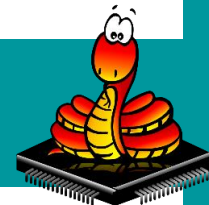# Introduction to Coding

Content

- What is Python and Micro Python

- Basic Operations

    - Variable and Data (Integer, Float, String)

    - Data Array (List, Dictionary)

- Process Flows

    - Functional Programming

    - Loop

    - Statements

Author: Andi Dinata
December 2018

MicroPython

Why coding

- What is it for me?
  - Coding is basic literacy and important in digital world today and prepare for future
  - Understand with technology around you
 - Helps with communication, creativity, math, writing and confidence

MicroPython
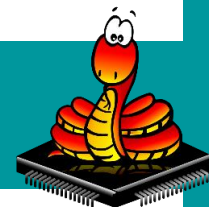
What is python

- Interpreted, interactive language
- Invented by Guido van Rossum
- 2018 most popular language

Language Rank	Types	Spectrum Ranking
1. Python	🌐 🖥🔲	100.0
2. C++	📱🖥🔲	99.7
3. Java	🌐📱🖥	97.5
4. C	📱🖥🔲	96.7
5. C#	🌐📱🖥	89.4
6. PHP	🌐	84.9
7. R	🖥	82.9
8. JavaScript	🌐📱	82.6
9. Go	🌐🖥	76.4
10. Assembly	🔲	74.1

YouTube
DropBox
Google
Quora
Instagram
BitTorrent
Spotify
Reddit
YahooMaps
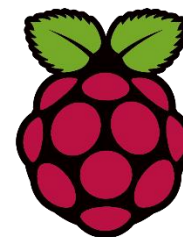
Raspberry Pi

- Web Development, Desktop
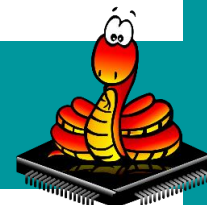
MicroPython

What is Micro Python

- Implementation of Python in Microcontroller

- Invented by Damien George

PC, Laptop, Server
Large resource
- Processor (GHz)
- Memory (Gb)

Microcontroller
Constraint resource
- Processor MHz (160 MHz)
- Memory 64 Kb

- Same language structure. Complete rewrite of Python 3 version

- Learning Micro Python = Python
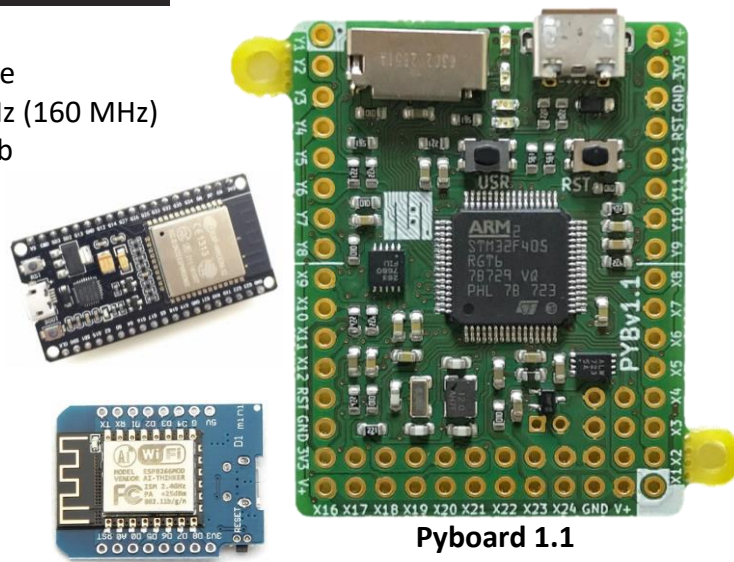
Pyboard 1.1

Why Micro Python

Internet of Things
(IoT)

Automation

Robotics

Smart Devices

Digital Education

Games

Analytic

Digital Agriculture

Machine Vision
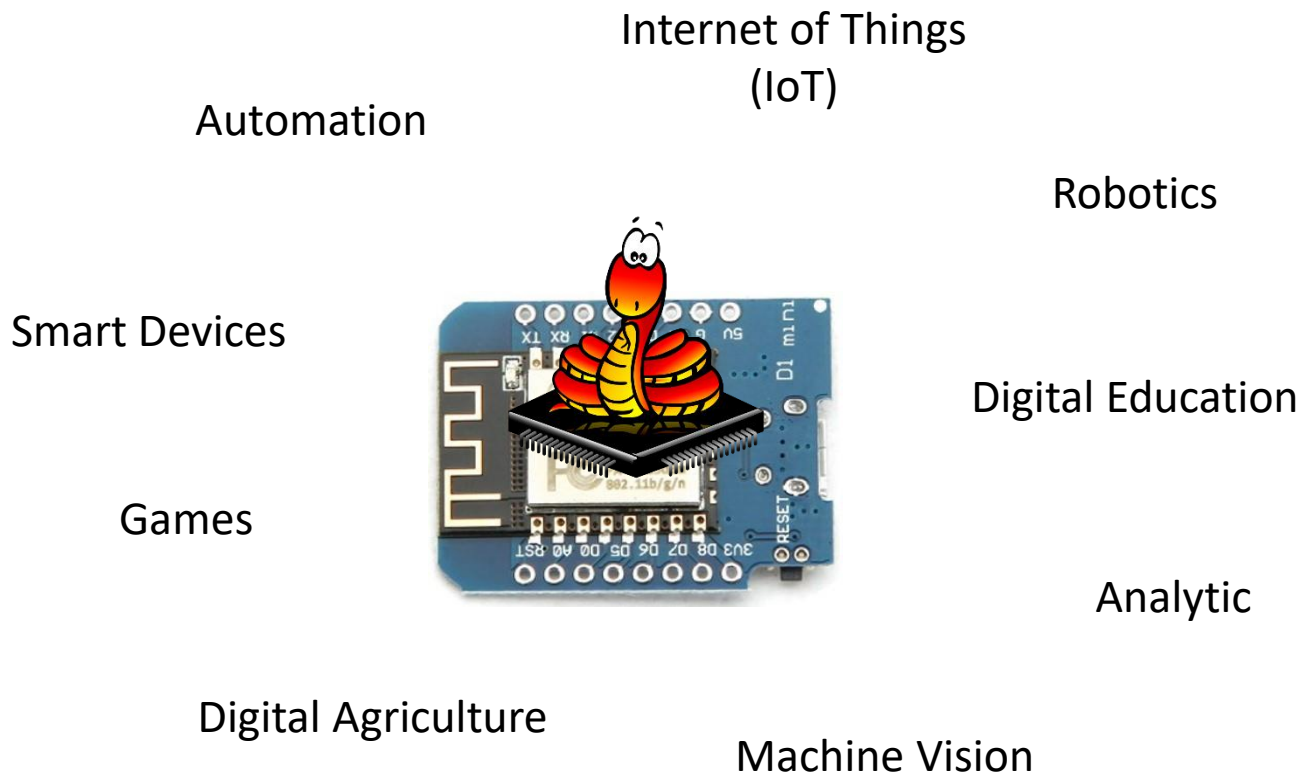
MicroPython

WHOAMI

- Andi Dinata

- Multi-national company for > 10 years

- Coding self-learning since high school

- Pycon 2017 speaker

- Book author now in store. The 2nd book is coming.

- Teach coding class from age 8

- Workshops

- Find out more:
  - andidinata.com
  - @mdinata
  - @codeclubjr

Introduction to Coding

3

Next

- Physical Computing

- Go futher

2
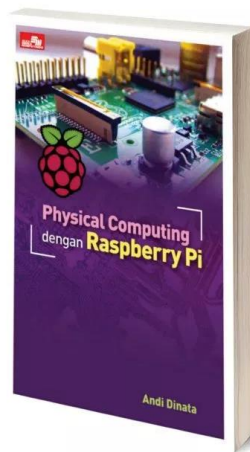
Process Flow

- Functional Programming

- Conditional

- Loop

1

Basic Operations

- Variable and Data
  (Integer, Float, String)

- List, Dictionary
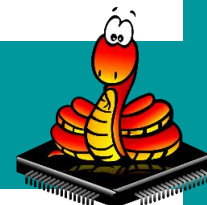
MicroPython

Micro Python prompts

>>>

. . .

===

Do not need to write it

MicroPython
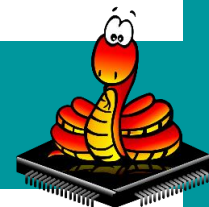
Understanding Variable and Data

Variable is a assignment of a value

```
>>> var = 5
```

Value 5 is represented with a variable called var. To read the value represented by variable, type in the variable name

```
>>> var
```

Create more variables and assign value

```
>>> A = 8
>>> B = 9.5
>>> C = "hello world"
```

Variable A value is 8. This type of value is called integer (=whole number)

Variable B value is 9.0. This type of value is called float or decimal number. Because the decimal place can be anywhere

Variable C value is text. This type of value is called string. The string value is located between single quotes '' or double quotes sign ""

MicroPython
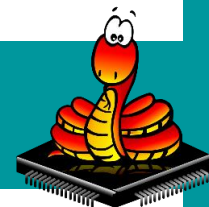
Variables and Data

Let's operate the variables

```
>>> A + B
17.5
```

Create a new variable to assign value of A+B and press ENTER

```
>>> C = A + B
```

Nothing is printed on the screen. To display the value of variable C we can type variable name

```
>>> C
17.5
```

Or use `print()` command and put the variable or value in the parenthesis

```
>>> print(C)
17.5
```

`print()` is very useful command to display the output from the operation on the screen so we could check if the result is OK. And very useful for debugging purposes.

MicroPython
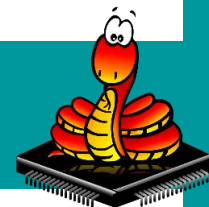
Variables and Data

Multiplication   : *

Addition   : +

Subtraction   : -

Division (float)   : /

Division (integer)   : //

Remainder   : %

```
>>> 3/2
>>> 3//2
>>> 6//3
>>> 6%5
```

Numbers in parenthesis () will be operated first

```
>>> 5*(10+10)
>>> 5*10+10
```

MicroPython
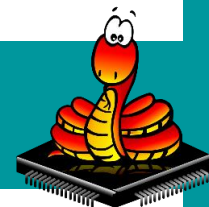
Variables and Data

To find the values type, use type() command and put the value or variables in the parenthesis

```
>>> type(A)
```

Values with type integer, float and string can be converted to any type.

To convert, use `int()` to convert to integer, `float()` to convert to a float/decimal value and, `str()` to convert to text

```
>>> A=8
>>> str(A)
'8'
```

When a value is converted to string, it returns value with ' ' sign.

```
>>> egg = 20
>>> ham = '30'
>>> breakfast = egg + ham
```

This will return `TypeError` because `int` and `str` cannot be operated

```
>>> breakfast = egg + int(ham)
>>> print(breakfast)
```

No error generated because variable `ham` is converted to integer

MicroPython
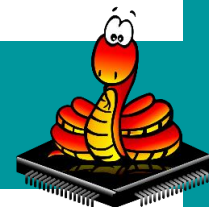
Variables and Data

String addition will result concatenation

```
>>> text1 = "my birthday is"
>>> text2 = "25 December 2009"
>>> message = text1+text2
>>> print(message)
'my birthday is25 December 2009'
```

How to combine text and numbers ?

```
>>> age= 9
>>> msg = "My age is %d years old" %age
>>> print(msg)
```

What ever comes after the % sign will be replaced by value of variable called var. Type in the variable name to see the result.

d is the representation of integer value. And f is float

```
>>> age= 10.5
>>> msg = "My age is %d years old" %age
>>> print(msg)
```

The result will be 10 instead of 10.5 because %d also converts the float automatically to integer
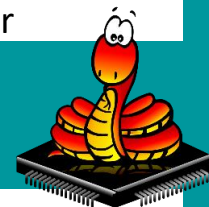
Variables

```
>>> num= 1.5
>>> msg = "He ate %f portion of noodles at home" %num
>>> print(msg)
'He ate 1.500000 portion of noodles at home'
```

There are too many zeros. To cut it to decimal places we need to add the decimal places between % and f

```
>>> msg = "He ate %.1f portion of noodles at home" %num
'He ate 1.5 portion of noodles at home'
```

MicroPython
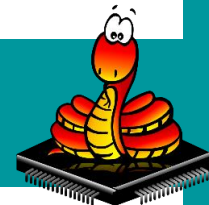
Understanding List

Like school bag, list is used to store integer, float or string. We can add, remove, organize items put in the list

Let's make a list called bag. List is using `[]` sign

```
>>> bag =[]
```

That's an empty list. To add item in the bag we can use append

```
>>> bag.append('pencil')
>>> bag.append('ruler')
>>> bag.append('eraser')
>>> bag.append('cap')
>>> bag.append('book')
>>> bag.append('ballpoint')
>>> print(bag)
['pencil', 'ruler', 'eraser', 'cap', 'book', 'ballpoint']
```

Item in the list can be long or short. To calculate the length of a list use len

```
>>> len(bag)
```

Item inside the list has order. Micro Python counts from 0, not 1. Thus, pencil is placed at number 0, ruler at 1, and so on.

MicroPython
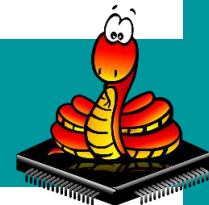
Understanding List

To call the item in the list, just put the order number in [] sign

```
>>> bag[0]
```

```
>>> bag[3]
```

Another way to add item in the list is using insert. With insert, we have to specify at which order we are going to place the item in the list

```
>>> bag.insert(1,'book')
```

```
>>> print(bag)
```

To remove item from list, use pop(). If the order number is specified, then item at order 2 will be removed. If not specified, the last item in the list will be removed

```
>>> bag.pop(2)
```

```
>>> bag.pop()
```

MicroPython
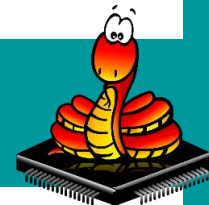
Understanding Dictionary

Dictionary is just like list, but dictionary contains pair of key and value. Think about phone book, the key is name of our friend, value is their phone number.

Dictionary is written in {}

Let's create empty dictionary called phonebook

```
>>> phonebook={}
```

To start adding pair of key and value

```
>>> phonebook['mom']=0218997893
>>> phonebook['friend']=0217783312
>>> phonebook['uncle']=0218890033
>>> print(phonebook)
```

To delete use del

```
>>> del phonebook['uncle']
>>> print(phonebook)
```
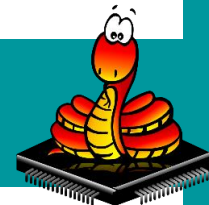
Functional Programming

Function is a way to combine many actions in a single process. Use keyword def, means define, then followed by the function name and followed by colon (:) sign.

Here is example.

```
>>>def toast():
        print('warm up pan')
        print('get the bread')
        print('toast both sides')
        print('cool down')
        print('enjoy')
```

Pay attention with the space. Use tab key to indent. To run a function just call the function name followed by ()

```
>>> toast()
```

Function is powerful, we only need to write it once but reuse it for many times

MicroPython
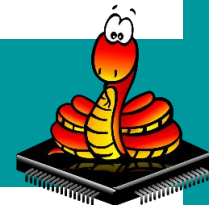
Functional Programming

We can pass the argument in a function. Let's make a function to add value to the list of fruits. First let's make empty list of fruit

```
>>> fruit=[]
>>> def add(a):
        fruit.append(a)
        print(fruit)
        message="%d of fruits added" %len(fruit)
        print(message)
```

Try to run the function without argument.

```
>>> add()
```

It will return error because the function is defined with one argument. Now try to run the function with argument

```
>>> add('banana')
>>> add('apple')
>>> add('mango')
```

Conditional

A way to <u>check</u> for something using IF ELSE statement

```
if condition is true:          if condition is true:
    do this                        do this
else:                          elif another condition is true:
  do that                          do this one
                               else:
                                   do that
```
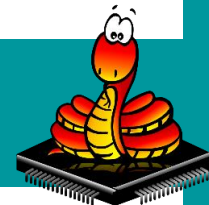
Let's make a function to check conditional statement

```
>>> def myresult(x):           >>> def myresult(x):
        if x >= 70:                    if x >= 80:
            print("PASSED")                print("PASSED")
        else:                          elif 70>= x >=60:
            print("NOT PASSED")            print("REPEAT")
                                       else:
                                           print("NOT PASSED")
```

MicroPython
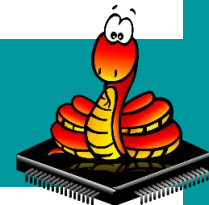
while loop

Loop is way to repeat actions over and over and over <u>with less code</u>

Two kinds of loops, indefinite and definite

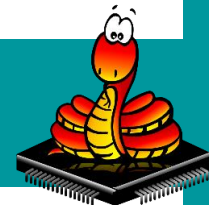Definite loop will repeat action until a condition is reached, then the action stops. Let's make one function to count number with limit

```
>>> def counter():
        count=0
        while count < 10:
            count += 1
            print(count)
```

To run the function, call the function name and see what happens

```
>>> counter()
```

The loop will break/stop when the condition is met.

MicroPython

while loop

Infinite loop will repeat actions FOREVER

Let's make one function to count number
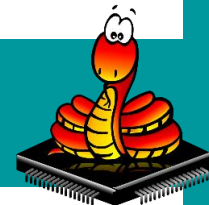
```
>>> def counter2():
        count=0
        while True:
            count += 1
            print(count)
```

To run the function, call the function name and see what happens

```
>>> counter2()
```

While the condition is True, then the addition will continue, no stop. To break the loop, press Ctrl-C

while True is very important to keep the program running.

for loop

For loop is used to action each item in <u>list</u> or <u>range of number.</u> Let's make a list of breakfast menu

```
>>> number=[1,2,3,4,5,6,7,8,9,10]
```

Then let's display each individual item in the list
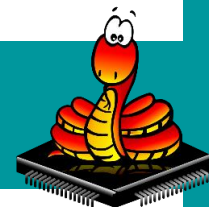
```
>>> for i in number:
        print(i)
```

What if we display only even number

```
>>> for i in number:
        if i%2 == 0:
            print(i)
```

Then display the odd number

```
>>> for i in number:
        if i%2 == 1:
            print(i)
```
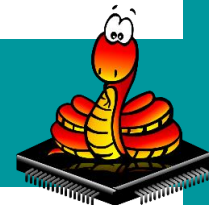
for loop

Let's work with range of number

```
>>> for i in range(30):
        print(i)
```

Remember that Micro Python starts the count from 0. You will find the last number is 29 instead of 30.

We could provide further customization by specifying start, end and step. In this example we want the count start from 0, ends at 30 and execute every 5 times

```
>>> for i in range(0,30,5):
        print(i)
```

Physical Computing

Computer that interacts with our physical world e.g. lights, sound/vibration, temperature, pressure through digital and analog pin

A demo that combines Physical Computing and IoT application is provided on your microcontroller board

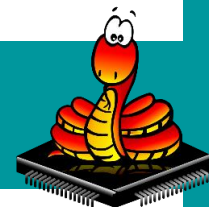First find your microcontroller name

```
>>> wifi.apname()
```

Then connect your laptop or smartphone to the access point name. Once connected type to activate the webserver

```
>>> import webserver
```

Open your internet browser on the phone and type address 192.168.4.1

Try to interact with on-board led from internet

MicroPython

end of slide
go to
github.com/mdinata/reference/introduction_to_coding
to download the slides

MicroPython