Learn to Program via Visual Programming and micro:bit

A Study Notes

Outline

- Intent
- What is Visual Programming?
- Visual Programming for Learning
- IoT Devices and micro:bit
- Visual Programming for micro:bit

Intent

- To walk-thru a visual programming environment for learning multiple programming languages
  - JavaScript
  - Python

  Not cover here
  - C/C++
  - Lua, …

Visual Programming

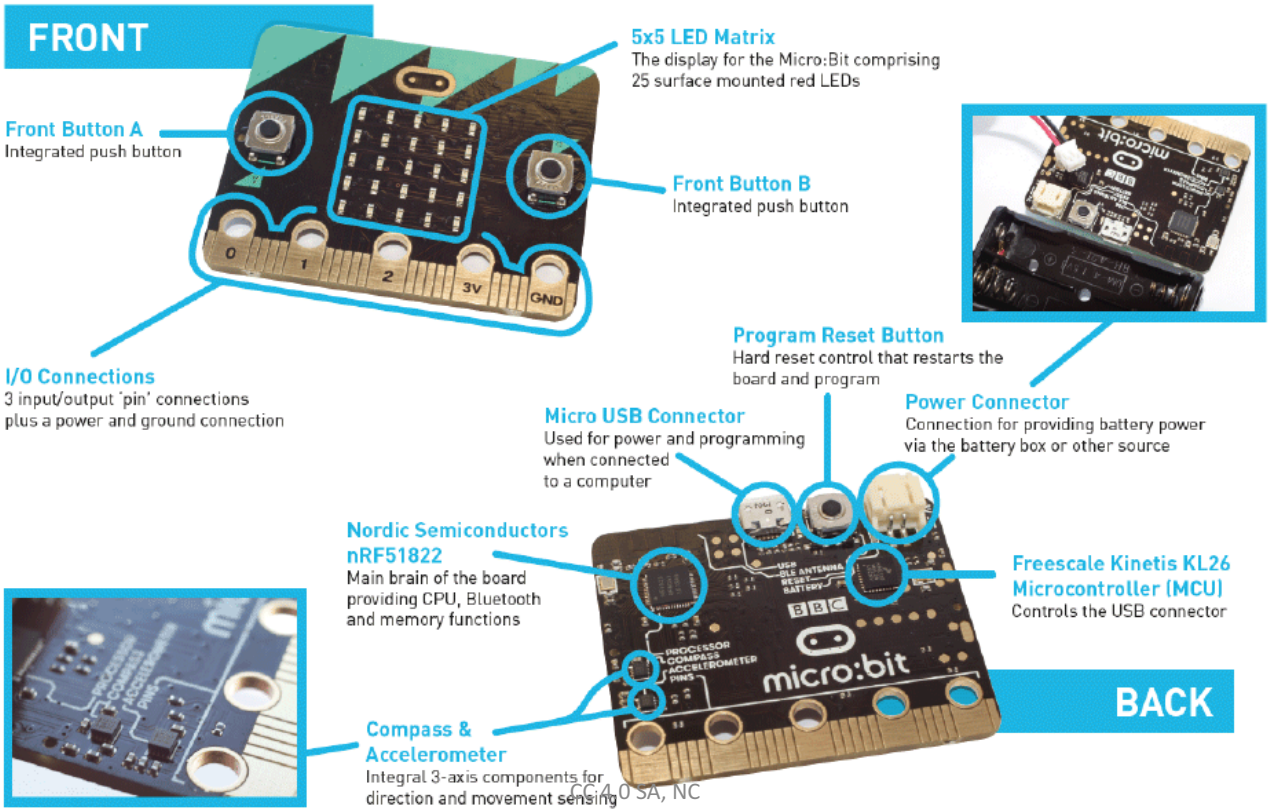
- In computing, a **visual programming language** (**VPL**) is any programming language that lets users create programs by manipulating program elements graphically rather than by specifying them textually. A VPL allows programming with visual expressions, spatial arrangements of text and graphic symbols, used either as elements of syntax or secondary notation.

- From [wikipedia](https://en.wikipedia.org/wiki/Visual_programming_language)

Different Types

- Computer Programming paradigms
  - OO, flow-based, …
- Programming stage
  - conceptual, modeling, algorithm, ...
- Application Domains
  - finance, embedded system, education, multimedia, …
- System Automation
  - process, UI, logic validation, …

micro:bit (1/2)



micro:bit (2/2)

micro:bit is an ARM-based embedded system designed by the BBC for use in computer education in the UK, released in February 10th 2016.

- The board is 4 cm × 5 cm and has
  - an ARM Cortex-M0 processor,
  - powered by either USB or an external battery pack.
- Input/Output, Interface
  - three ring connectors (plus one power one ground) which accept crocodile clips or 4 mm banana plugs
  - a 23-pin edge connector with two or three PWM outputs, six to 17 GPIO pins (depending on configuration), six analog inputs, serial I/O, SPI, and I^2C.
  - accelerometer and magnetometer sensors,
  - Bluetooth and USB connectivity,
  - a display consisting of 25 LEDs, two programmable buttons one reset button.
- Software
  - Programming Language: Javascript, micropython, C/C++, TouchDevelop, Free Pascal
  - Editor: [pxt](https://makecode.microbit.org/) based on Blockly for JavaScript, [mu](https://www.microbit.co.uk)

JavaScript Cheat Sheet

Cheatography

JavaScript Cheat Sheet
by Dave Child (DaveChild) via cheatography.com/1/cs/7/

Regular Expressions Syntax

^	Start of string
$	End of string
.	Any single character
(a|b)	a or b
(...)	Group section
[abc]	In range (a, b or c)
[^abc]	Not in range
\s	White space
a?	Zero or one of a
a*	Zero or more of a
a*?	Zero or more, ungreedy
a+	One or more of a
a+?	One or more, ungreedy
a{3}	Exactly 3 of a
a{3,}	3 or more of a
a{,6}	Up to 6 of a
a{3,6}	3 to 6 of a
a{3,6}?	3 to 6 of a, ungreedy
\	Escape character
[:punct:]	Any punctuation symbol
[:space:]	Any space character
[:blank:]	Space or tab

There's an excellent regular expression tester at: http://regexpal.com/

Pattern Modifiers

g	Global match
i *	Case-insensitive
m *	Multiple lines
s *	Treat string as single line

Pattern Modifiers (cont)

x *	Allow comments and whitespace in pattern
e *	Evaluate replacement
U *	Ungreedy pattern

* PCRE modifier

JavaScript RegExp Object

compile()	lastParen
exec()	leftCOntext
global	multiline
ignoreCase	rightContext
input	source
lastIndex	test()
lastMatch	

JavaScript Event Handlers

onabort	onmousedown
onblur	onmousemove
onchange	onmouseout
onclick	onmouseover
ondblclick	onmouseup
ondragdrop	onmove
onerror	onreset
onfocus	onresize
onkeydown	onselect
onkeypress	onsubmit
onkeyup	onunload
onload	

JavaScript Arrays

concat()	slice()
join()	sort()
length	splice()
pop()	toSource()
push()	toString()
reverse()	unshift()
shift()	valueOf()

JavaScript Numbers and Maths

abs()	min()
acos()	NEGATIVE_INFINITY
asin()	PI
atan()	POSITIVE_INFINITY
atan2()	pow()
ceil()	random()
cos()	round()
E	sin()
exp()	sqrt()
floor()	SQRT1_2
LN10	SQRT2
LN2	tan()
log()	toSource()
LOG10E	toExponential()
LOG2E	toFixed()
max()	toPrecision()
MAX_VALUE	toString()
MIN_VALUE	valueOf()
NaN	

Cheatography

JavaScript Cheat Sheet
by Dave Child (DaveChild) via cheatography.com/1/cs/7/

JavaScript Booleans

toSource()	valueOf()
toString()	

JavaScript Dates

Date()	setMonth()
getDate()	setFullYear()
getDay()	setHours()
getMonth	setMinutes()
getFullYear	setSeconds()
getYear	setMilliseconds()
getHours	setTime()
getMinutes	setUTCDate()
getSeconds	setUTCDay()
getMilliseconds	setUTCMonth()
getTime	setUTCFullYear()
getTimezoneOffset()	setUTCHours()
getUTCDate()	setUTCMinutes()
getUTCDay()	setUTCSeconds()
getUTCMonth()	setUTCMilliseconds()
getUTCFullYear()	toSource()
getUTCHours()	toString()
getUTCMinutes()	toGMTString()
getUTCSeconds()	toUTCString()
getUTCMilliseconds()	toLocaleString()
parse()	UTC()
setDate()	valueOf()

JavaScript Strings

charAt()	slice()
charCodeAt()	split() x
concat()	substr()
fromCharCode()	substring()
indexOf()	toLowerCase()
lastIndexOf()	toUpperCase()
length	toLocaleLowerCase()
localeCompare()	toLocaleUpperCase()
match() x	toSource()
replace() x	valueOf()
search() x	

String object methods with an x support regular expressions.

JavaScript Functions

decodeURI()	isNaN()
decodeURIComponent()	Number()
encodeURI()	parseFloat()
encodeURIComponent()	parseInt()
escape()	String()
eval()	unescape()
isFinite()	

By **Dave Child** (DaveChild)
cheatography.com/davechild/
www.getpostcookie.com

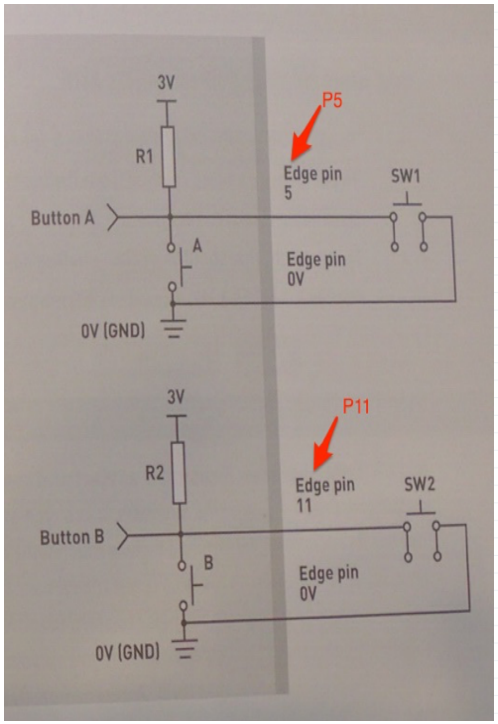
Published 19th October, 2011.
Last updated 11th May, 2016.
Page 2 of 2.

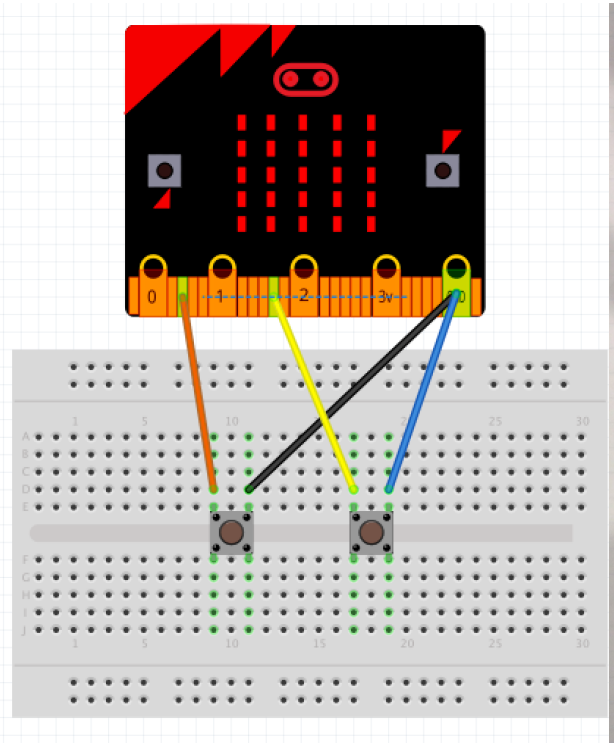
MicroPython Summary (supporting library)

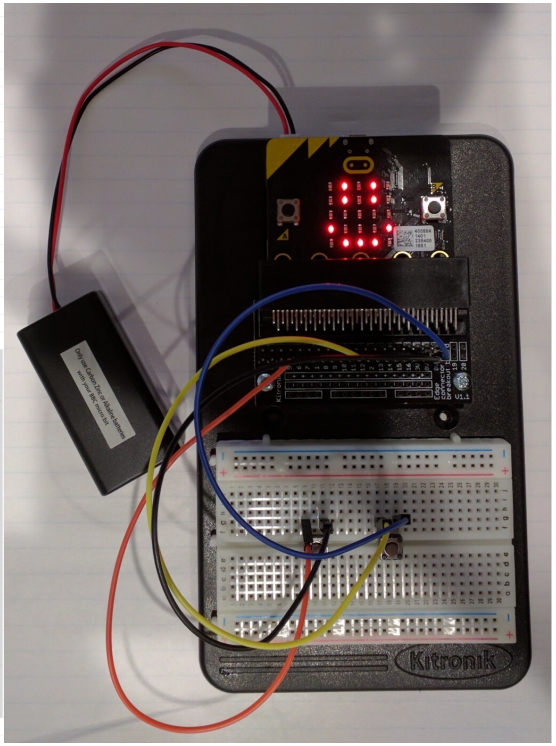
- array: numeric data arrays
- builtins: basic functions
- cmath: mathematical functions for complex numbers
- gc: garbage collector
- math: mathematical functions
- select: wait for events on a set of streams
- sys: system specific functions
- ubinascii: binary/ASCII conversions
- ucollections: collection and container types
- uhashlib: hashing algorithm
- uheapq: heap queue algorithm
- uio: input/output streams
- ujson: JSON encoding and decoding • uos: basic “operating system” services
- ure: regular expressions
- usocket: socket module
- ustruct: pack and unpack primitive data types
- utime: time related functions
- uzlib: zlib decompression
- uctypes: access binary data in a structured way

- [Micropython differences from cpython](https://docs.micropython.org/en/latest/pyboard/genrst/index.html)

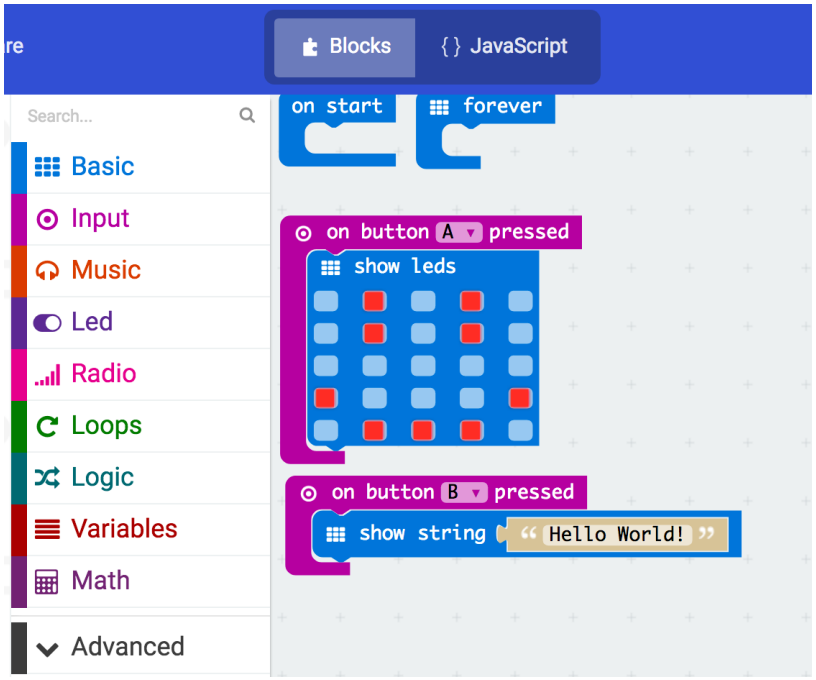
Schematic, Breadboard, and Wiring







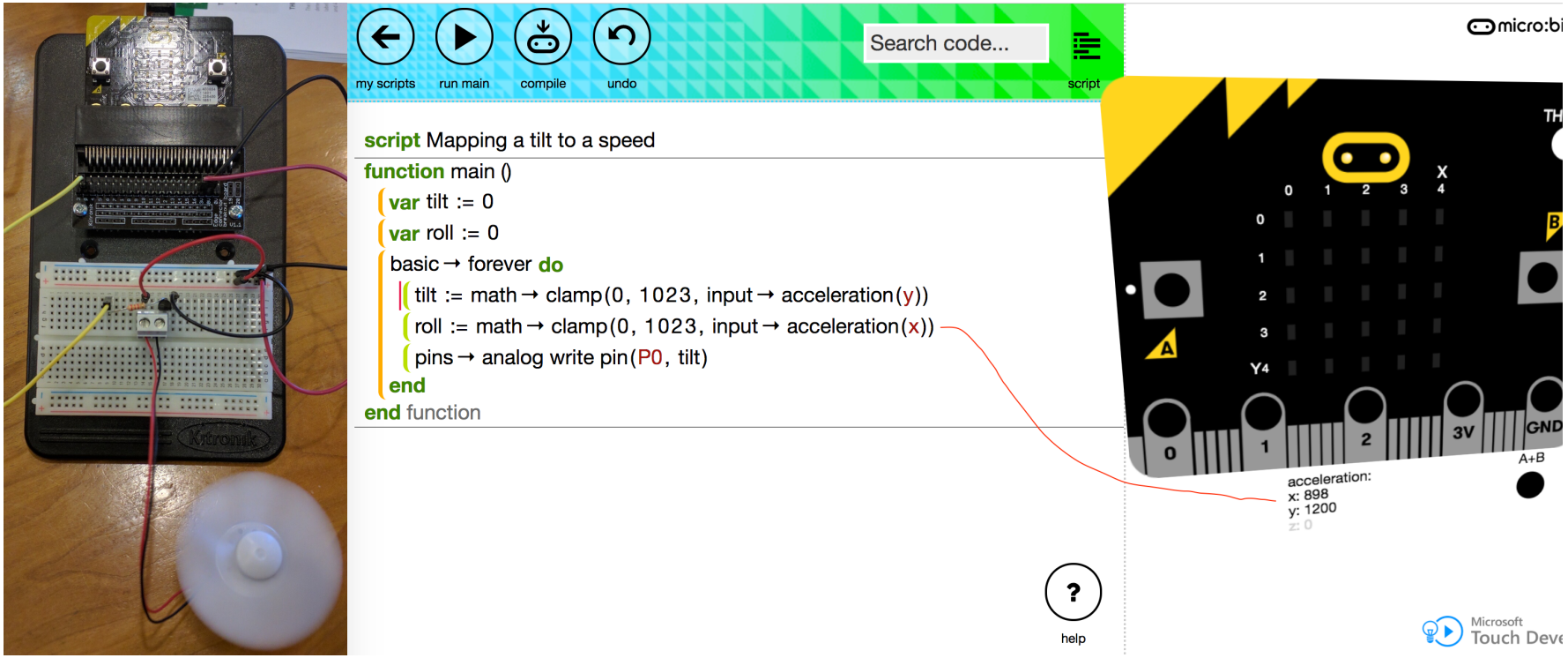
Block and Script in PXT Editor



Javascript

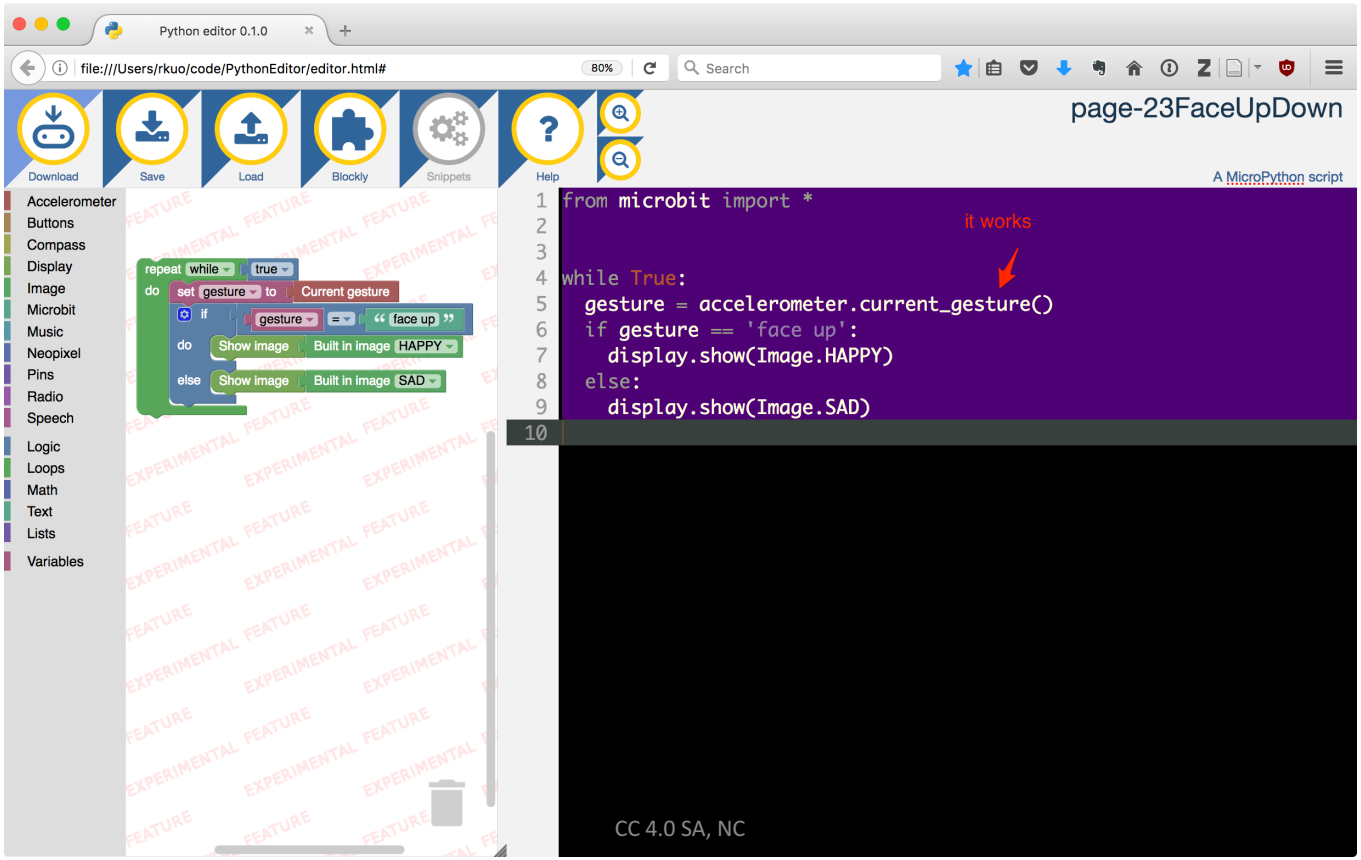
```
basic.forever(() => {

})
input.onButtonPressed(Button.A, () => {
    basic.showLeds(`
        . # . # .
        . # . # .
        . . . . .
        # . . . #
        . # # # .
        `)
})
input.onButtonPressed(Button.B, () => {
    basic.showString("Hello World!")
})
```

Microsoft Touch Develop Example
Using the Accelerometer to Control Motor Speed



MicroPython Example in Mu Editor

open file:///Users/rkuo/code/PythonEditor/editor.html



Review

Gain

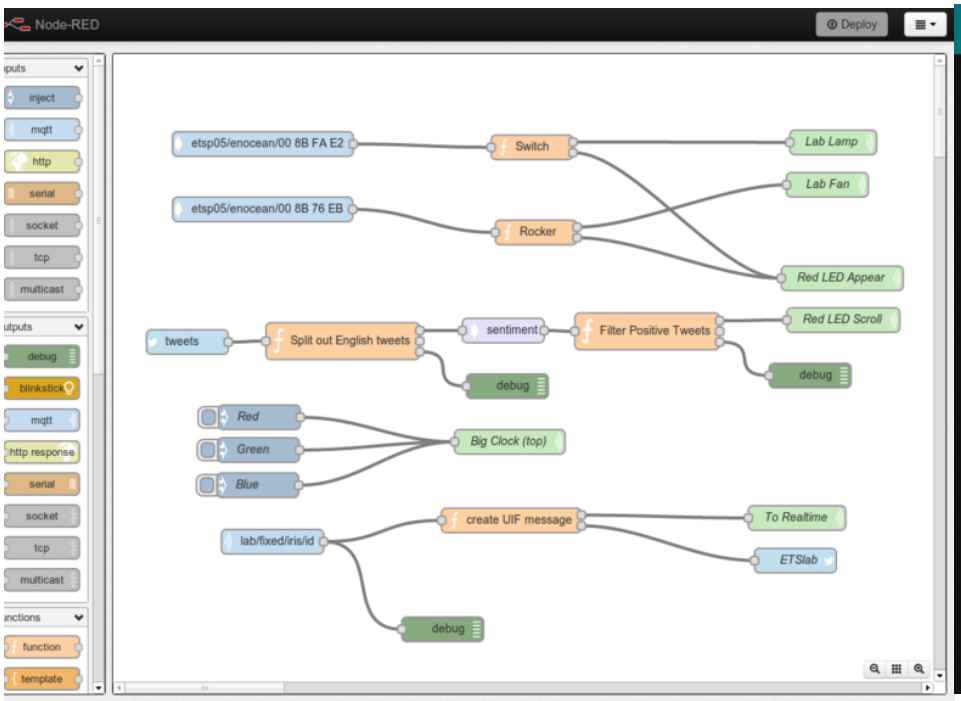
- Improve understanding of
  - Visual Programming Language,
  - JavaScript, Python.
- Gain some hands-on experience on
  - Microcontroller, sensor, wiring.
- Improve knowledge
  - Prototyping software.
- Discover ESP32 microcontroller

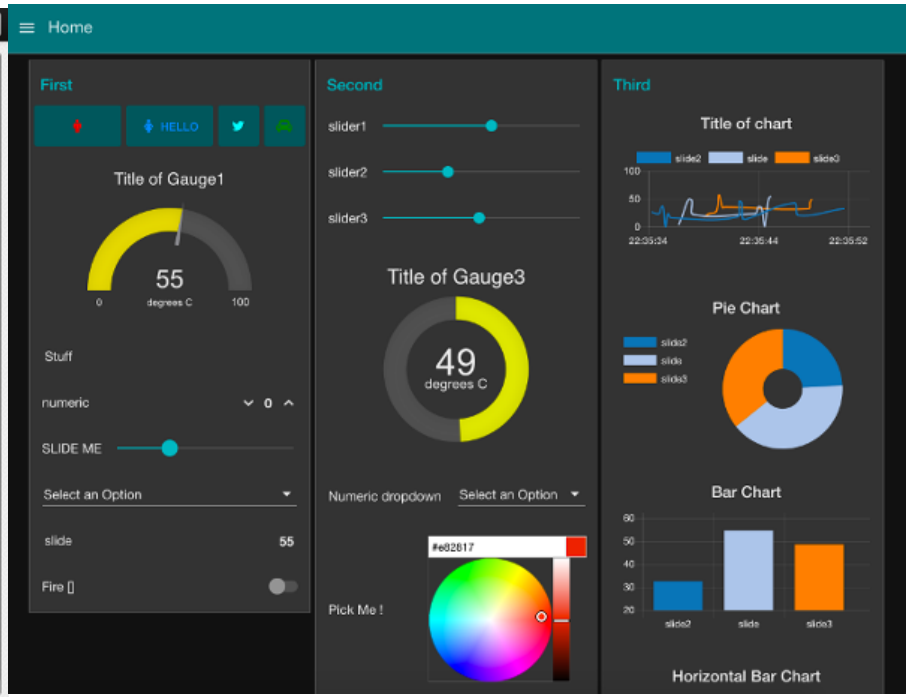
Plan

- Component design in Node-Red, Blockly, App-Inventor.
- Domain specific visual programming: data analytics.
- Automation control

Supplement Slides

Node-Red

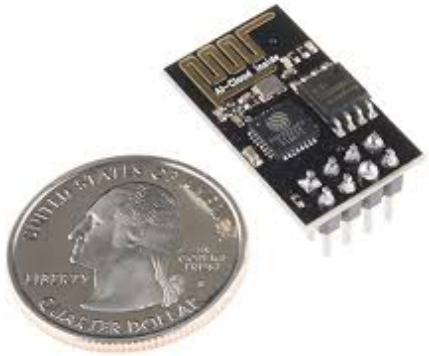




AppInventor

ESP8266 (https://en.wikipedia.org/wiki/ESP8266)

- 32-bit RISC CPU: Tensilica Xtensa L106 running at 80 MHz*
- 64 KiB of instruction RAM, 96 KiB of data RAM
- External QSPI flash: 512 KiB to 4 MiB* (up to 16 MiB is supported)
- IEEE 802.11 b/g/n Wi-Fi
  - Integrated TR switch, balun, LNA, power amplifier and matching network
  - WEP or WPA/WPA2 authentication, or open networks
- 16 GPIO pins
- SPI
- I^2C
- I^2S interfaces with DMA (sharing pins with GPIO)
- UART on dedicated pins, plus a transmit-only UART can be enabled on GPIO2
- 10-bit ADC (this is a Successive Approximation ADC)



- The successor to these microcontroller chips is the ESP32.

	Espresif ESP32	Espresif ESP8266	Ameba RTL8195	Ameba RTL8710
RAM	520 KB	36 KB available to user		48 KB available to user
CPU	Tensilica Xtensa LX6 32 bit Dual-Core @ 160 / 240Mhz	Tensilica LX106 32 bit @ 80 MHz (up to 160 MHz)	ARM Cortex M3 32 bit @ 166 MHz	ARM Cortex M3 32 bit @ 83 MHz
Flash	up to 64 MBytes	1, 2, 4, 8 or 16 MB		1MB Built-in
Input Voltage	2.2 V - 3.6 V	3.0 V - 3.6 V	3.0 V - 3.6 V	3.0 V - 3.6 V
Operating Current	80 mA average	80 mA average	80 mA average	80 mA average
Temperature Range	-40°C to 125°C	-40°C to 125°C	-40°C to +125°C	-40°C to +125°C
Standards	FCC/CE/TELEC/KCC	FCC/CE/TELEC/SRRC	FCC/CE/TELEC/SRRC	FCC/CE/TELEC/SRRC
Connectivity				
Wi-Fi	✓ 802.11b/g/n	✓ 802.11b/g/n (max 65Mbps)	✓ 802.11b/g/n	✓ 802.11b/g/n
Bluetooth®	✓ 4.2 BR/EDR + BLE	✗	✗	✗
UART	3x UART	2x UART	3x UART	2x UART
I/O				
I2C	✓ up to 2	✓ up to 1	✓ up to 4	✓ up to 1
GPIO	32 (up to)	17 (up to)	30 (up to)	17 (up to)
PCM	✓ up to 1	✗	✓ up to 2	✗
PWM	✓ up to 8	✗	✓ up to 4	✓ up to 4
ADC	✓ 12 bit SAR ADC up to 18 channels	✓ 10 bit	✓ 2 channels	✗
DAC	✓ 8 bit up to 2 channels	✗	✓ 1 channel	✗
Dimensions				
Package	QFN-48	QFN-32	QFN-48	QFN-48
Height	0.23622 in (6 mm)	0.19685 in (5 mm)	0.23622 in (6 mm)	0.23622 in (6 mm)
Width	0.23622 in (6 mm)	0.19685 in (5 mm)	0.23622 in (6 mm)	0.23622 in (6 mm)
Links				
Website	espressif.com/...	espressif.com/...	amebaiot.com/en/	amebaiot.com/en/
Wikipedia	wikipedia.org/...	wikipedia.org/...		
See on Amazon		amazon.com/...	amazon.com/...	