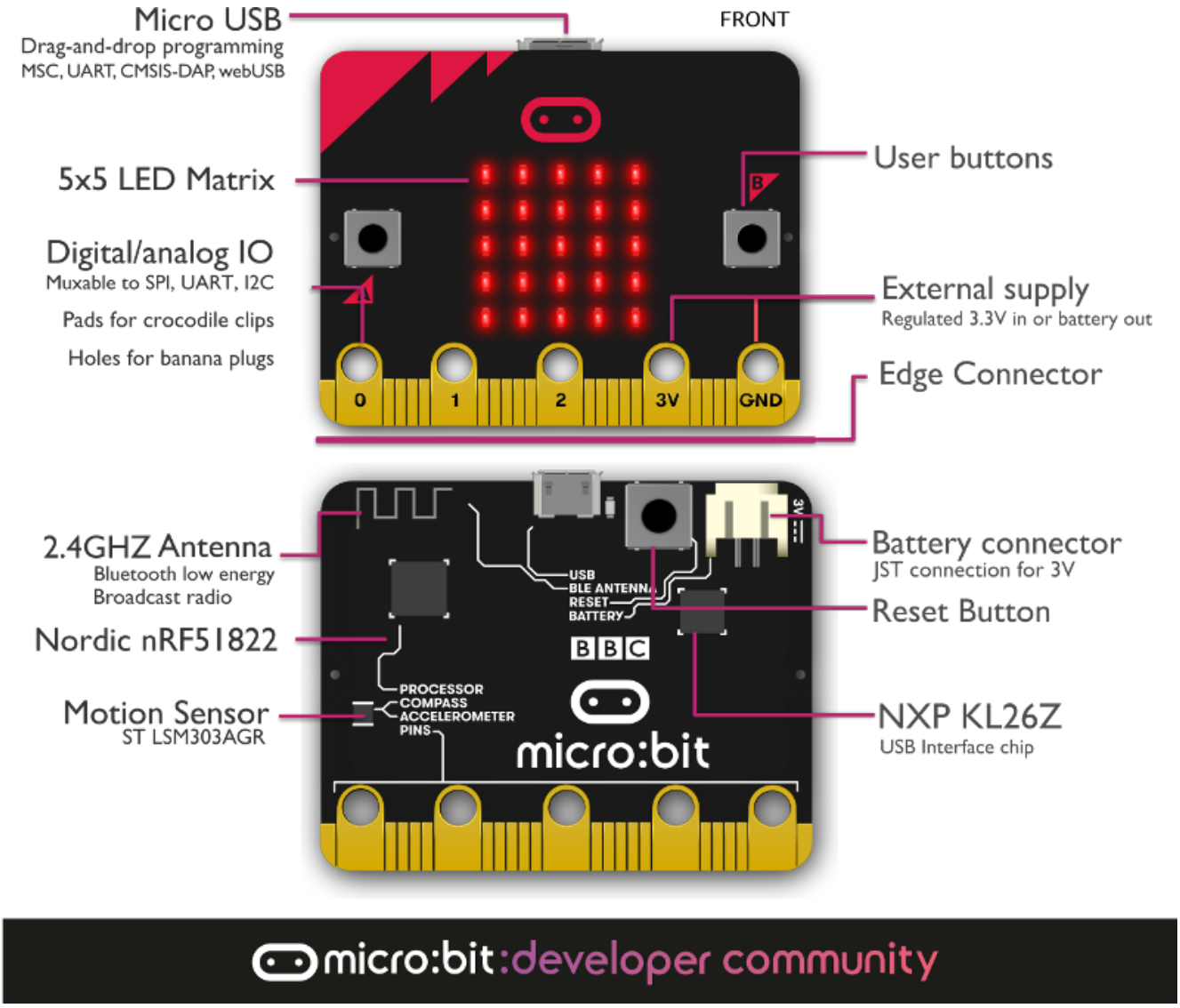
ในปัจจุบันก็มีตัวเลือกหลากหลายสำหรับซอฟต์แวร์ประเภท **IDE** ที่รองรับการเขียนโค้ดภาษานี้ และใช้งานได้ฟรี ทั้งแบบ **Offline** และ **Online** (**Web-based/Cloud-based IDE**)

คำถาม: ถ้าอยากจะใช้ภาษา **Python 3** สำหรับการเขียนโค้ดเพื่อใช้งานสำหรับไมโครคอนโทรลเลอร์ จะเป็นไปได้หรือไม่ ?

ในเดือนมีนาคมปีค.ศ. **2015** ทาง **BBC** (**British Broadcasting Corp.**) ซึ่งเป็นสถานีโทรทัศน์สื่อสาธารณะของสหราชอาณาจักร ได้เปิดตัวโครงการบอร์ด **BBC Micro:bit** (ไมโครบิต) ภายใต้ชื่อ **BBC's Make It Digital Campaign** โดยมีวัตถุประสงค์ เพื่อส่งเสริมการเรียนรู้ด้านโค้ดดิ้งและวิทยาการคำนวณให้แก่เยาวชน และมีการแจกจ่ายบอร์ดไปยังโรงเรียนต่าง ๆ ในประเทศอังกฤษ

บอร์ดไมโครบิต มีชิป **nRF51822** (**32-bit ARM Cortex-M0**) ของบริษัท **Nordic Semiconductor** เป็นตัวประมวลผลหลัก มีหน่วยความจำแบบ **Flash** ขนาด **256 KB** หน่วยความจำแบบ **SRAM** ขนาด **16 KB** และใช้ความเร็วในประมวลผลที่ความถี่ **16 MHz**

บอร์ดไมโครบิตใช้แรงดันไฟเลี้ยงจากพอร์ต **microUSB (5V)** แต่วงจรบนบอร์ดนั้นทำงานที่ระดับแรงดัน **3.3V** หรือจะใช้แบตเตอรี **3V (2x 1.5V)** ต่อเข้าที่ช่อง **Battery Connector** (ระวังการต่อกลับขั้ว + และ -) เป็นแหล่งจ่ายไฟเลี้ยงแทน **USB** ก็ได้ (ศึกษาข้อมูลเพิ่มเติมได้จาก **Micro:bit Power Supply**)



รูปภาพ: BBC Micro:bit Board

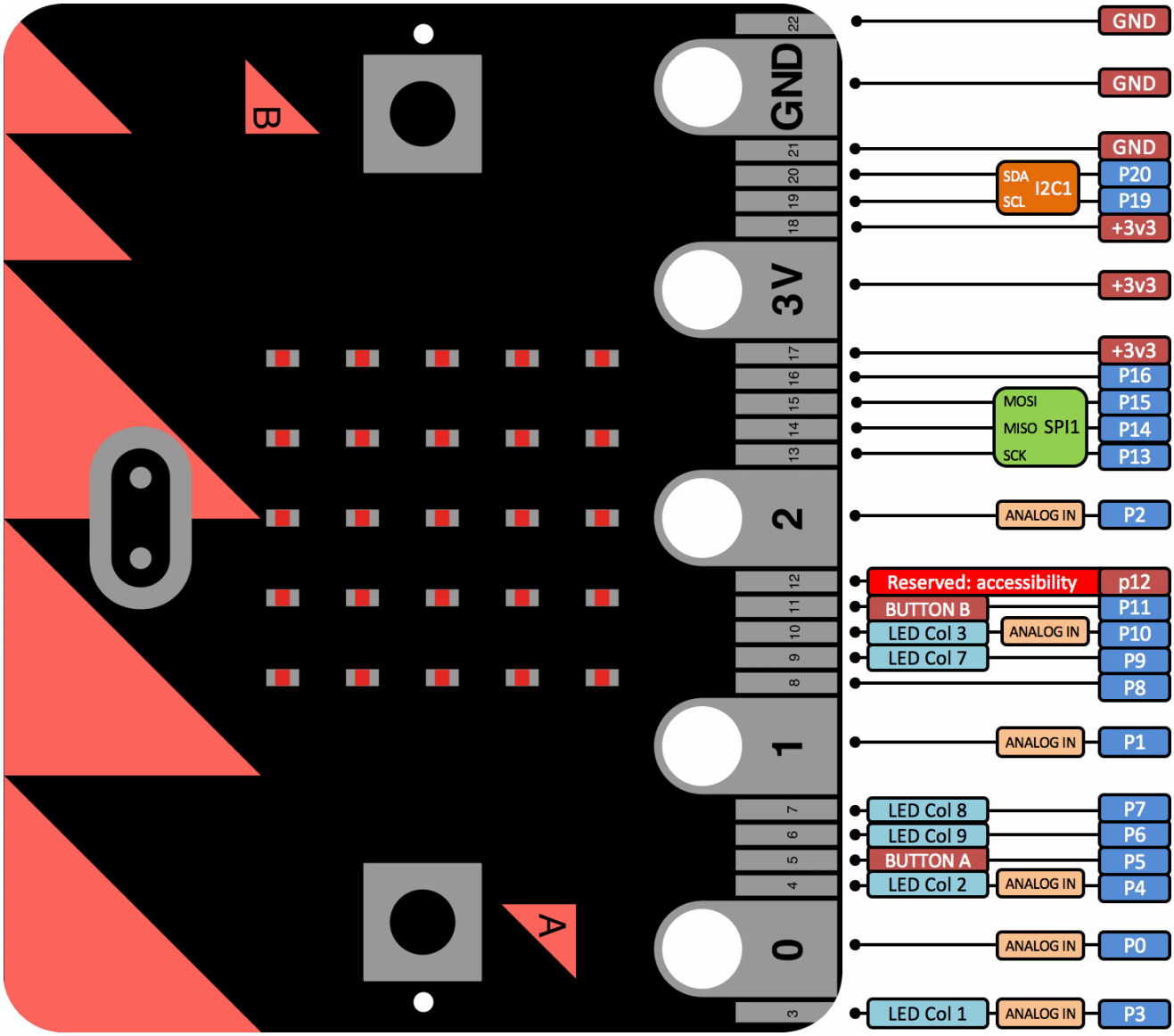
ผู้อ่านสามารถศึกษาข้อมูลเพิ่มเติมเกี่ยวกับรายละเอียดในเชิงฮาร์ดแวร์ของบอร์ดไมโครบิตได้จาก https://tech.microbit.org/hardware/

Micro:Bit Schematic Files (PDF)

ทางผู้พัฒนาได้แชร์ไฟล์ **Schematic** เช่น สำหรับบอร์ดเวอร์ชัน **v1.3B** และ **v1.5** (ให้ลองศึกษาฝังวงจรของบอร์ดไมโครบิต เพื่อเป็นกิจกรรมการเรียนรู้ด้านระบบสมองกลฝังตัว)

Schematic File v1.3B	SCH_BBC-Microbit_V1.3B.pdf - 1005KB
Schematic File v1.5	SCH_BBC-Microbit_V1.5.pdf - 1021KB

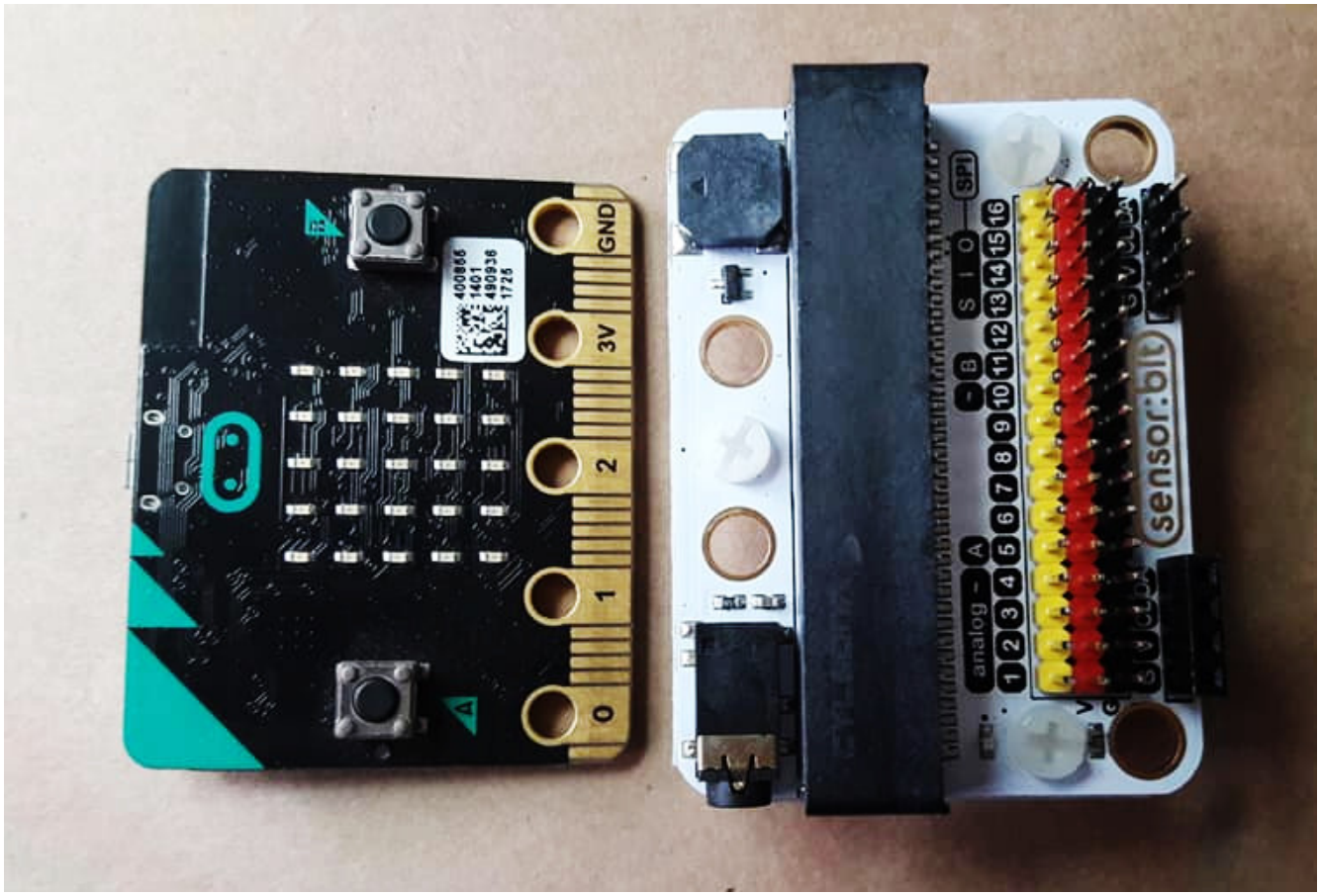
ข้อสังเกต: ขอบด้านหนึ่งของบอร์ดไมโครบิต เป็นบริเวณที่เรียกว่า **Edge Connector** และมีทั้งสองด้าน (**Front & Back Side**) แต่ไม่เหมือนขา **Pin Header** แบบทั่วไป ขามีลักษณะเป็น **Pads** มีสองขนาด ขาที่มีขนาดใหญ่กว่าปรกติ และมีการเจาะรูขนาด 4 มม. ได้แก่ **0**, **1**, **2** (หรือ **P0**, **P1**, **P2**) ซึ่งเป็นขา **I/O** (เรียกว่า **Touch Pads**) และ **3V** กับ **GND** สามารถนำใช้เพื่อจ่ายแรงดันไฟเลี้ยง **+3.3V** ให้อุปกรณ์อื่นได้ (แต่ต้องใช้ปริมาณกระแสไม่มาก โดยรวมไม่ควรเกิน **100 mA**)



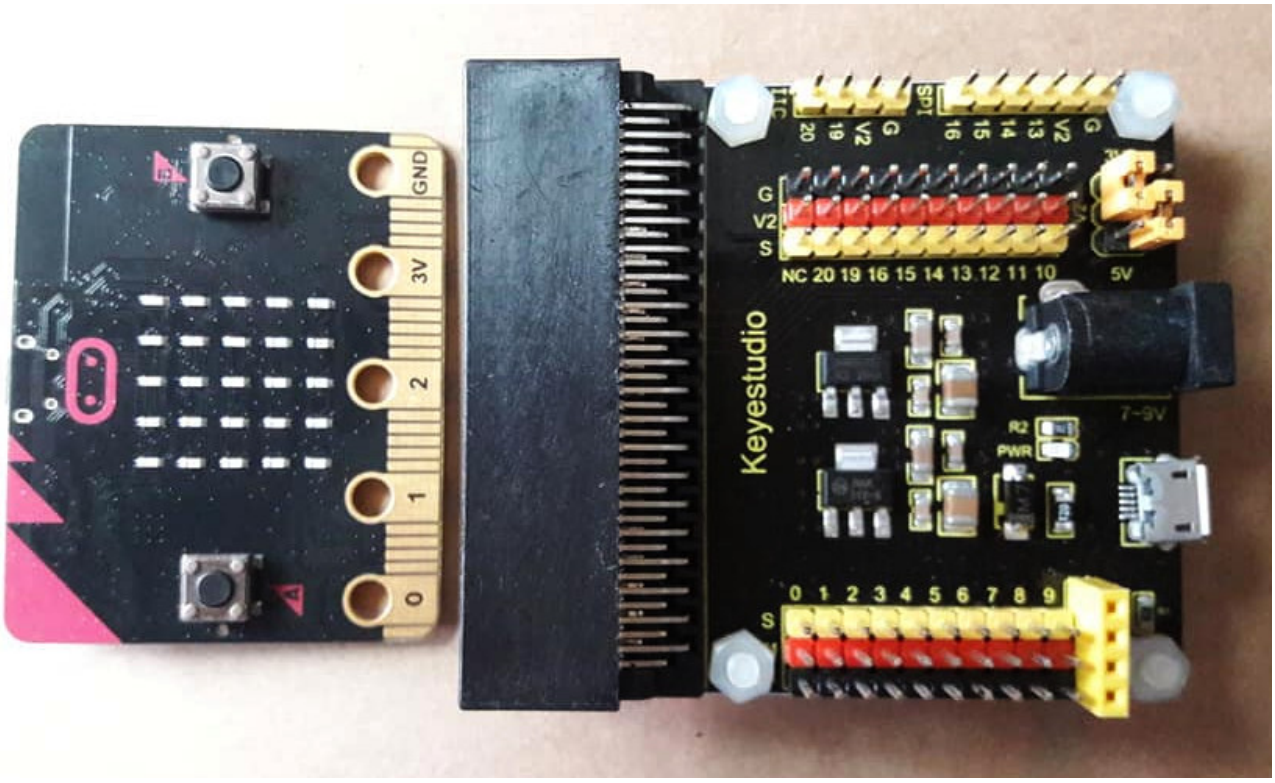
รูปภาพ: Micro:bit Edge Connector & Pins

ข้อสังเกต: ขา **P0**, **P1**, **P2** (หรือ **PAD1**, **PAD2**, **PAD3**) สามารถใช้เป็นขา **Touch Input** ได้ เนื่องจากจะมีตัวต้านทานขนาด **10 เมกกะโอห์ม** ที่ขาเหล่านั้นต่อไปยัง **VCC (+3.3V)**

ในการต่อวงจรภายนอก อาจจำเป็นต้องใช้โมดูลเสริม เพื่อนำบอร์ดไมโครบิตมาเสียบเข้าที่บริเวณ **Edge Connector** ในปัจจุบันก็มีหลายบริษัทที่ผลิตโมดูลหรืออุปกรณ์เสริมประเภทนี้ออกมาจำหน่าย ดังนั้นควรศึกษาคู่มือการใช้งาน ก่อนนำมาใช้สำหรับการต่อวงจร



รูปภาพ: ตัวอย่างโมดูล Edge Connector (ElecFreaks sensor:bit)



รูปภาพ: ตัวอย่างโมดูล Edge Connector (ของบริษัท Keyestudio)

ตัวเลือกสำหรับการเขียนโค้ด

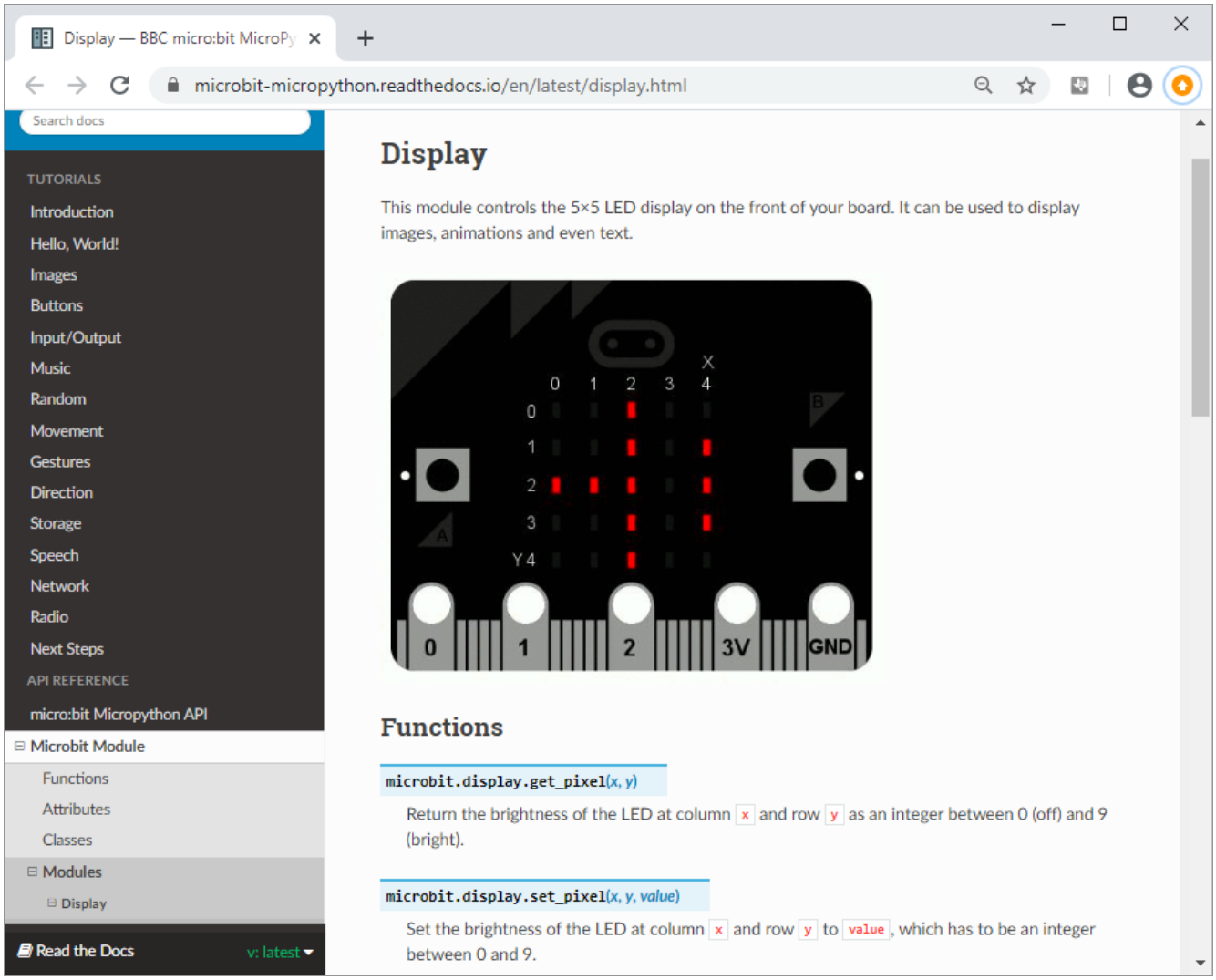
รูปแบบการเขียนโปรแกรมสำหรับไมโครบิตที่คนส่วนใหญ่รู้จักและใช้งานกัน เป็นวิธีการต่อบล็อก (**Block-based Coding**) บนหน้าเว็บเบราว์เซอร์ ซึ่งจะถูกแปลงให้เป็นโค้ดในภาษา **Static TypeScript (STS)** และไฟล์ **.hex** ได้โดยอัตโนมัติ การทำงานในส่วนนี้เป็นผลงานโดยทีมวิจัยของบริษัท **Microsoft** ภายใต้โปรเจกต์ที่มีชื่อว่า **MakeCode** หรือ **PXT (Programming Experience Toolkit)**

ข้อดีของการใช้งาน **Microsoft MakeCode** สำหรับผู้เริ่มต้นคือ สามารถใช้วิธีการเขียนโค้ดแบบต่อบล็อก ซึ่งเป็นวิธีที่ง่าย ถัดไปสามารถเรียนรู้ตัวอย่างการเขียนโค้ดด้วยภาษา **JavaScript / Static TypeScript** หรือ **Python** สำหรับไมโครบิต ช่วยให้เห็นความเชื่อมโยงระหว่างสองรูปแบบในการเขียนโค้ดได้ (**Visual** และ **Text-based**)

ในปี ค.ศ. **2015** ราวเดือนตุลาคม **Damien George** ก็ได้เผยแพร่เวอร์ชันของ **MicroPython** (ไมโครไพธอน) ที่สามารถใช้งานร่วมกับบอร์ดไมโครบิตได้ นอกจากนั้นได้มีการพัฒนา **Web App** เป็น **Online MicroPython Editor** ให้ผู้ใช้สามารถเขียนโค้ด โดยใช้เว็บเบราว์เซอร์ได้ด้วย ดังนั้นภาษาไมโครไพธอนก็เป็นอีกหนึ่งตัวเลือกที่น่าสนใจสำหรับบอร์ดไมโครบิต

ผู้ที่สนใจสามารถศึกษาเพิ่มเติมได้จากเอกสารออนไลน์ (**Online Documentation**) ที่ทางทีมผู้พัฒนาได้จัดทำไว้ เช่น

- **BBC micro:bit MicroPython API documentation**
- **BBC micro:bit MicroPython tutorial**



รูปภาพ: Online Document MicroPython API for Micro:bit

ตัวเลือกซอฟต์แวร์ IDE สำหรับเขียนโค้ด

หลังจากได้ติดตั้งไฟล์เฟิร์มแวร์ของไมโครไพธอนไว้ในหน่วยความจำ **Flash** ของบอร์ดไมโครคอนโทรลเลอร์แล้ว เราสามารถใช้วิธีสื่อสารผ่านทาง **USB-to-Serial** ระหว่างบอร์ดไมโครบิตกับคอมพิวเตอร์ของผู้ใช้ เช่น การทำคำสั่งหรือรันโค้ดผ่านทางรูปแบบที่เรียกว่า **REPL prompt (Read Evaluate Print Loop)** และการสร้างไฟล์ใหม่ แก้ไขและบันทึกไฟล์ **.py** ไว้ในระบบไฟล์ (**Flash-based File System**) ของไมโครไพธอน เป็นต้น

ในปัจจุบันก็มีหลายตัวเลือกในกลุ่มซอฟต์แวร์ประเภท **Editors / IDEs** ที่ใช้งานได้ จำแนกออกเป็นประเภท **Offline** (ต้องติดตั้งโปรแกรมในเครื่องคอมพิวเตอร์ของผู้ใช้) และ **Online** (ใช้งานผ่านเว็บเบราว์เซอร์) และในบางกรณีเราก็สามารถจำลองการทำงานของโค้ดโดยไม่จำเป็นต้องใช้ฮาร์ดแวร์จริงได้

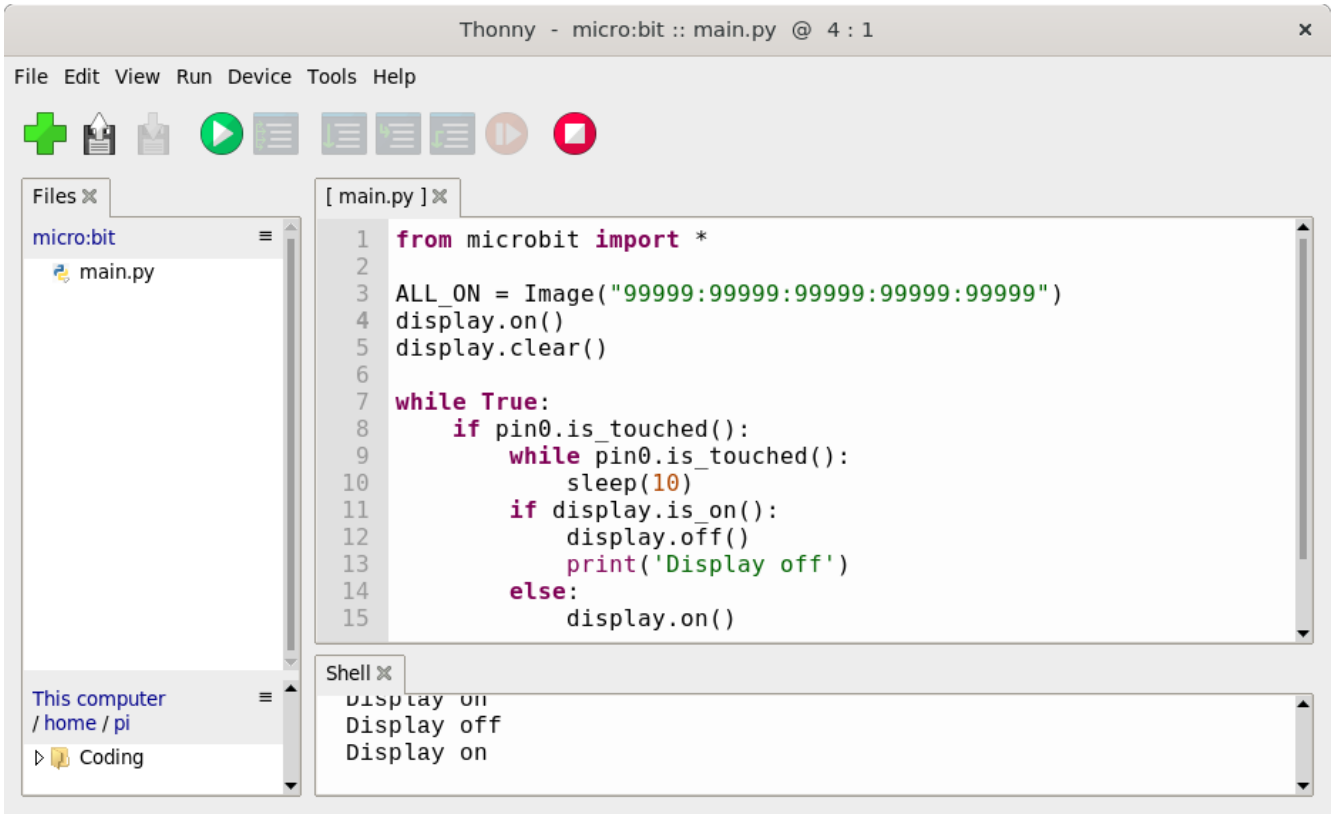
- **MicroPython Editor for Microbit (Online):** https://python.microbit.org/v/2.0
- **Microsoft MakeCode Editor (Version: beta):** https://makecode.microbit.org/beta#
- **Mu Editor**: https://codewith.mu/en/download
- **Thonny IDE**: https://thonny.org/
- **EduBlocks Python Editor (Online):** https://app.edublocks.org/#MicroBit
- **PyCharm IDE (Community edition) + Plugin:** https://plugins.jetbrains.com/plugin/9777-micropython
- **Microsoft VS Code + Device Simulator Express:** https://www.microsoft.com/en-us/garage/profiles/device-simulator-express/

ซอฟต์แวร์ที่เป็น **Web App** อย่างเช่น **Python Editor for Micro:bit** (**Source Code**) และ **Microsoft MakeCode for Micro:bit** (**Source Code**) ทำงานโดยใช้เว็บเบราว์เซอร์ เช่น **Chrome** บนเครื่องคอมพิวเตอร์ของผู้ใช้ ไม่ต้องติดตั้งซอฟต์แวร์ และเมื่อเสียบสาย **USB** เชื่อมต่อกับบอร์ด เราก็สามารถดาวน์โหลดโค้ดที่เขียนไปยังบอร์ดไมโครบิตทางผ่าน **WebUSB** ได้ค่อนข้างสะดวก

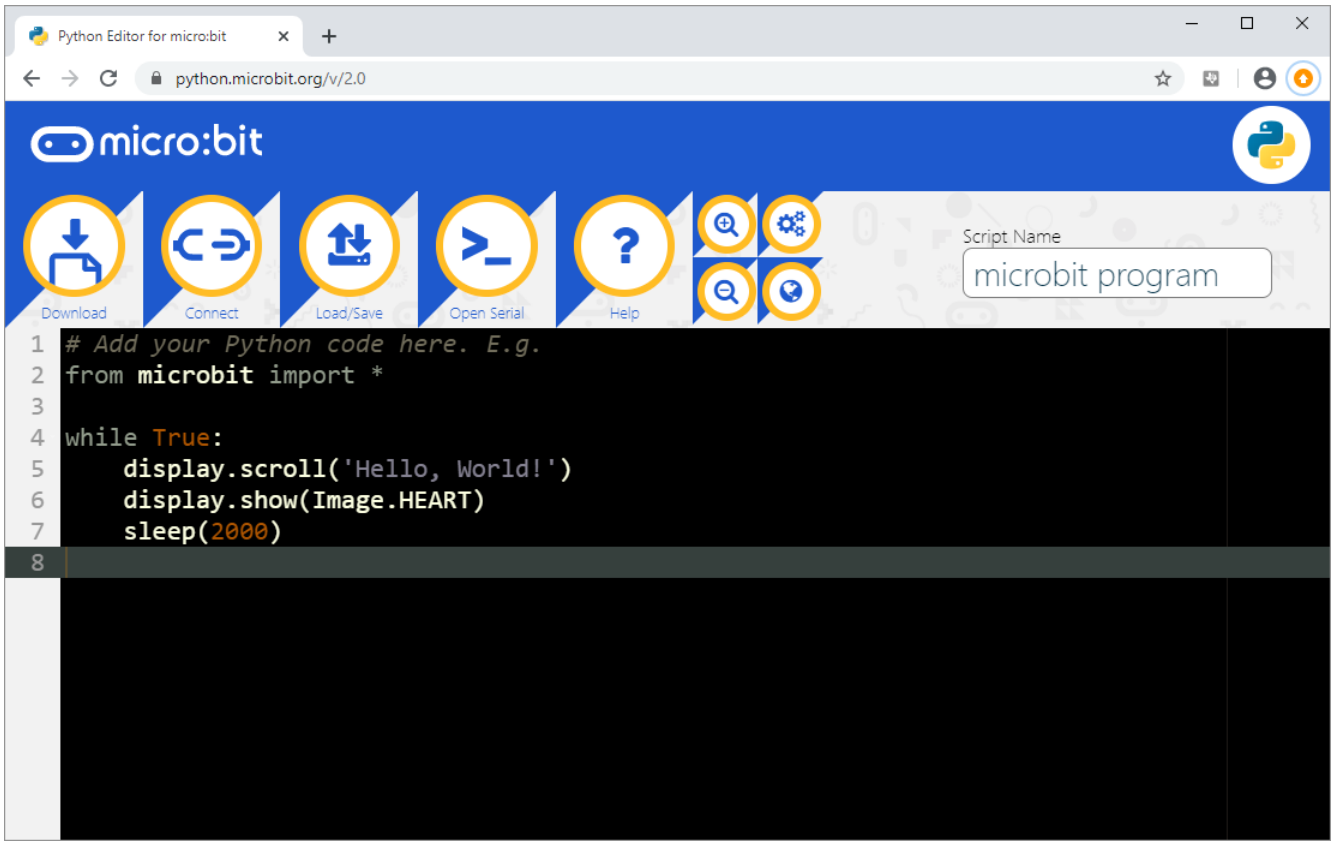
Python Editor for Micro:bit เหมาะสำหรับการฝึกเขียนโค้ดด้วยภาษา **MicroPython** และทดสอบการทำงานของโค้ดโดยใช้บอร์ดไมโครบิต ในขณะที่ **Microsoft MakeCode for Micro:bit** สามารถเขียนโค้ดด้วยวิธีการต่อบล็อก (**Block Mode**) หรือเลือกเขียนโค้ด (**Text Mode**) โดยใช้ภาษา **Python** (แต่ **MakeCode Python** ใช้ชุดคำสั่งที่แตกต่างจาก **MicroPython for Micro:bit**) หรือใช้ภาษา **Static TypeScript (STS)** อย่างใดอย่างหนึ่งได้ อีกทั้งสามารถจำลองการทำงานได้ด้วย (มี **built-in Simulator**)

Thonny IDE และ **Mu Editor** สามารถนำมาใช้เป็น **IDE** แบบ **Offline** สำหรับเขียนโค้ดไมโครไพธอน สามารถได้กับบอร์ดไมโครบิต บอร์ด **STM32** และบอร์ด **ESP32** เป็นต้น เพื่อทดสอบการทำงานของโค้ดด้วยฮาร์ดแวร์จริง

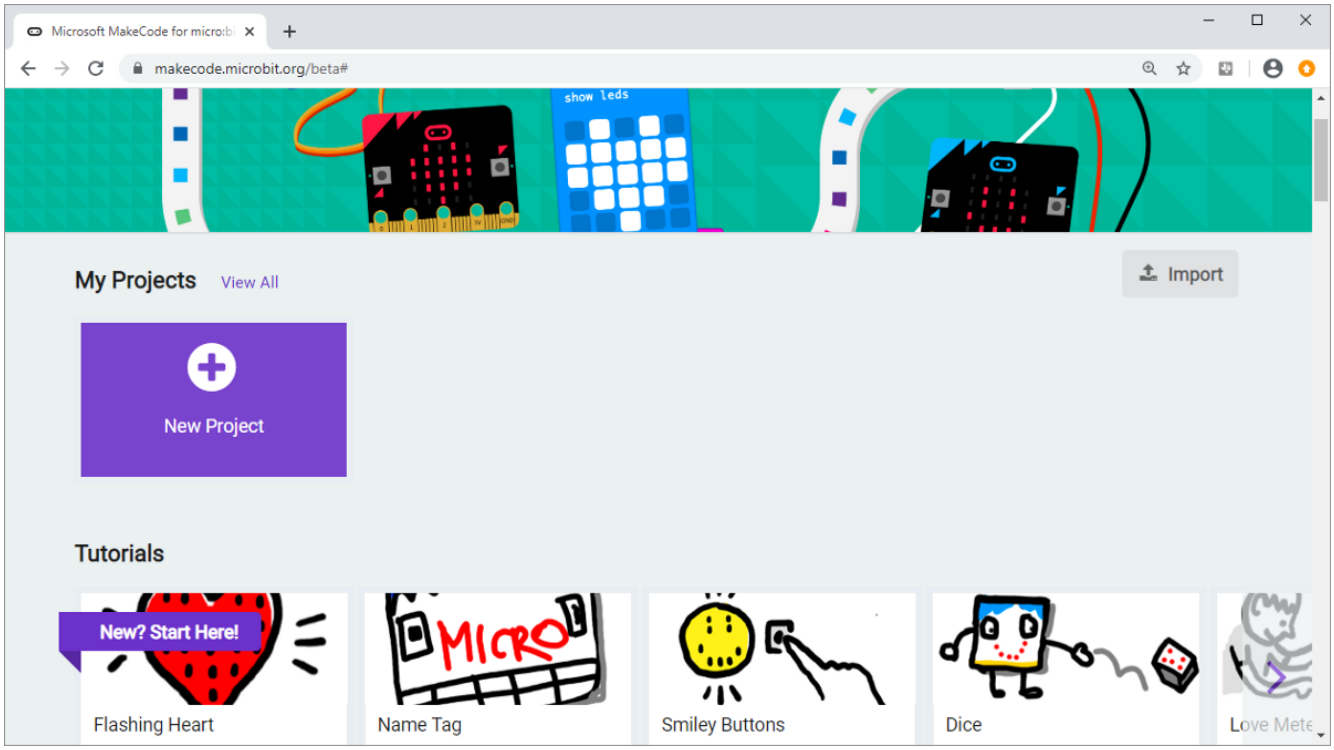
ถ้าเลือกใช้ **Microsoft VS Code + Device Simulator Express Extension** เป็น **IDE** สำหรับเขียนโค้ดไมโครไพธอน ก็อาจดูซับซ้อนกว่า แต่ก็จำลองการทำงานของโค้ดได้เช่นกัน



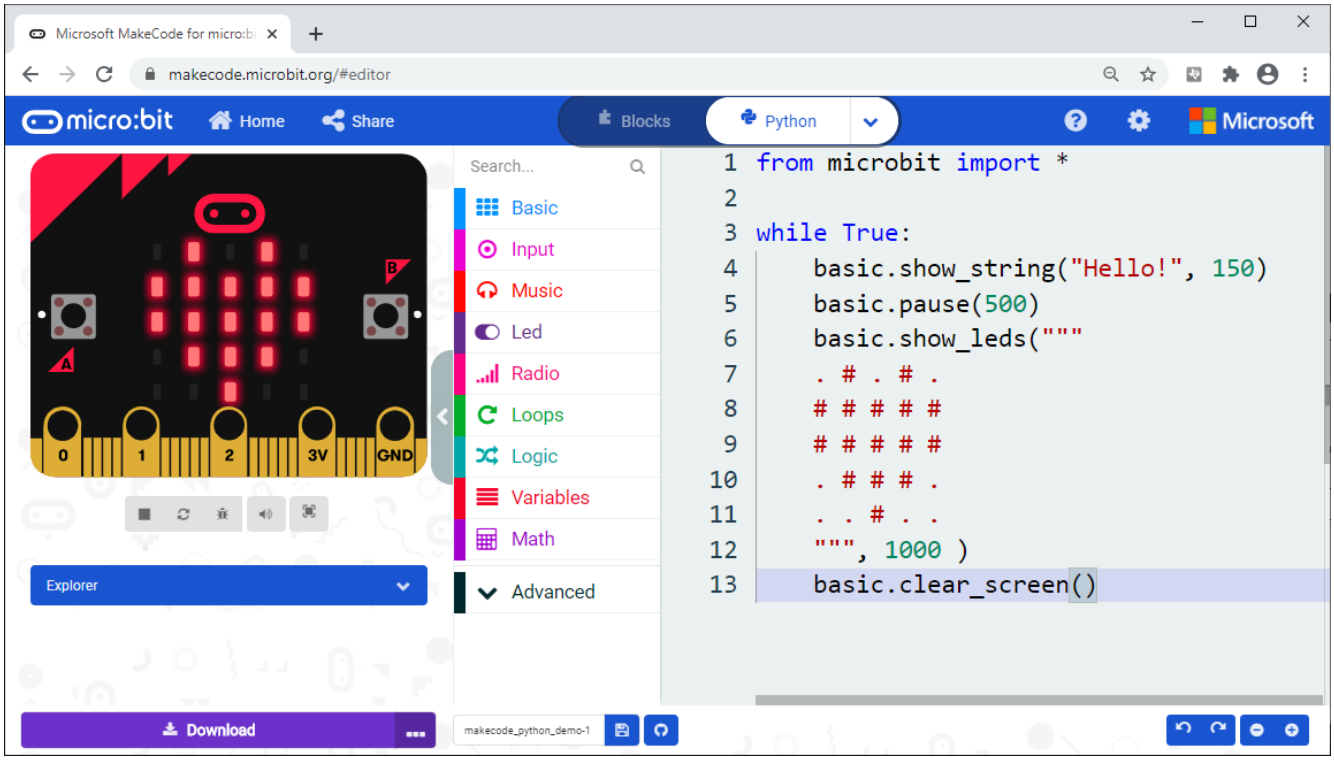
รูปภาพ: Thonny IDE Editor



รูปภาพ: Online Python Editor for Micro:bit (Version 2.0)



รูปภาพ: Online MakeCode Editor for Micro:bit (Version: Beta)



รูปภาพ: การเขียนโค้ดไพธอนสำหรับไมโครบิต โดยใช้ Online MakeCode Editor

MicroPython-Micro:bit API

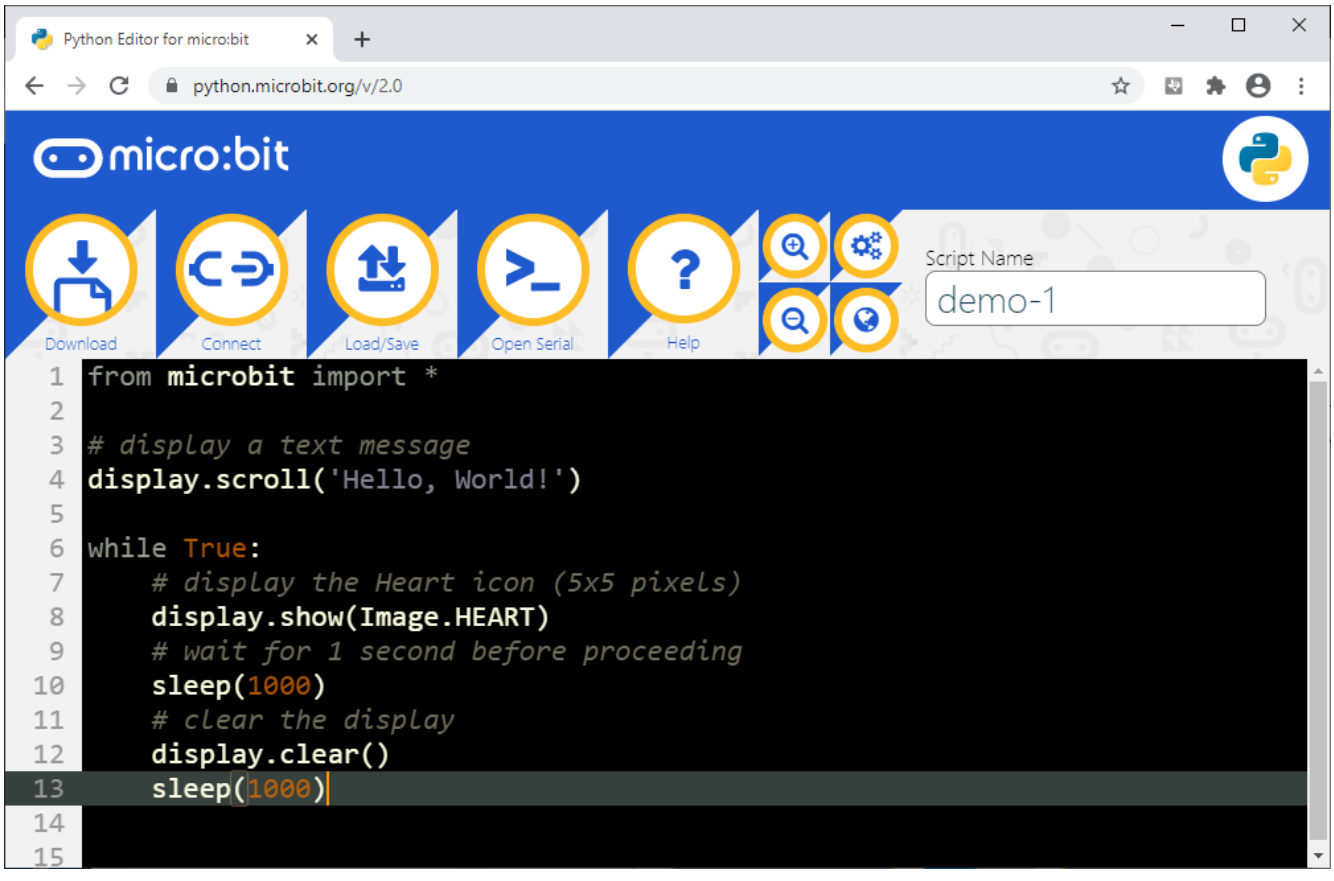
ชุดคำสั่งของไมโครไพธอนสำหรับไมโครบิต ได้มีการจัดแบ่งเป็นกลุ่มตามฟังก์ชันการทำงาน ดังนี้

- **Display** เป็นกลุ่มคำสั่งที่เกี่ยวข้องกับการแสดงผลบนแผง **5x5 LED Matrix** (**Screen**) สีแดงขนาดเล็ก (มีลักษณะเป็น **SMD LEDs**) บนบอร์ดไมโครบิต (อยู่ด้านที่เรียกว่า ด้านหน้า หรือ **Front Side**) เช่น ใช้ในการแสดงตัวเลข หรือข้อความแบบเลื่อนไป
- **Images** เป็นกลุ่มคำสั่งที่เกี่ยวข้องกับการกำหนดชื่อของรูปกราฟิกหรือไอคอนที่ได้มีการประกาศใช้งานไว้แล้ว
- **Buttons** เป็นกลุ่มคำสั่งเกี่ยวข้องกับปุ่มกดบนบอร์ดไมโครบิต (มีสองปุ่มได้แก่ **A** และ **B**) เช่น ตรวจสอบดูว่า มีการกดปุ่ม **A** หรือ **B** หรือไม่ หรือระบุว่า มีการกดปุ่มไปแล้วกี่ครั้งหลังจากที่ได้ตรวจสอบไปคราวที่แล้ว เป็นต้น
- **Input/Output** หรือ **Pins** กลุ่มคำสั่งเกี่ยวกับการใช้งานขา **I/O** ที่ **Edge Connector** ของบอร์ดไมโครบิต เช่น การกำหนดค่าให้ขาเอาต์พุต-ดิจิทัล การอ่านค่าจากขาอินพุต-ดิจิทัล การอ่านค่าจากขาอินพุต-แอนะล็อก การสร้างสัญญาณเอาต์พุตแบบ **PWM** และการตรวจสอบสถานะของขาแบบสัมผัส เป็นต้น
- **Music** เป็นกลุ่มคำสั่งเกี่ยวกับการสร้างสัญญาณเอาต์พุตให้เป็นเสียงดนตรี เมื่อนำไปต่อกับวงจรประเภทบัซเซอร์เสียง (**Sound Buzzer**)
- **Movement** การอ่านค่าจากไอซีวัดความเร่งแบบสามแกน (**Accelerometer**) ในแกน x, y, z ค่าที่อ่านได้มีหน่วยเป็น 1/1000 ของค่า g (หรือ milli-g)
- **Gestures** เป็นกลุ่มคำสั่งที่ใช้ **Accelerometer** (ไอซีตรวจวัดความเร่ง) ตรวจสอบท่าทางแบบต่าง ๆ ของบอร์ด เช่น การเอียง การวางคว่ำหรือหงายของบอร์ด การเขย่าบอร์ด เป็นต้น
- **Compass** เป็นกลุ่มคำสั่งสำหรับการอ่านค่าจากไอซีเข็มทิศแบบดิจิทัล (**Digital Compass**)
- **I2C** เป็นกลุ่มคำสั่งสำหรับการสื่อสารข้อมูลกับอุปกรณ์อื่นด้วยบัส **I2C** โดยใช้ขาสัญญาณเพียง 2 เส้นคือ **SCL (Serial Clock Line)** และ **SDA (Serial Data Line)** และสามารถนำไปใช้เชื่อมต่อเพื่อรับส่งข้อมูลกับอุปกรณ์หรือโมดูลอิเล็กทรอนิกส์ได้หลายชนิด เช่น โมดูลเซ็นเซอร์ เป็นต้น
- **SPI** เป็นกลุ่มคำสั่งสำหรับการสื่อสารข้อมูลกับอุปกรณ์อื่นด้วยบัส **SPI** โดยใช้ขาสัญญาณ 3 เส้น ได้แก่ **SCK (Serial Clock), MOSI (Master-Out-Slave-In)** และ **MISO (Master-In-Slave-Out)** โดยทั่วไปแล้วก็จะมีอัตราการรับหรือส่งข้อมูลได้สูงกว่าบัส **I2C**
- **UART** เป็นกลุ่มคำสั่งสำหรับการสื่อสารข้อมูลกับอุปกรณ์อื่นแบบบิตอนุกรม (**Serial**) แบบไม่ต้องมีสัญญาณ **Clock** ควบคุม โดยใช้ขาสัญญาณ **TX** (ข้อมูลบิตส่งออก) และ **RX** (ข้อมูลบิตรับเข้า)
- **Radio** เป็นกลุ่มคำสั่งสำหรับการติดต่อสื่อสารแบบไร้สายระหว่างบอร์ดไมโครบิตด้วย **Bluetooth** แม้ว่าบอร์ดไมโครบิตไม่ได้รองรับการใช้งานผ่าน **Wi-Fi** แต่การใช้ **Bluetooth** ก็ช่วยให้บอร์ดไมโครบิตสามารถสื่อสารกันเอง หรือสื่อสารกับคอมพิวเตอร์หรือสมาร์ทโฟนได้

การใช้งาน Python Editor for Micro:bit

มีขั้นตอนดังนี้

- ให้เปิดเว็บเบราว์เซอร์ เช่น **Google Chrome** แล้วไปที่เว็บ https://python.microbit.org/v/2.0 หรือ https://python.microbit.org/v/beta (ถ้าต้องการลองใช้เวอร์ชัน **Beta**)
- เสียบบอร์ดไมโครบิตกับคอมพิวเตอร์ด้วยสาย **USB** จากนั้นจะปรากฏ **Drive** ชื่อ **MICROBIT** ในเครื่องคอมพิวเตอร์
- ในแถบเมนูที่เป็นปุ่มกดของ **Python Editor** ให้กดปุ่ม **Connect** เพื่อเชื่อมต่อระหว่างคอมพิวเตอร์ของผู้ใช้กับบอร์ดไมโครบิต จากนั้นจะมี **Pop-up window** แสดงข้อความว่า "**BBC micro:bit CMSIS-DAP": paired** และให้กดปุ่ม **Connect** (ถ้าเชื่อมต่อได้ ชื่อปุ่มจะเปลี่ยนจาก **Connect** เป็น **Disconnect**)
- เมื่อเชื่อมต่อได้แล้ว ให้ลองใช้โค้ดตัวอย่าง และหลังจากได้เขียนโค้ดในบริเวณ **Code Editor** แล้ว ให้ตั้งชื่อไฟล์ในช่อง **Script Name** สำหรับโค้ดดังกล่าว
- กดปุ่ม **Flash** เพื่อทำการคอมไพล์โค้ด และอัปโหลดไปยังบอร์ด
- ผู้ใช้สามารถบันทึกการแก้ไขลงไฟล์ (เก็บไว้ในเครื่องคอมพิวเตอร์ของผู้ใช้) โดยกดปุ่ม **Save** หรือถ้าต้องการเปิดไฟล์แก้ไข ก็ให้กดปุ่ม **Load** เป็นต้น
- หรือจะโหลดไฟล์ **.hex** ที่ได้จากการแปลงโค้ดไมโครไพธอน มาเก็บไว้ในเครื่องคอมพิวเตอร์ของผู้ใช้ก็ได้เช่นกัน

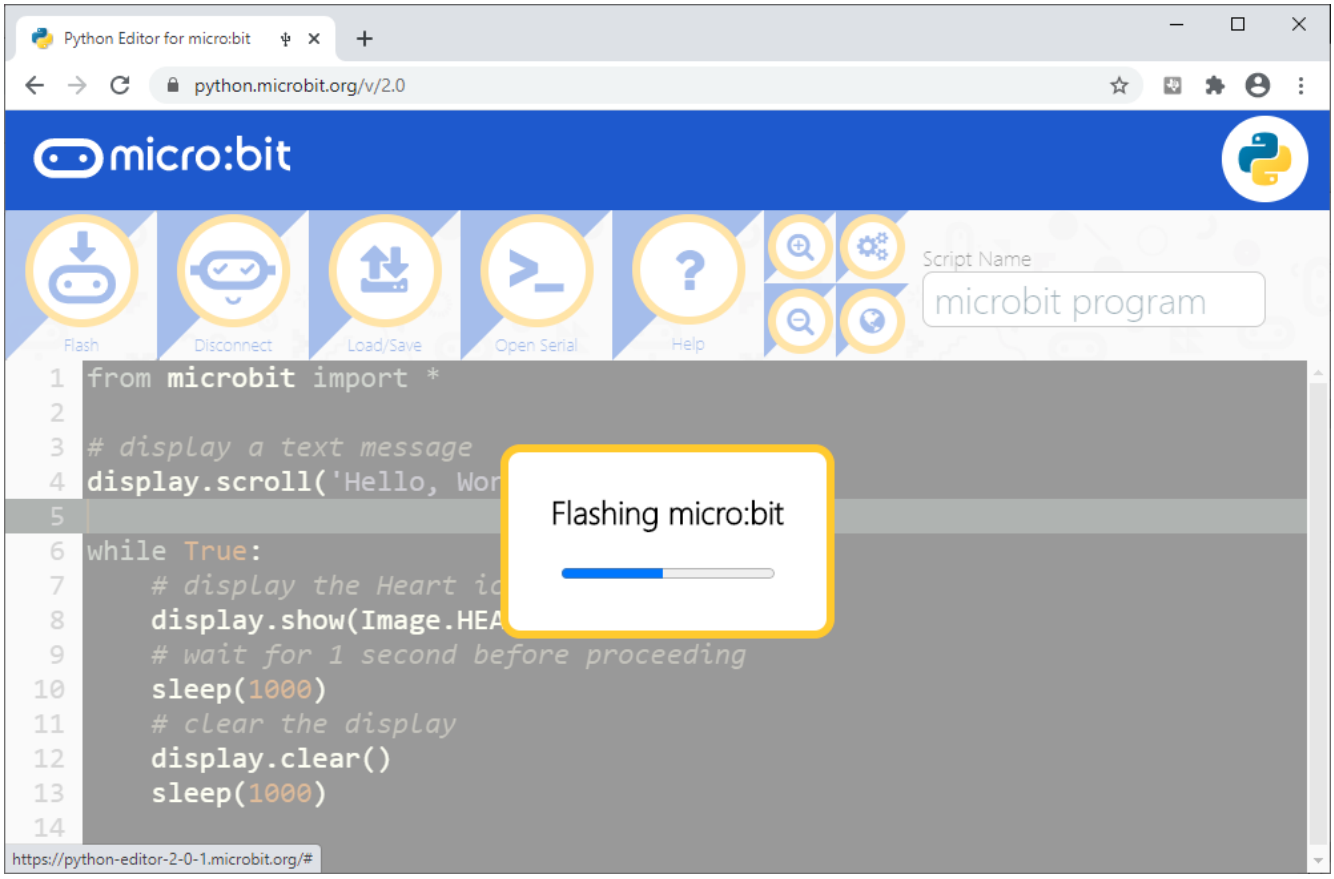


รูปภาพ: Python Editor for Micro:bit

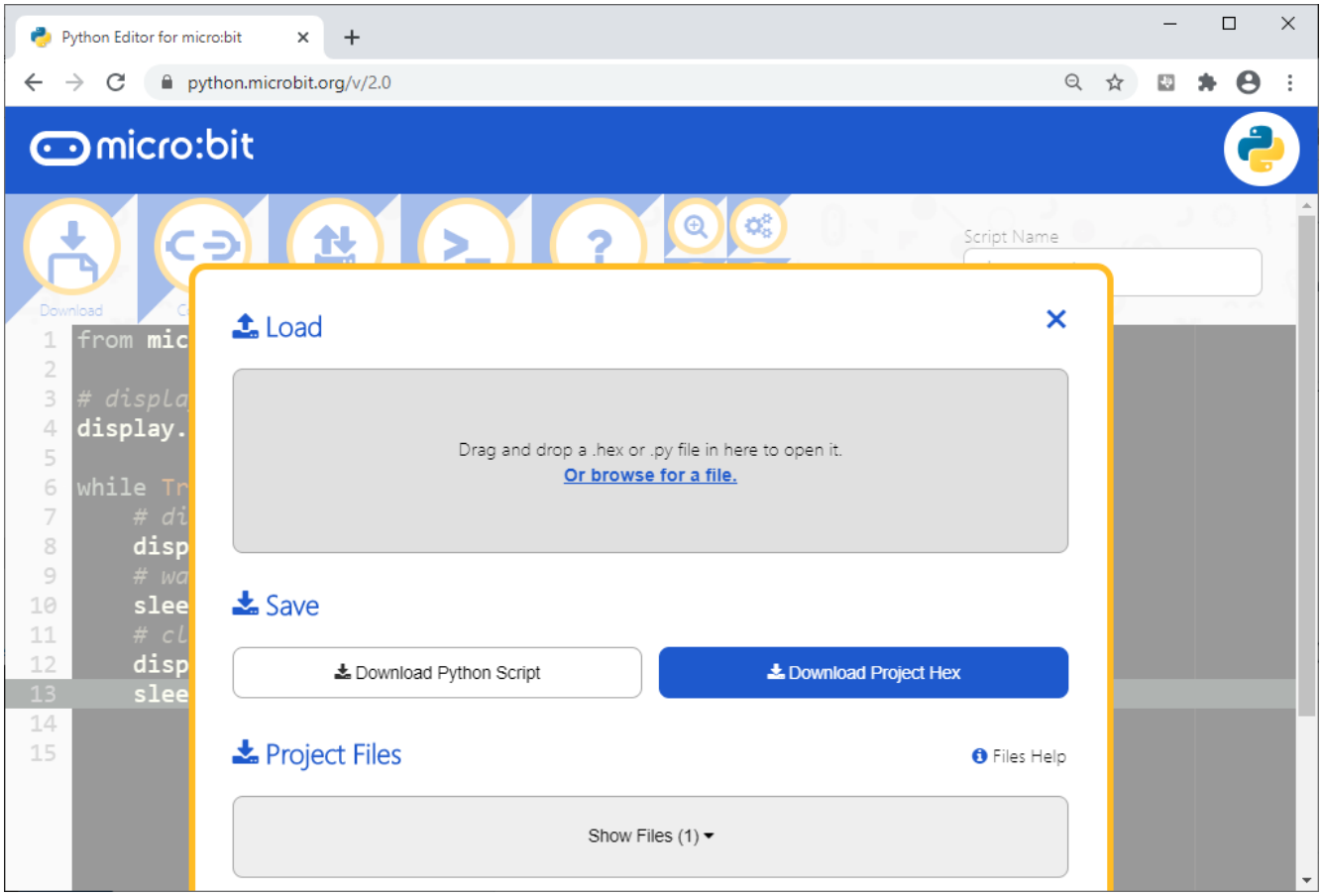
ตัวอย่างโค้ดสาธิตที่แสดงข้อความและรูปสัญลักษณ์ (หัวใจ) บน **5x5 LED Matrix Display**

```
from microbit import *

while True:
    display.scroll('Hello, World!')
    display.show(Image.HEART)
    sleep(2000)
```



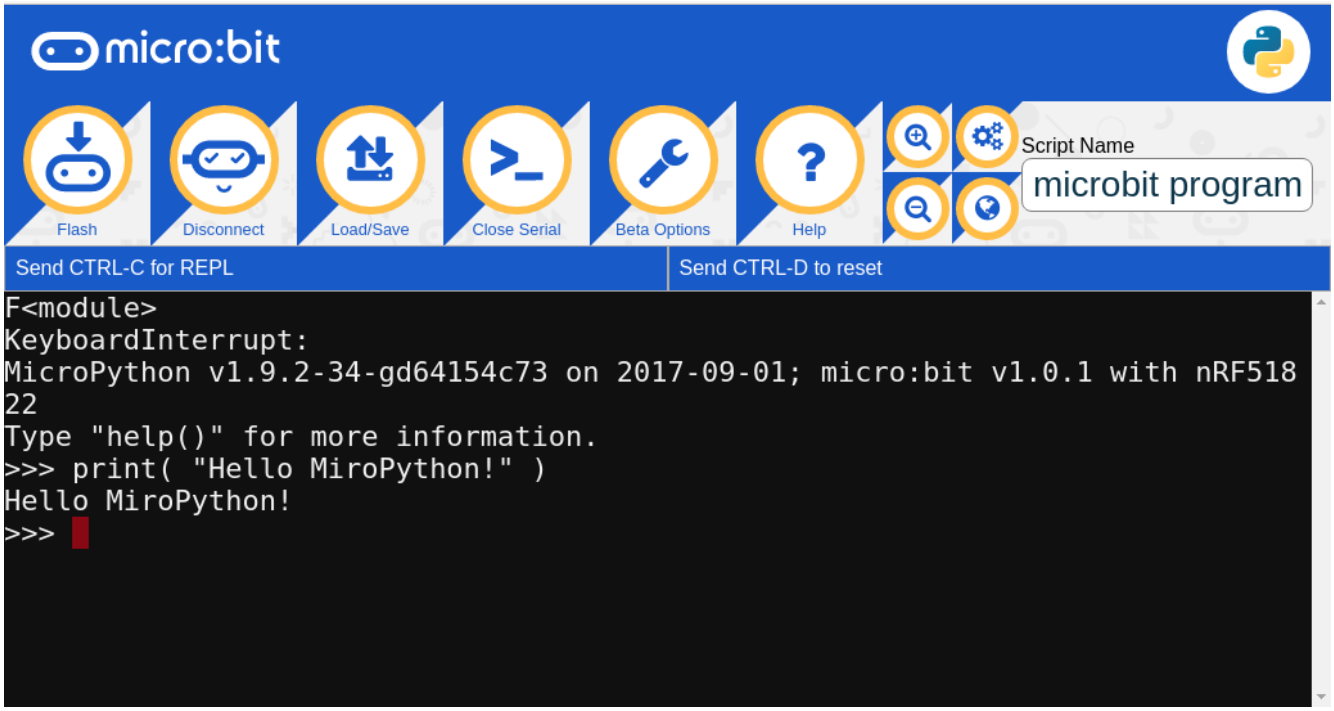
รูปภาพ: ขั้นตอนการอัปโหลดโค้ดไปยังบอร์ดไมโครบิต (Flashing)



รูปภาพ: การทำขั้นตอน Load หรือ Save File

เมื่อได้ลองใช้ **Python Editor** แล้ว ถัดไปก็ให้ศึกษาและทดลองโค้ดตัวอย่าง โดยใช้อุปกรณ์ฮาร์ดแวร์จริง (บอร์ดไมโครบิต)

เมื่อได้เชื่อมต่อกับบอร์ดและอัปโหลดโค้ดตัวอย่าง (หรือการทำขั้นตอน **Flashing** สำหรับไฟล์ .hex) ได้แล้ว ถ้าเราลองกดปุ่ม **Open Serial** จะเปลี่ยนไปหน้าสำหรับ **REPL** และถ้ากดปุ่ม **Ctrl+C** จะเข้าสู่โหมด **REPL Shell** และจะสังเกตเห็นสัญลักษณ์ **>>>** จากนั้นก็สามารถลองทำคำสั่งในภาษาไพธอนได้



รูปภาพ: ตัวอย่างการทำคำสั่งใน REPL Shell สำหรับ MicroPython

ถ้าต้องการกลับไปสู่หน้า **Code Editor** และออกจาก **REPL Shell** ก็ให้กดปุ่ม **Close Serial**

นอกจากความสามารถในการทดลองคำสั่งต่าง ๆ และรันโค้ดโดยใช้บอร์ดไมโครบิตได้ทันทีแล้ว ประโยชน์อีกประการหนึ่งของ **REPL Shell** คือ ถ้าเรารันโค้ดไพธอนแล้วเกิดปัญหา **Runtime Error** เราสามารถเปิดดูข้อความใน **REPL** ผ่านทาง **Serial** ได้ เช่น ในกรณีที่ได้เขียนโค้ดไม่ถูกต้อง (**Invalid Syntax**) จะมีการระบุหมายเลขบรรทัดของโค้ดที่ผิด เพื่อให้เราตรวจสอบและแก้ไข