Expert Course

2024.08.

Daesung Kim

INDEX

0 Course Overview

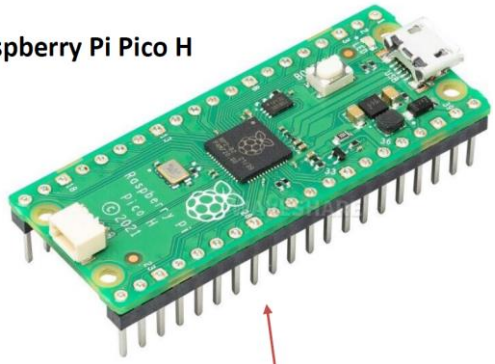
0 Course Overview

Based on Raspberry Pi Pico board

- Raspberry Pi Pico is a low-cost, high-performance micro-controller.
- It adopt the RP2040 chip developed by Raspberry Pi.
- It is possible to program with C or python language.

Raspberry Pi Pico (original)

Raspberry Pi Pico H



Pre-soldered header pins included

Raspberry Pi Pico W

WiFi Module and Antenna

IOT Expert Course

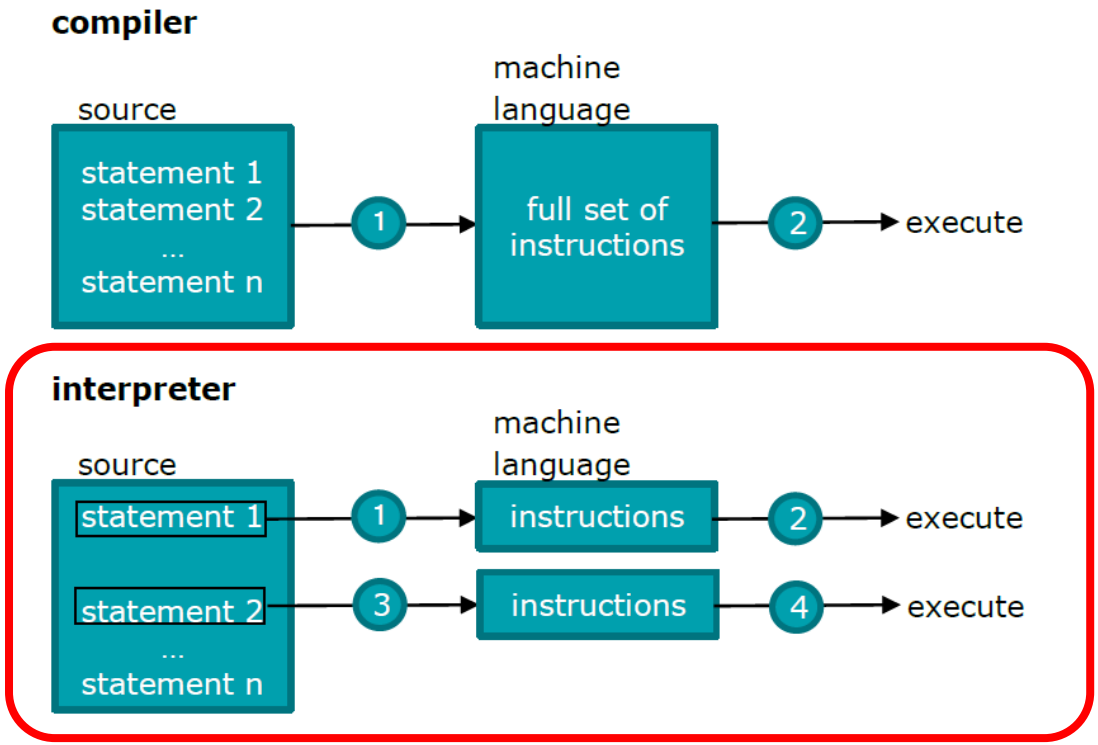
0 Course Overview

- Introduction

0 Course Overview

Compiler vs. interpreter

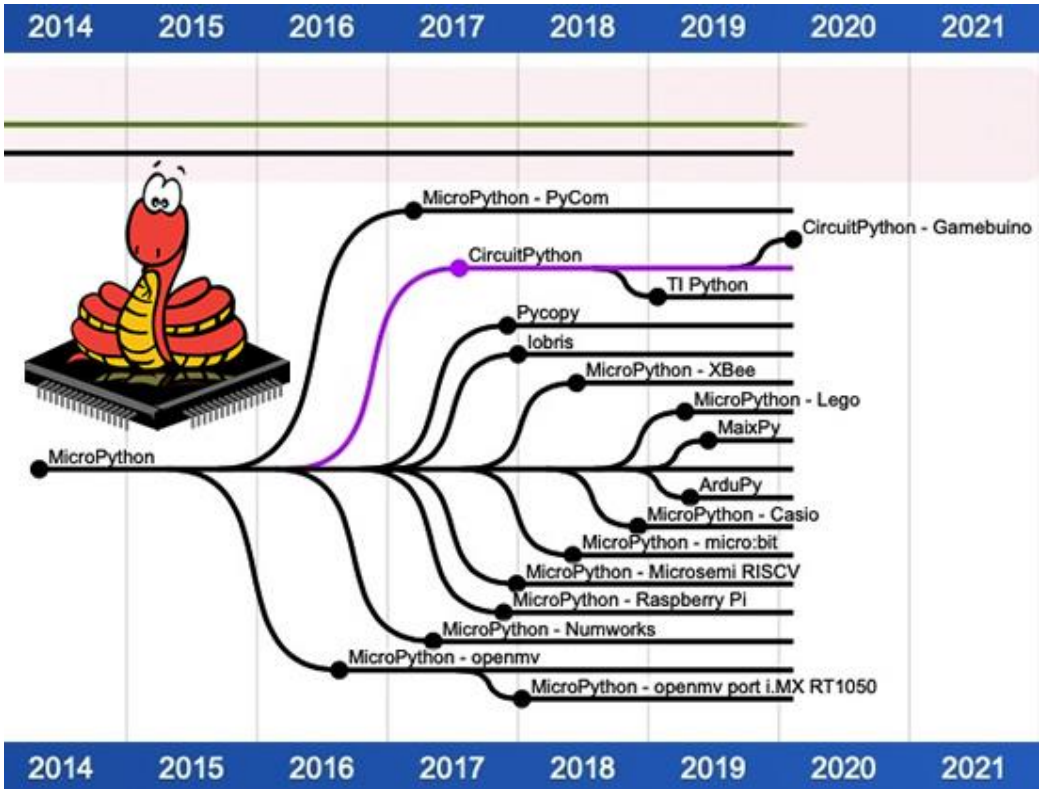
- A program is incomprehensible to a computer until translated into machine language. This translation is accomplished with a compiler or an interpreter.
  - ✓ A compiler translates the complete source program at once.
  - ✓ An interpreter translates one instruction at the time and executes that instruction directly.



3 Course Overview

Python Vs. MicroPython

- Python is a general-purpose programming language.
- MicroPython is optimized to run on devices, focusing on reducing size and resource usage.
- MicroPython provides libraries and modules necessary for interacting with microcontrollers.



1 Internet of Things(IoT)

1 Internet of Things(IoT)

Definition

- Internet of Things is a system of interrelated computing devices or objects which can transfer the data over a network without requiring any human to human or human to computer interaction uniquely addressable based on standard communication protocol.

  - When we speak about the “Things” in IoT, these are objects not precisely identifiable.
  - The sensors are used in the devices and objects and these feed the data to various IoT platforms.
  - Further, IoT platforms are used to gather the pinpointed information, detect patterns.
  - Thus, with the above process the IoT helps the organizations and institutions in reducing the cost through improved processes efficiency, asset utilization and productivity.

SRC : IBM, What is the Internet of Things?

1 Internet of Things(IoT)

Reasons of IoT

Data deluge

- The explosion of the amount of data collected and exchanged is one of the major reason why IoT came in existence. So, we need novel mechanisms to find, fetch, and transmit data.

Energy Issue

- There is decrease in energy required to operate intelligent devices. The search will be for a zero level of entropy where the device or system will have to harvest its own energy.

Miniaturization

- The devices are becoming increasingly smaller.

Autonomic management

- The devices/systems of future will have self-management, self-healing, and self-configuration capabilities.

1 Internet of Things(IoT)

IoT Enablers

- Energy
- Intelligence
- Communication
- Integration
- Interoperability
- Standards

1 Internet of Things(IoT)

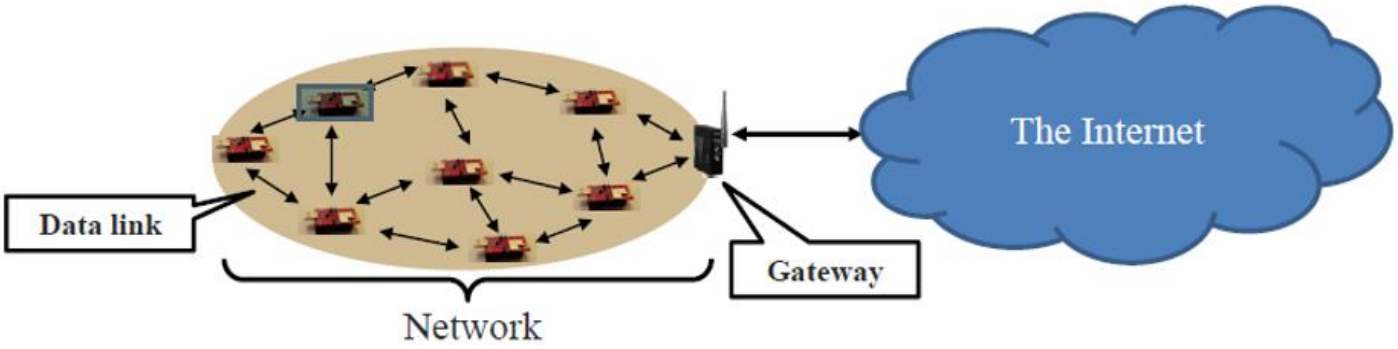
Things

- We can turn almost every object into a “thing”.
- A “thing” still looks much like an embedded system currently.
- A “thing” generally consists of four main parts:
  - ✓ Sensors & actuators
  - ✓ Microcontroller
  - ✓ Communication unit
  - ✓ Power supply
- A “thing” has the following properties:
  - ✓ It’s usually powered by battery. This implies limited source of energy.
  - ✓ It’s generally small size and low in cost. This limits their computing capability.
  - ✓ It doesn’t usually perform complicated tasks.
- Power consumptions is the main design issue.

1 Internet of Things(IoT)

Communications

- Popular RF-based communication solutions:
  - IEEE 802.15.4
  - IEEE 802.11 (or Wi-Fi)
  - Bluetooth
  - NFC(Near Field Communication) e.g. RFID



Internet of Things(IoT)

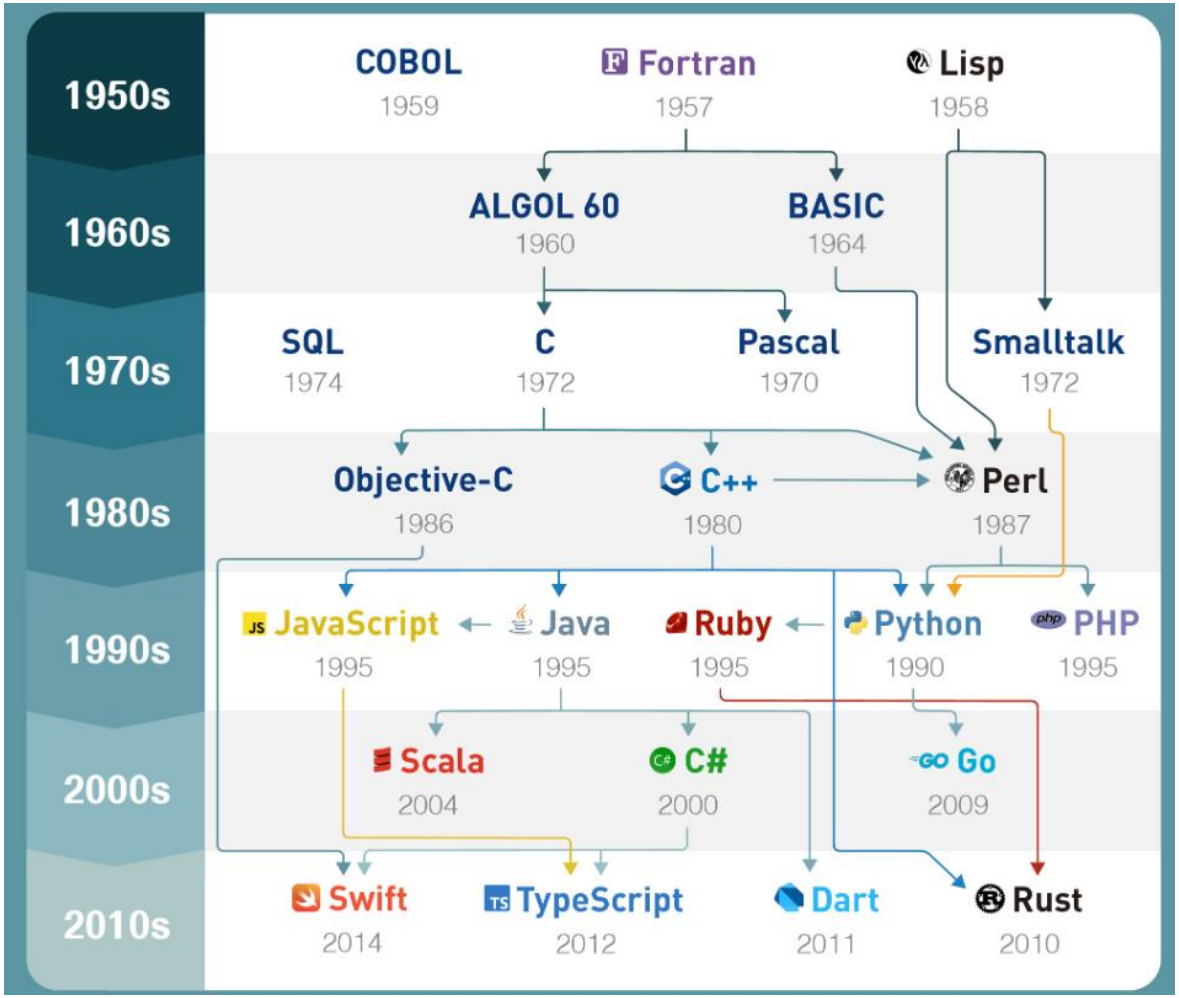
IoT Protocol Stacks

	IoT Stack	Web Stack
TCP/IP Model	IoT Applications and Device Management	Web Applications
Data Format	Binary, JSON, CBOR	HTML, XML, JSON
Application Layer	CoAP, MQTT, XMPP, AMQP	HTTP, DHCP, DNS, TLS/SSL
Transport Layer	UDP, DTLS	TCP, UCP
Internet Layer	IPv6/IP Routing and 6LoWPAN	IPv6, IPv4, IPSec
Network Layer	IEEE 802.15.4 MAC IEEE 802.15.4 PHY Physical Radio	Ethernet(IEEE 802.3), DSL, ISDN, Wireless LAN(IEEE 802.11), Wi-Fi

2 Python Programming

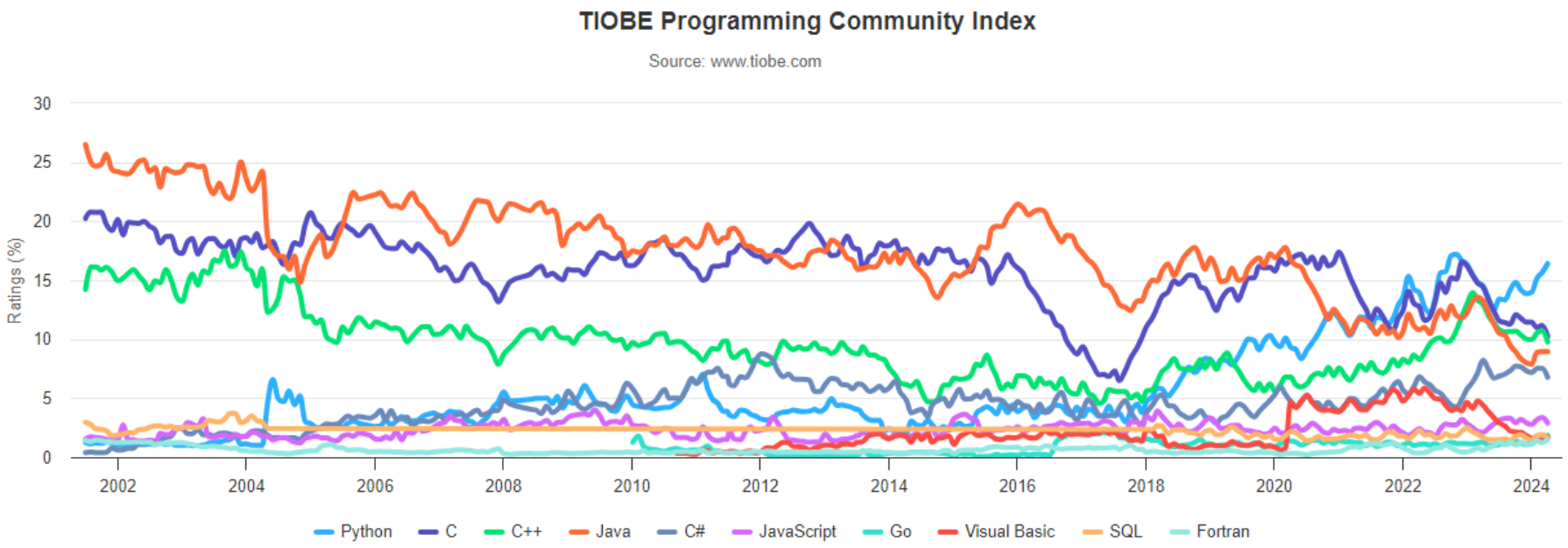
2 Python Programming

- History of programming language



2 Python Programming

- History of programming language

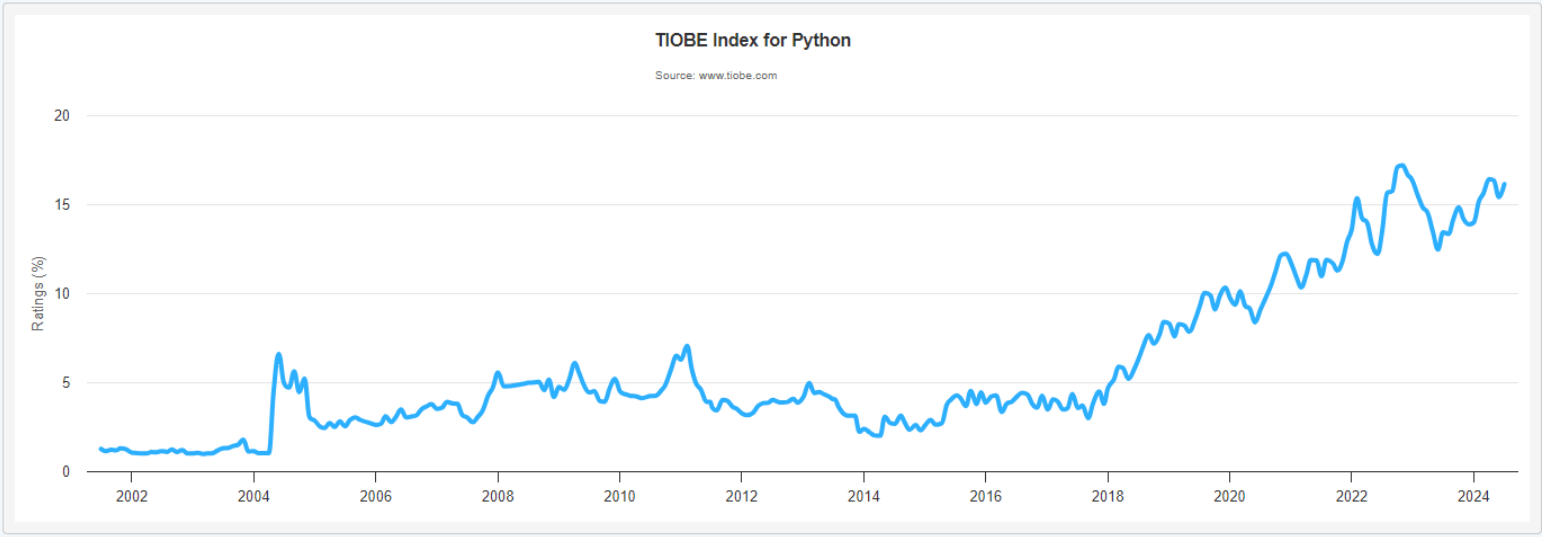


SRC : https://www.tiobe.com/tiobe-index/

2 Python Programming

History

- Invented in the Netherlands, early 90s by Guido van Rossum
- Named after Monty Python
- Open sourced from the beginning
- Considered a scripting language, but is much more scalable, object oriented and functional from the beginning
- Used by Google from the beginning
- Increasingly popular

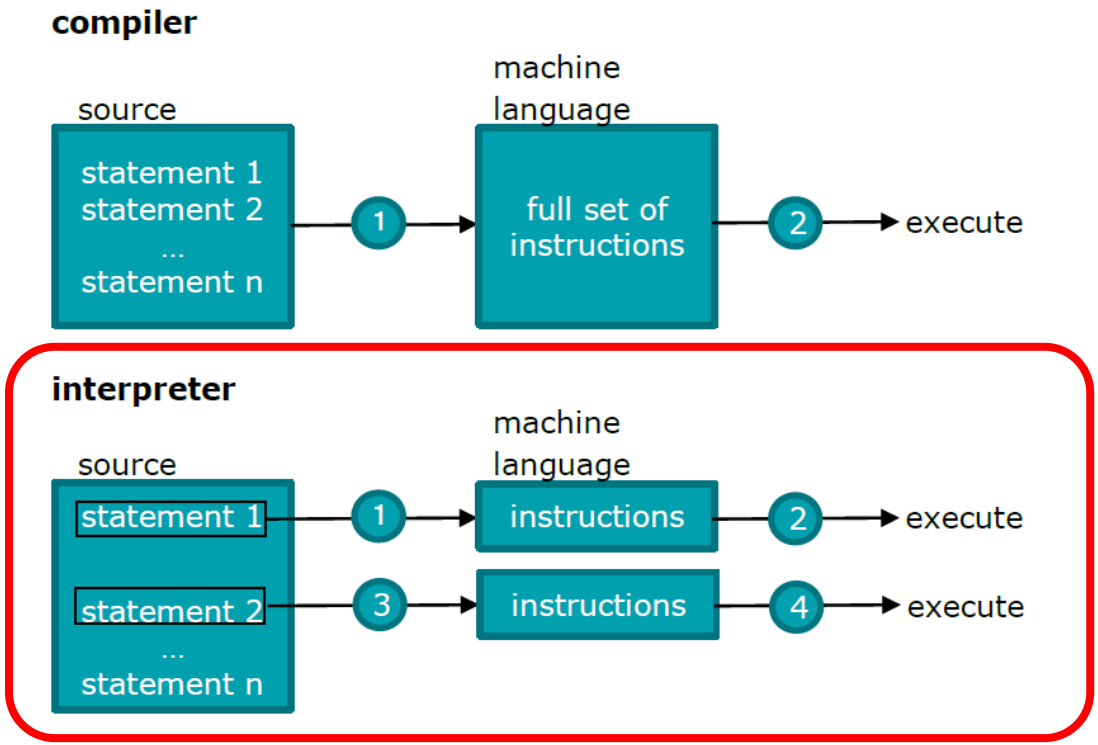


IOT Expert Course

2 Python Programming

Compiler vs. interpreter

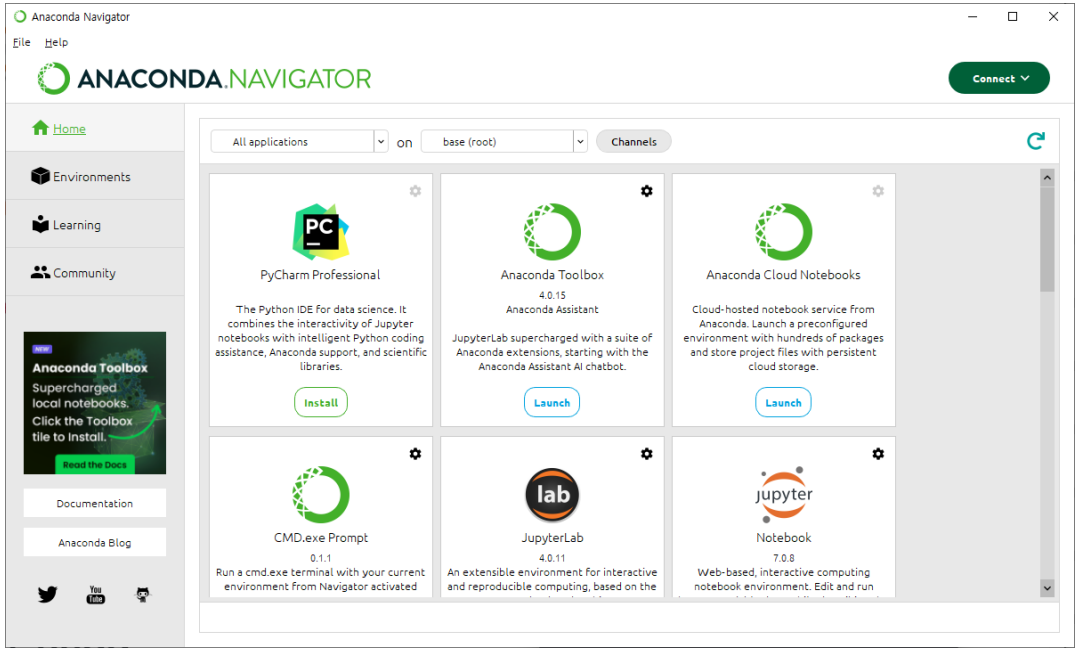
- A program is incomprehensible to a computer until translated into machine language. This translation is accomplished with a compiler or an interpreter.
  - ✓ A compiler translates the complete source program at once.
  - ✓ An interpreter translates one instruction at the time and executes that instruction directly.



2 Python Programming

Python with ANACONDA

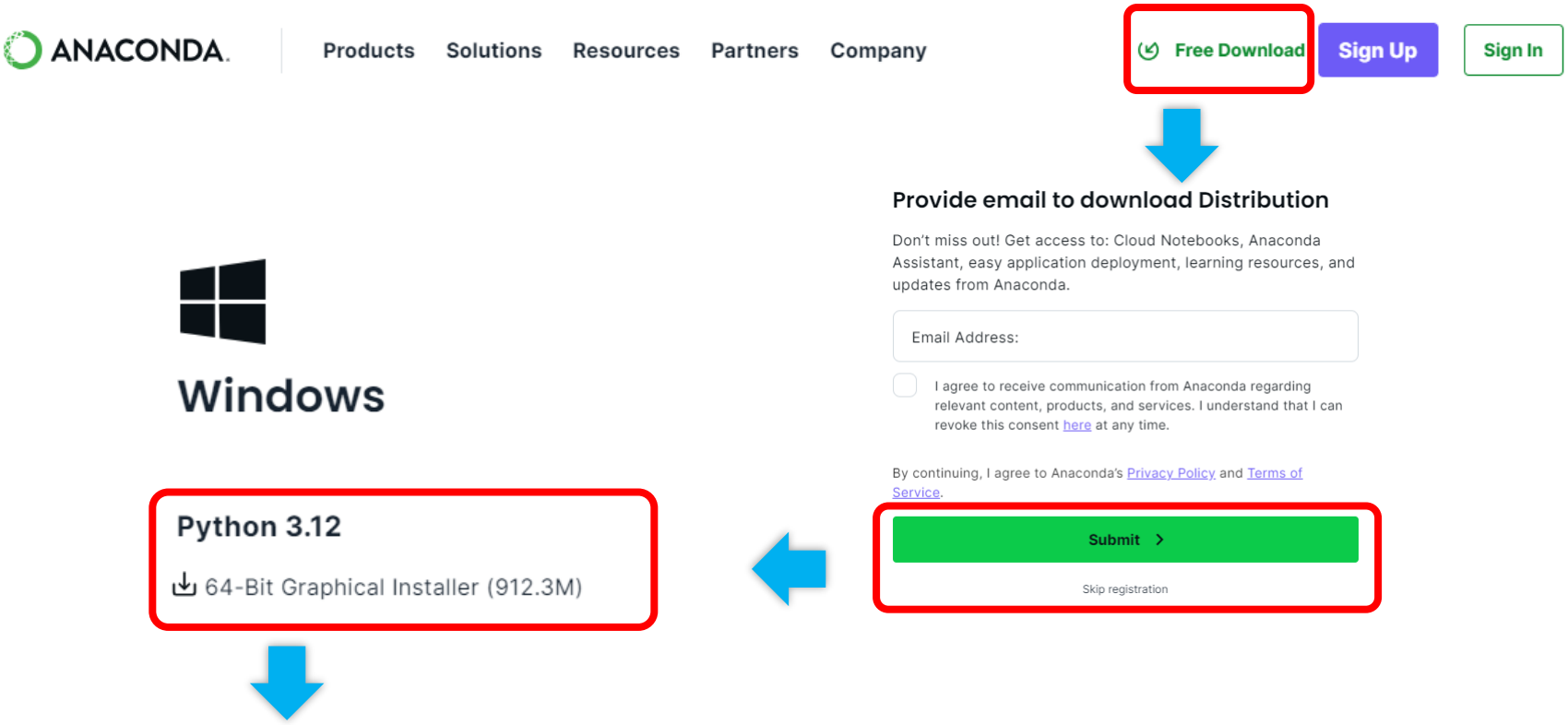
- The world's most popular open-source Python distribution platform
- Anaconda Distribution contains conda, which is a package and environment manager, which helps users to manage a collection of over 8,000 open-source data science and machine learning packages.



2 Python Programming

Python with ANACONDA

homepage : https://www.anaconda.com/



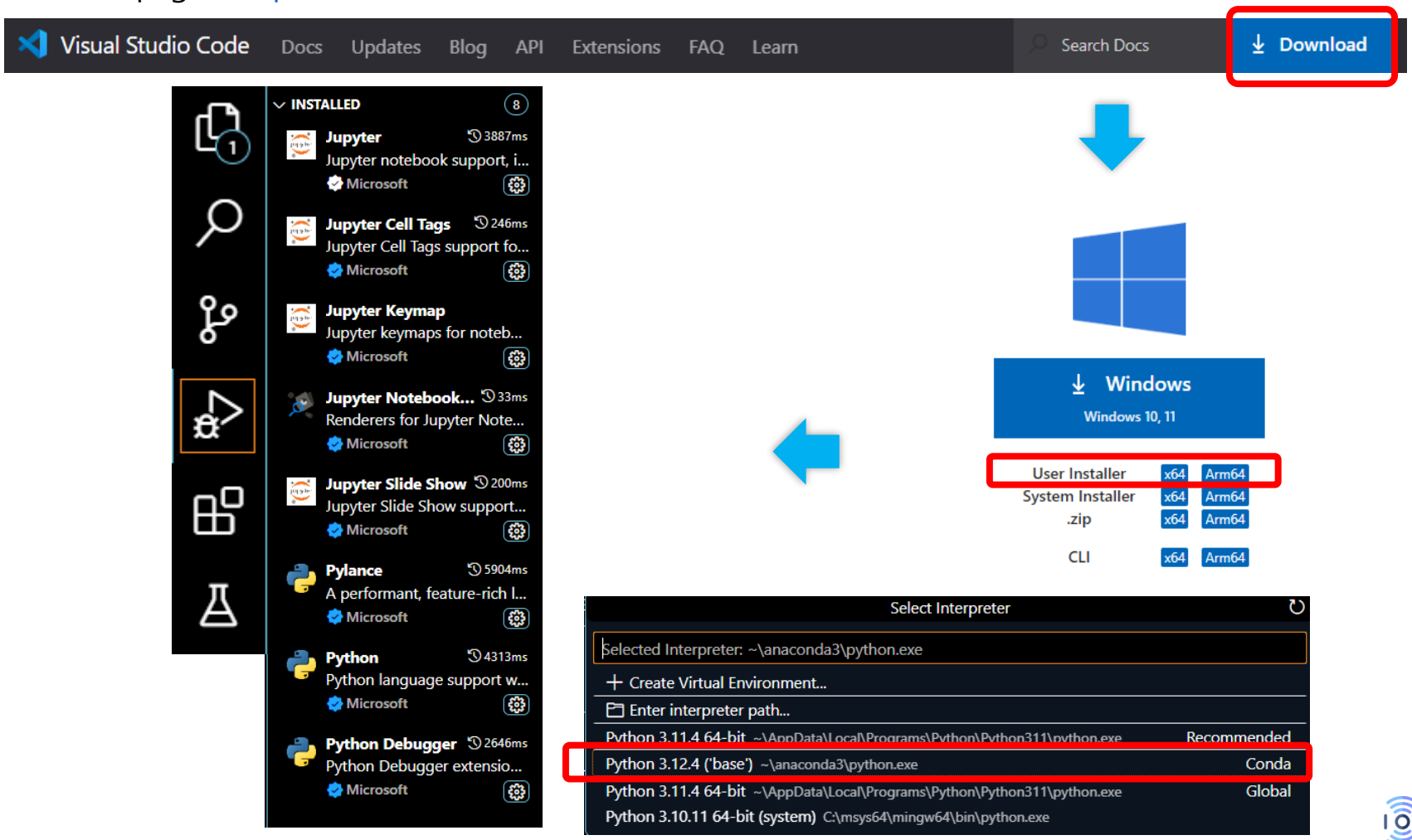
Install for : Just Me (recommended)

Advanced Options : Add Anaconda to my_PATH environment variable check

2 Python Programming

IDE Setting with Visual Studio Code

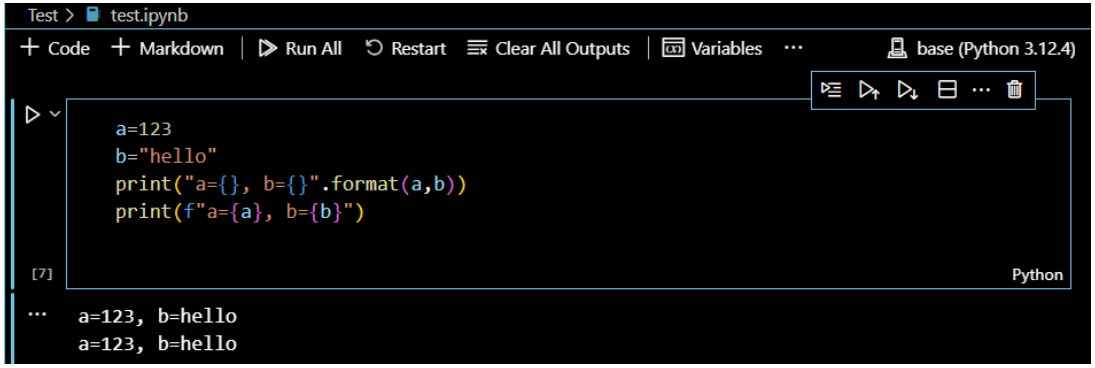
homepage : https://code.visualstudio.com/



IOT Expert Course

2 Python Programming

Python with Jupyter Notebooks



- A kernel is a “computational engine” that executes the code contained in a notebook document.
- A cell is a container for text to be displayed in the notebook or code to be executed by the notebook’s kernel.
- A code cell contains code to be executed in the kernel. When the code is run, the notebook displays the output below the code cell that generated it.
- A Markdown cell contains text formatted using Markdown and displays its output in-place
- when the Markdown cell is run.
- You can use Jupyter Notebooks files in VS Code with “~~~.ipynb” file extension.

Python Programming

Python Basics – Variables

Print()

```
print(“1 hello”)
print(“2 hello”, “hi”)
print(“3 hello”+”hi”)
print(“4 hello”, end=“”)
print(‘5 hello’)
print(‘6 hello’, “hi”)
print(“7 ‘hello’ “)
print(‘8 “hello” ‘)
```

```
a=123
b=“hello”
print(“a={}, b={}”.format(a,b))
print(f”a={a}, b={b}”)
```

Input()

```
input()

input(“Input value = “)

a = input(“Input 1st value = “)
b = input(“Input 2nd value = “)
print(a+b)
```

Python Programming

Python Basics – Variables

Variable Types

```
a=10
b=20
c=a+b
print(c)
```

30

```
e=3.14
f=10
print(e+f)
```

13.14

```
a=10
b=20
c=float(a)+float(b)
print(c)
```

30.0

```
d='10'
print(c+d)
```

TypeError

```
d='10'
print(c+int(d))
```

40

```
d='10'
print(str(c)+d)
```

3010

```
a_bool = True
b_bool = False
a_int = 1
b_int = 0
print(a_bool)
print(b_bool)
print(type(a_bool))
print(type(b_bool))
print(type(a_int))
print(type(b_int))
```

```
True
False
<class 'bool'>
<class 'bool'>
<class 'int'>
<class 'int'>
```

Python Programming

Python Basics – Data Types

List (use '[ ]')

```
a_list = [1,2,3,4,5]
print(a_list)
print(a_list[0])
print(a_list[1])
```

```
[1,2,3,4,5]
1
2
```

```
print(a_list[:2])
print(a_list[2:])
```

```
[1,2]
[3,4,5]
```

```
b_list = []
b_list.append(1)
b_list.append(2)
b_list.append(3)
print(b_list)
```

```
[1,2,3]
```

```
c_list = [ 1, 3.14, 'hello', [1,2,3] ]
print(c_list)
print(c_list[1:3])
```

```
[1, 3.14, 'hello', [1,2,3] ]
[3.14, 'hello' ]
```

```
d_list = [ 1, 2, 3, 4, 5 ]
print(d_list)
d_list[0] = 5
print(d_list)
```

```
[1, 2, 3, 4, 5 ]
[5, 2, 3, 4, 5 ]
```

Python Programming

Python Basics – Data Types

Tuple

- Tuple is similar to list, but values are immutable.(use '( )')

```
a_tuple = (1,2,3,4,5)
print(a_tuple)
print(a_tuple[0])
a_tuple[0] = 5
```

```
(1,2,3,4,5)
1
TypeError
```

Set

- Set removes duplicated data and doesn't sort it in order.(use '{ }')

```
a_set = set( [1,2,3,4] )
print(a_set)
b_set = set( [1,1,2,2,3,4] )
print(b_set)
```

```
{1,2,3,4}
{1,2,3,4}
```

```
c_set = set("python40s")
print(c_set)
```

```
{'o', 'h', 't', 'n', 'y', '4', 's', '0', 'p'}
```

Python Programming

Python Basics – Data Types

Dictionary

- Dictionary consists of keys and values.
- Dictionary is expressed in the following form: { key1:value1, key2:value2, key3:value3}

```
a_dic = {'a':1, 'b':2, 'c':'3'}
print(a_dic)
print(a_dic['a'])
print(a_dic['c'])
```

```
{'a':1, 'b':2, 'c':'3'}
1
3
```

```
b_dic = {1:'a', 'b':[1,2,3], 'c':3}
print(b_dic[1])
print(b_dic['b'])
print(b_dic['c'])
```

```
a
[1, 2, 3 ]
3
```

```
b_dic['d'] = 4
print(b_dic)
```

```
{ 1:'a', 'b':[1,2,3], 'c':3, 'd':4 }
```

Python Programming

Python Basics – Operator

Arithmetic

```
print("Add : ", 10+20)
print("Sub : ", 10-20)
print("Mul : ", 10*20)
print("Div : ", 10/20)
```

```
Add : 30
Sub : -10
Mul : 200
Div : 0.5
```

```
print( 10**2 )
print( 10**3 )
print( 10**4 )
```

```
100
1000
10000
```

```
print( 40//6 )
print( 40%6 )
```

```
6
4
```

Logic

```
print(0 or 0)
print(0 or 1)
print(1 or 0)
print(1 or 1)
print(False or False)
print(False or True)
```

```
0
1
1
1
False
True
```

```
print(0 and 0)
print(0 and 1)
print(1 and 0)
print(1 and 1)
print(False and True)
print(True and True)
```

```
0
0
0
1
False
True
```

```
print( not 0)
print( not 1)
```

```
True
False
```

```
print( not False)
print( not True)
```

```
True
False
```

Python Programming

Python Basics – Operator

Comparison

```
print( 10 == 10 )
print( 10 >= 10 )
print( 10 <= 10 )
print( 10 < 5 )
print( 10 > 5 )
print( 10 != 10 )
```

```
True
True
True
False
True
False
```

```
a_list = [ 'a', 2, 'hello', 3 ]
print( 'a' in a_list )
print( 1 in a_list )
print( 'hello' in a_list )
print( '3' in a_list )
```

```
True
False
True
False
```

```
a_str = "hello python"
print( "python" in a_str )
print( "py" in a_str )
print( "40" in a_str )
```

```
True
True
False
```

Python Programming

Python Basics – Condition

If

```
A = 1
B = 2
if A == B :
    print( “Equal” )
else :
    print( “Unequal” )
```

Unequal

```
a_str = “hello python”
if a_str == “hello python” :
    print(“string is equal.”)
if ‘hi’ not in a_str :
    print(“hi is not included.”)
```

string is equal.

hi is not included.

```
A = 1
B = 2
if A > B :
    print( “A is bigger” )
elif A<B :
    print( “B is bigger” )
else :
 print( “Equal” )
```

B is bigger

```
a_list = [“hello”, 1, 2, ‘python’]
if “hello” in a_list :
    print(“string is included.”)
if 2 not in a_list :
    print(“it’s not working.”)
```

string is included.

Python Programming

Python Basics – Loop

For

```
for i in range(7) :
    print(i)
```

```
0
1
2
3
4
5
6
```

```
a_list = [1,2,3,4,5, 'hi', 'good']
for i in a_list :
    print(i)
```

```
1
2
3
4
5
hi
good
```

```
for i in range(5, 10) :
    print(i)
```

```
5
6
7
8
9
```

```
word_list = ['kim', 'dae', 'sung']
num_list = [ 100, 200, 300 ]
for i, k in enumerate(word_list) :
    print(k, end=' ')
    print(num_list[i])
for i, k in enumerate(word_list) :
    print(word_list[i], end=' ')
    print(num_list[i])
```

```
kim 100
dae 200
sung 300
kim 100
dae 200
sung 300
```

```
for i in range(10, 5, -1) :
    print(i)
```

```
10
9
8
7
6
```

Python Programming

Python Basics – Loop

For

```
word_list = [‘kim’, ‘dae’, ‘sung’]
num_list = [ 100, 200, 300 ]
for i in range(len(word_list)) :
    print(word_list[i], end=‘ ’)
    print(num_list[i])
```

```
kim 100
dae 200
sung 300
```

```
test_list = [ i for i in range(5) ]
print(test_list)

test2_list = []
for i in range(len(test_list)) :
    test2_list.append(i)
print(test2_list)
```

```
[0, 1, 2, 3, 4]
[0, 1, 2, 3, 4]
```

```
test_list = [ i*5 for i in range(5) ]
print(test_list)

test2_list = [ 0 for i in range(5)]
print(test2_list)
```

```
[0, 5, 10, 15, 20]
[0, 0, 0, 0, 0]
```

Python Programming

Python Basics – Loop

While

```
a = 0
while a<5 :
    print(a)
    a = a+1
```

```
0
1
2
3
4
```

```
a = 0
while True :
    print(a)
    a = a+1
    if a>=5 :
        break
```

```
0
1
2
3
4
```

Python Programming

Python Basics – Error and Exception

Try / Except

```
try :
    dlksjdslkjsfd
except:
    print('Error!!')
```

Error!!

```
try :
    dlksjdslkjsfd
except:
    pass
print('Skip Error')
```

Skip Error

```
try :
    dlksjdslkjsfd
except Exception as e :
    print('Error : ', e)
```

Error : name 'dlksjdslkjsfd' is not defined

Python Programming

Python Basics – Function

Using functions

```
def func():
    print("This is func print.")
func()
```

```
This is func print.
```

```
def func_add(a,b) :
    return a+b

c = func_add(1,2)
print(c)
```

```
3
```

```
def func_add_mul(a,b) :
    add = a+b
    mul = a*b
    return add, mul

c, d = func_add_mul(3,4)
print(c, d)
```

```
7  12
```

```
def func_add_mul(a,b) :
    add = a+b
    mul = a*b
    return add, mul

_, d = func_add_mul(3,4)
print(d)
```

```
12
```

Python Programming

Python Basics – Class

Declaring and using Class

```
class Greet():
    def hello(self):
        print("hello")
    def hi(self):
        print("hi")


human1 = Greet()
human2 = Greet()
human1.hello()
human1.hi()
human2.hello()
human2.hi()
```

```
hello
hi
hello
hi
```

```
class student():
    def __init__(self, name, age, like):
        self.name = name
        self.age = age
        self.like = like
    def stu_info(self):
        print(f'{self.name} / {self.age} / {self.like}')


A = student("kim", 17, 'Boxing')
B = student("dae", 27, 'Game')
A.stu_info()
B.stu_info()
```

```
kim / 17 / Boxing
dae / 27 / Game
```

Python Programming

Python Basics – Class

Declaring and using Class

```
class mom():
    def characteristic(self):
        print("Tall")
        print("Smart")
class daughter(mom):
    def characteristic(self):
        super().characteristic()
        print("Strong")

M = mom()
D = daughter()
print("[ Mother ]")
M.characteristic()
print("[ Daughter ]")
D.characteristic()
```

```
[ Mother ]
Tall
Smart
[ Daughter ]
Tall
Smart
Strong
```

```
class mom():
    def __init__(self):
        print("Tall")
        print("Smart")
class daughter(mom):
    def __init__(self):
        super().__init__()
        print("Strong")

print("[ Mother ]")
M = mom()
print("[ Daughter ]")
D = daughter()
```

```
[ Mother ]
Tall
Smart
[ Daughter ]
Tall
Smart
Strong
```

Python Programming

Python Basics – Comments

Using Comments

```
# Comment1
print("hello") # Comment2
```

```
hello
```

```
"""
Comment line 1
Comment line 2
Comment line 3
"""
```

```

```

```
a_str = """
line 1
line 2
line 3
"""


print(a_str)
```

```
line 1
line 2
line 3
```

- Drag the code areas and press [Ctrl + /] : Codes ←→ Comments

Python Programming

Python Basics – import

- Using import to call library or module

```
import random
print(random.randint(1,100))
```

63

```
import random as rd
print(rd.randint(1,100))
```

59

```
from random import randint
print(randint(1,100))
```

30

```
from random import *
print(randint(1,100))
```

2

Python Programming

Python Projects – 1) Number Guessing Game

- Make Python code that can work as following:

```
Input number 1~99 : eer
Error :  invalid literal for int() with base 10: 'eer'
Input number 1~99 : 50
Up
Input number 1~99 : 80
Down
Input number 1~99 : 60
Down
Input number 1~99 : 55
Down
Input number 1~99 : 53
Up
Input number 1~99 : 54
Congratulations, you got it right in 6 tries.
```

Python Programming

Python Projects – 2) Verifying IP Address

- Internal IPs are assigned by one's router

```
import socket
in_addr = socket.gethostbyname(socket.gethostname())
print(In_addr)
```

192.168.100.55
Different addresses are output depending on the assigned IPs

```
import socket
In_IP_addr = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
In_IP_addr.connect(("www.google.com", 443)) #https port is 443
print(In_IP_addr.getsockname()[0])
```

192.168.100.55
After accessing an external website, check the IP based on the information connected.

Python Programming

Python Projects – 2) Verifying IP Address

- External IPs are assigned by Internet Service Provider.

```
import socket
import requests
import re


in_addr = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
in_addr.connect(("www.google.com",443))
print("Int. IP : ", in_addr.getsockname()[0])


req = requests.get("http://ipconfig.kr")
out_addr = re.search(r'IP Address : (\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})', req.text)[1]
print("Ext. IP : ", out_addr)
```

```
Int. IP :  192.168.100.55
Ext. IP :  1.220.40.94
```

Different addresses are output depending on the assigned IPs

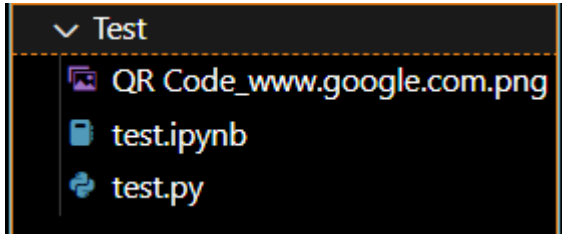
Python Programming

Python Projects – 4) Qr Codes

Using qrcode Library

```
pip install qrcode
Type these commands on terminal window
```

```
import qrcode

qr_data = 'www.google.com'
qr_img = qrcode.make(qr_data)

save_path = 'QR Code_'+qr_data+'.png'
qr_img.save(save_path)
```



Python Programming

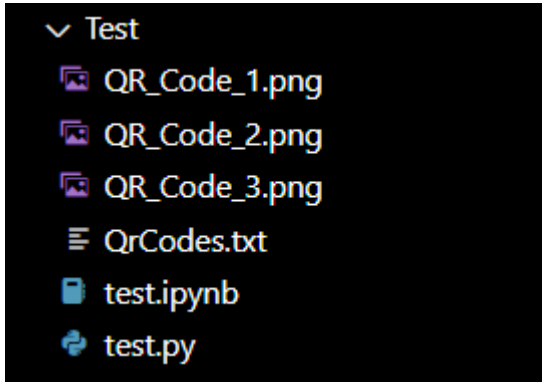
Python Projects – 4) Qr Codes

Make Qr Codes from *.txt file

```
import qrcode

file_path = r'QrCodes.txt'

with open(file_path, 'rt', encoding='UTF8') as f :
    read_lines = f.readlines()
    cnt=0
    for line in read_lines :
        cnt = cnt+1
        line = line.strip()
        print(line)
        qr_data = line
        qr_img = qrcode.make(qr_data)

        save_path = 'QR_Code'+ str(cnt) + '.png'
        qr_img.save(save_path)
```

QrCodes.txt

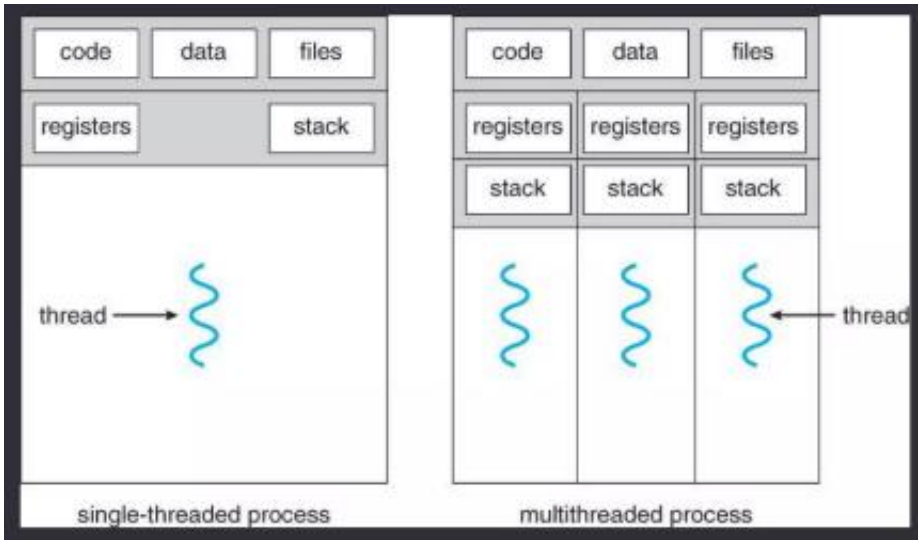
```
1  www.google.com
2  https://docs.python.org/3/tutorial/index.html
3  12345/678910/11
```



Python Projects – 5) Thread programming

What is thread?

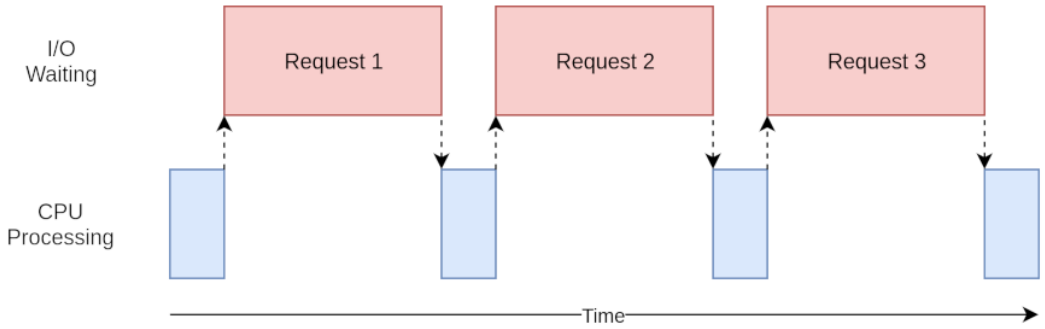
- A thread is a flow of execution through the process code, with its own program counter, system registers and stack.
- A thread is also called a lightweight process. Threads provide a way to improve application performance through parallelism.
- Each thread belongs to exactly one process and no thread can exist outside a process.



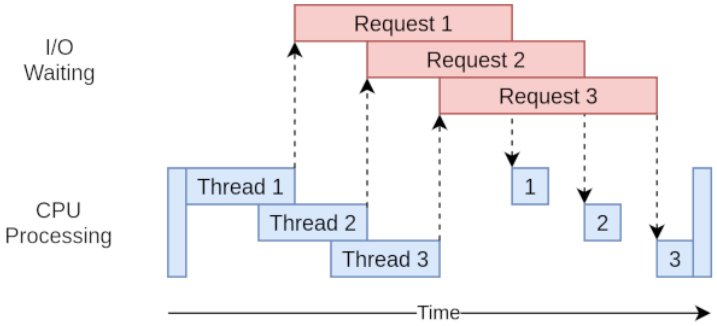
Python Programming

Python Projects – 5) Thread programming

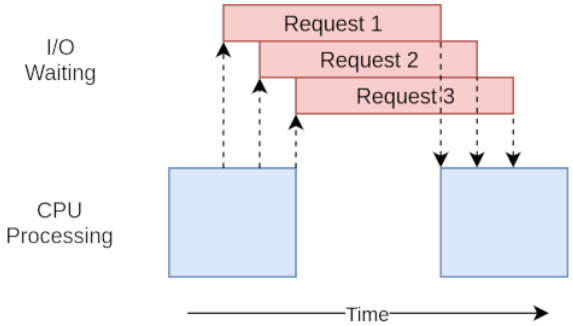
- What is thread?



(a) Single-threaded process



(b) Multi-threaded process
with GIL acquired by current thread



(c) Single-threaded process
with asyncio

Python Programming

Python Projects – 5) Thread programming

- Thread function only runs when the main code is executed.

```
import threading
import time

def thread_1():
    while True:
        print('Thread 1 is working')
        time.sleep(0.7)

t1 = threading.Thread(target=thread_1)
t1.daemon = True
t1.start()

while True:
    print('Main loop')
    time.sleep(1)
```

Python Programming

Python Projects – 5) Thread programming

Multi threads code stop by event

```
import threading
import time

def thread_1(stop_thread_1):
    print('Thread_1 Start')
    while True:
        print('Thread_1 working')
        time.sleep(0.5)
        if stop_thread_1.is_set() :
            print('Thread_1 is terminated')
            break

def thread_2(stop_thread_2):
    print('Thread_2 Start')
    while True:
        print('Thread_2 working')
        time.sleep(0.7)
        if stop_thread_2.is_set() :
            print('Thread_2 is terminated')
            break
```

```
stop_thread_1 = threading.Event()
stop_thread_2 = threading.Event()
t1 =  threading.Thread(target=thread_1, args=(stop_thread_1,))
t1.start()
t2 =  threading.Thread(target=thread_2, args=(stop_thread_2,))
t2.start()



cnt=0
while True:
    cnt = cnt+1
    print('Main code working')
    print(cnt)
    time.sleep(1)
    if cnt>10 :
        stop_thread_1.set()
        t1.join()
    if cnt>15 :
        stop_thread_2.set()
        t2.join()
```

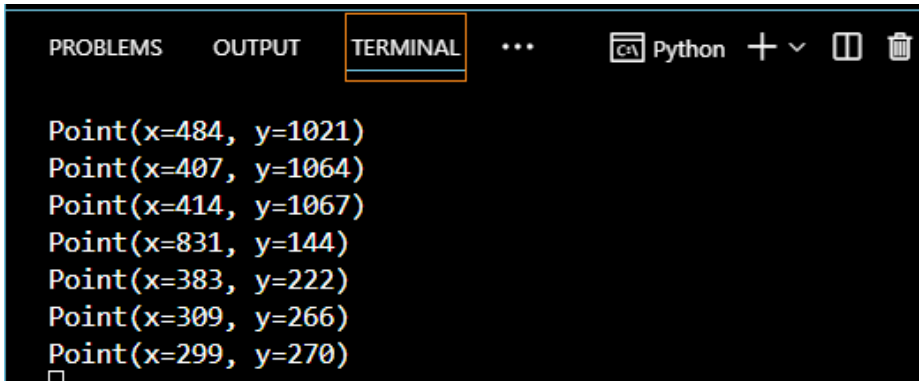
Python Programming

Python Projects – 6) Auto mouse

- Output the current mouse position

```
pip install pyautogui
pip install pyperclip
Type these commands on terminal window
```

```
import pyautogui
import time

while True:
    print(pyautogui.position())
    time.sleep(0.3)
```



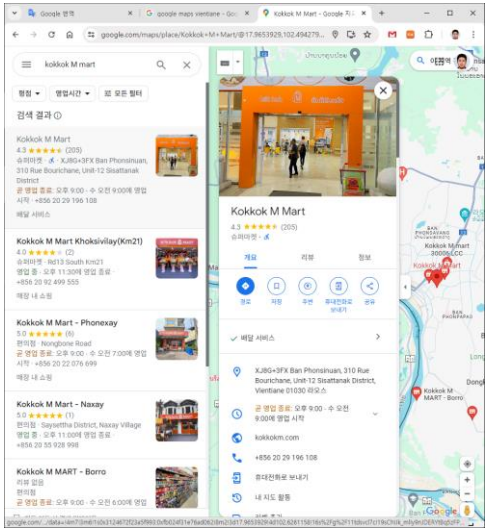
Python Programming

Python Projects – 6) Auto mouse

- Output the current mouse position

```
import pyautogui
import time
import pyperclip

pyautogui.moveTo(290,20,0.1)
pyautogui.click()
time.sleep(0.5)

pyperclip.copy("google maps vientiane")
pyautogui.hotkey('ctrl','v')
pyautogui.hotkey('enter')
time.sleep(2)

pyautogui.moveTo(375,400,0.2)
pyautogui.click()
time.sleep(4)
```

```
pyautogui.moveTo(110,120,1)
pyautogui.doubleClick()
pyperclip.copy("kokkok M mart")
pyautogui.hotkey('ctrl','v')
pyautogui.hotkey('enter')
time.sleep(2)

pyautogui.moveTo(100,260,0.2)
pyautogui.click()
```

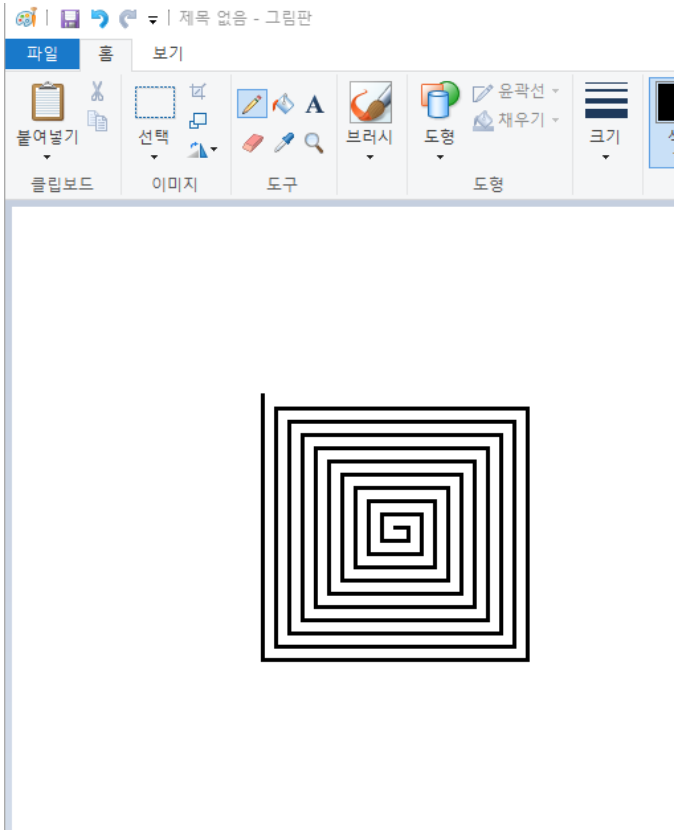


Python Programming

Python Projects – 6) Auto mouse

- Make Python code that can work as following:

  Make a python code that draw the line automatically.

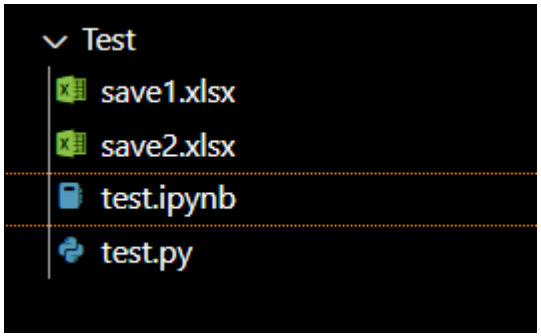


Python Programming

Python Projects – 7) Excel file R/W

- Write data as ‘data1.xlsx’ . Read data from saved file and change Row & Col

```
import pandas as pd
from openpyxl import load_workbook

d1 = ["kim", "dae", "sung"]
d2 = ["1977", "2000", "2024"]
d3 = ["1", "23", "47"]
d4 = ['22','33','55']
data = [d1,d2,d3,d4]
df1 = pd.DataFrame(data)
print(df1)
df1.to_excel(r'save1.xlsx', index=False, header=False)

load_wb = load_workbook(r'save1.xlsx')
load_ws = load_wb.active

Rld_data = [[0 for j in range(load_ws.max_row)] for i in range(load_ws.max_column)]
for r in range(1, load_ws.max_column+1):
    for c in range(1, load_ws.max_row+1):
        Rld_data[r-1][c-1] = load_ws.cell(c,r).value
df2 = pd.DataFrame(Rld_data)
print(df2)
df1.to_excel(r'save2.xlsx', index=False, header=False)
```

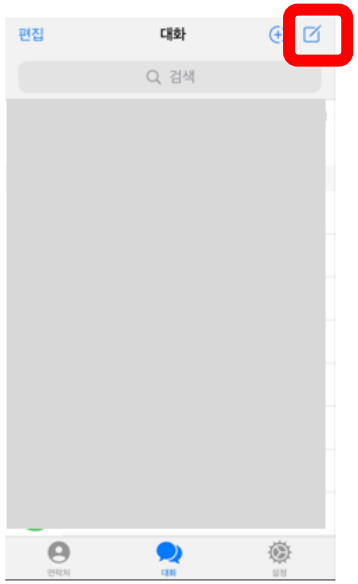


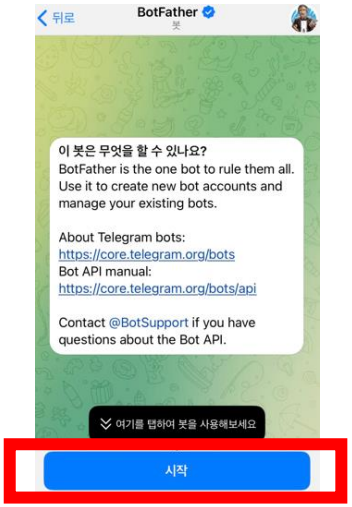

```
      0     1     2
0   kim   dae  sung
1  1977  2000  2024
2     1    23    47
3    22    33    55
      0     1   2   3
0   kim  1977   1  22
1   dae  2000  23  33
2  sung  2024  47  55
```

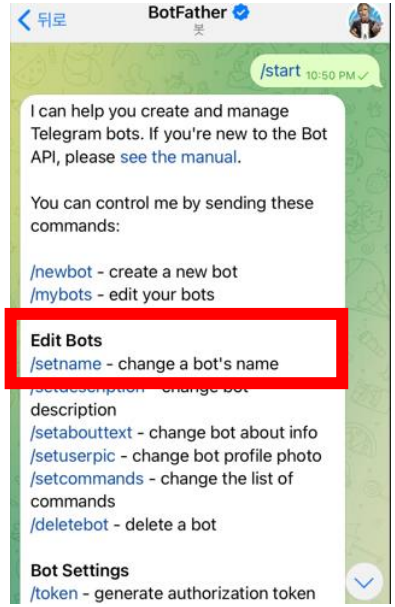
Python Programming

Python Projects – 8) Telegram

pip install python-telegram-bot
pip install asyncio
Type these commands on terminal window





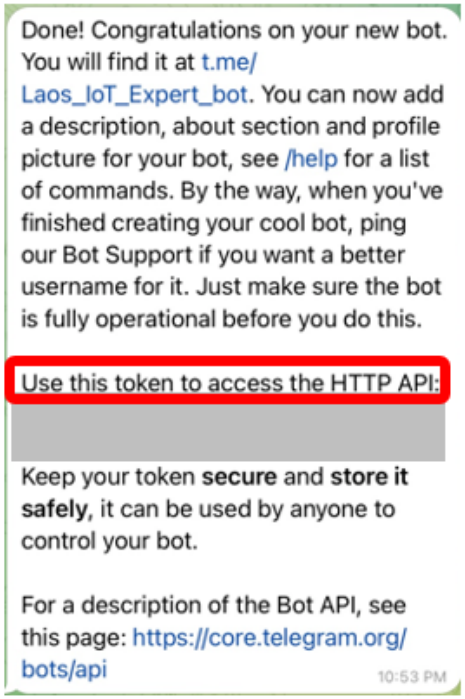


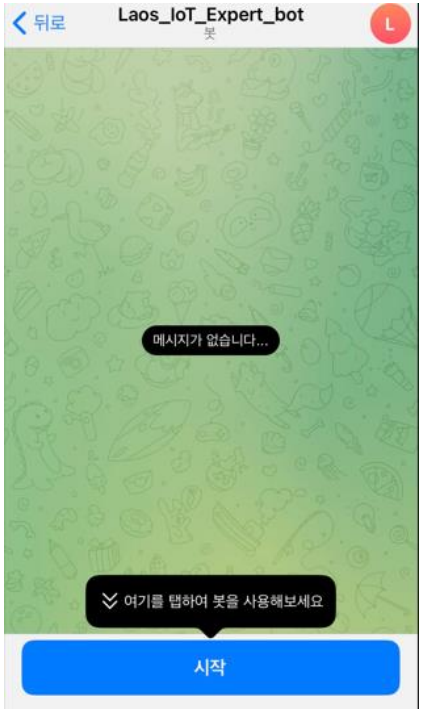
/newbot – create a new bot

Python Programming

Python Projects – 8) Telegram

Naming a new bot





Python Projects – 8) Telegram

Find bot ID

- Type URL to Web browser as follow :

```
https://api.telegram.org/bot__API_Token__/getUpdates
Change __API_Token__ to you bot API value
```

```
{"ok":true,"result":[ ]}
```

- Send a message to your bot

- Refresh the web browser.

```
{"ok":true,"result":[{"update_id":9348484847,
"message":{"message_id":8,"from":{"id":__Your_ID__,"is_bot":false,"first_name":"Daesung","last_name":"Kim","language_code":"ko"},"chat":{"id":12345678,"first_name":"Daesung","last_name":"Kim","type":"private"},"date":12344566544,"text":"hi"}}]}
```

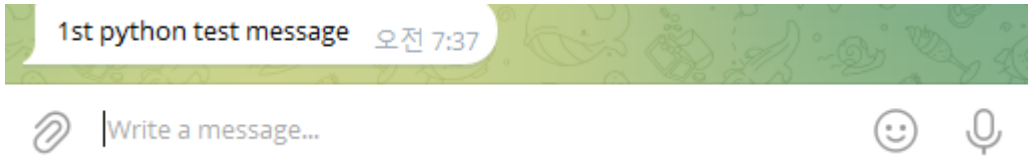
Python Programming

Python Projects – 8) Telegram

Send message to telegram bot

- Type URL to Web browser as follow :

```
import asyncio
import telegram
token = '__API_Token__'
chat_id = '__Your_ID__'
message = '1st python test message'
Message = ‘
async def main():
    bot = telegram.Bot(token)
    async with bot:
        await bot.send_message(text=message, chat_id=chat_id)
if __name__ == '__main__':
    asyncio.run(main())
```



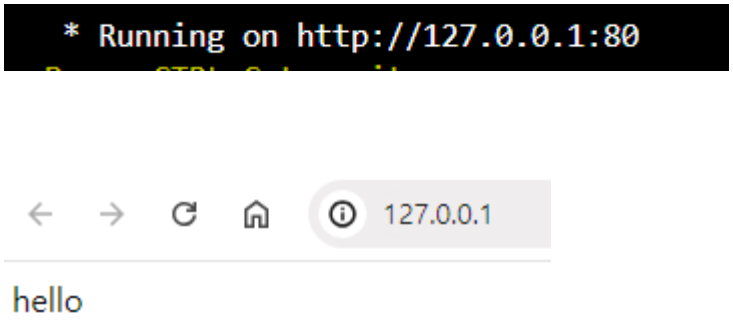
Python Programming

Python Projects – 9) Flask Web Server

- Make a simple web server using flask library

```
pip install flask
```
Type this command on terminal window

```
from flask import Flask

app = Flask(__name__)
@app.route('/')
def hello():
    return 'hello'

def main():
    app.run(debug=True, port=80)

if __name__ == '__main__' :
    main()
```

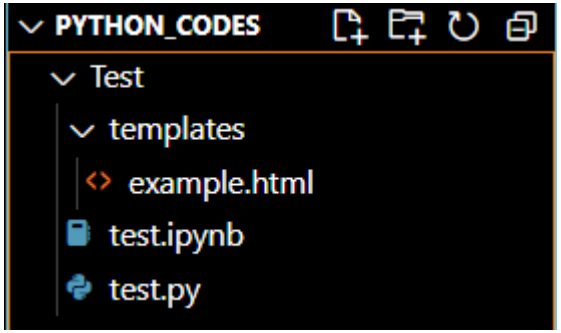


Python Programming

Python Projects – 9) Flask Web Server

- Make a simple web server using flask library

```
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/')
def hello():
    return 'hello'
@app.route('/1')
def test1page():
    return 'page 1'
@app.route('/ex')
def example():
    return render_template('example.html')

def main():
    app.run(debug=True, port=80)
if __name__ == '__main__' :
    main()
```




```
<!DOCTYPE html>
<html>
<body>

<a href="https://www.google.com">This is a link</a>

</body>
</html>
```

Python Programming

Python Projects – 10) GUI

- Make a simple gui using tkinter library

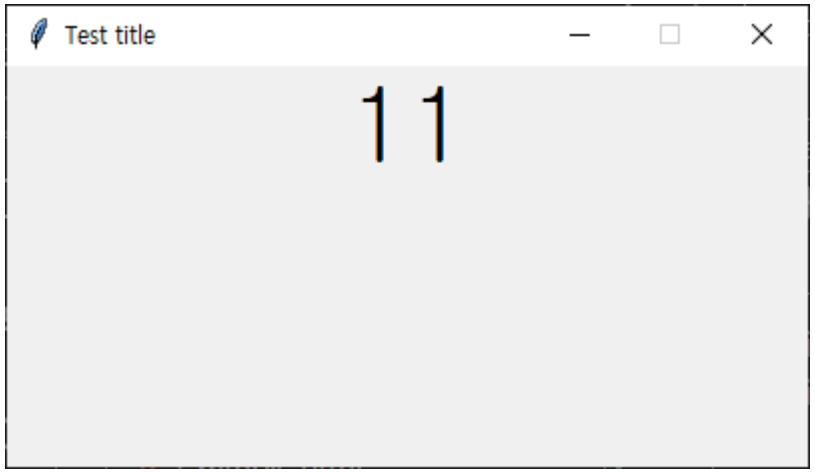
```
import tkinter
import tkinter.font

win = tkinter.Tk()
win.title("Test title")
win.geometry("400x200")
win.resizable(False,False)

font = tkinter.font.Font(size=40)
label = tkinter.Label(win, text='hello',
font=font)
label.pack()

win.mainloop()
```

Python Programming

Python Projects – 10) GUI

- Make a simple gui using tkinter library

```
import tkinter
import tkinter.font

win = tkinter.Tk()
win.title("Test title")
win.geometry("400x200")
win.resizable(False,False)

font = tkinter.font.Font(size=40)
label = tkinter.Label(win, text='', font=font)
label.pack()

cnt = 0
def run_1sec():
    global cnt
    cnt = cnt+1
    label.config(text=str(cnt))
    win.after(1000, run_1sec)

run_1sec()
win.mainloop()
```



Python Programming

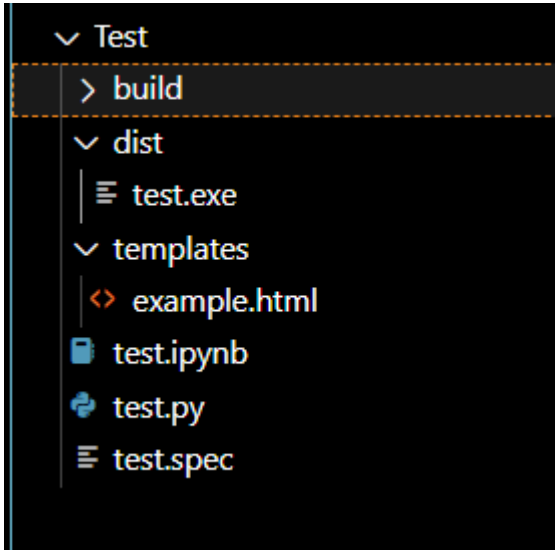
Python Projects – 11) Making an execution file.

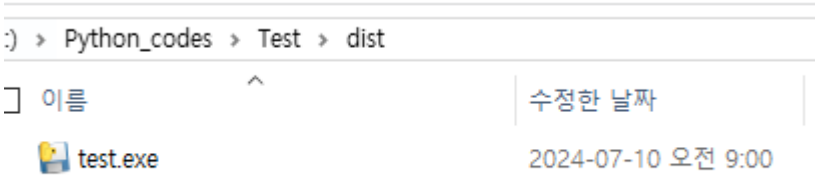
- Make an execution file using pyinstaller library

```
pip install pyinstaller
```
Type this command on terminal window

```
(base) C:\Python_codes\Test> pyinstaller –w –F test.py
```
Move to your python code folder

- Check your folder/dist



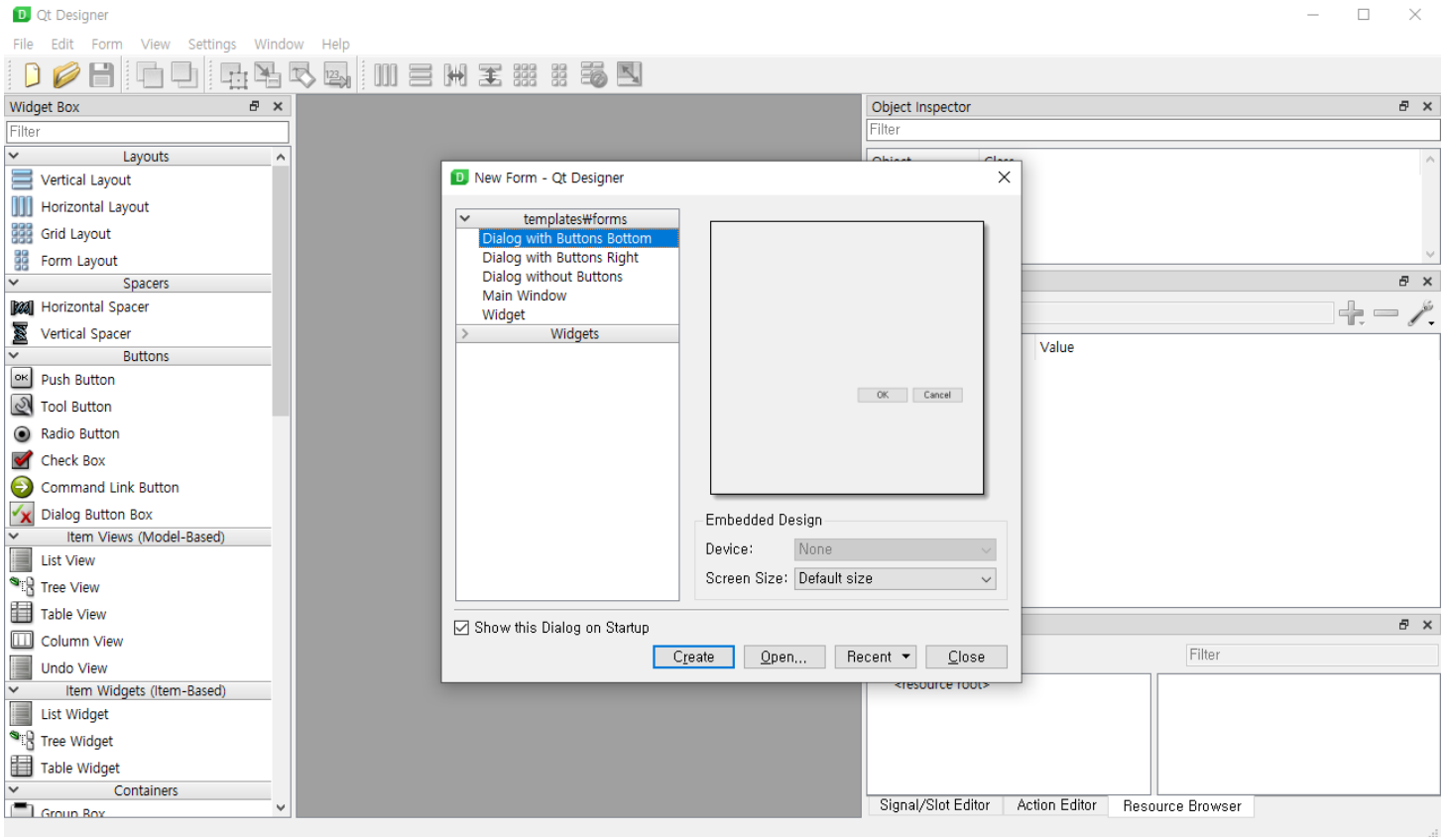


Python Programming

Python Projects – 12) Qt Designer

- Make a GUI using Qt designer

```
(base) C:\Python_codes\Test> designer
Type command
```



Python Projects – 12) Qt Designer

- Make a Calculator using Qt designer

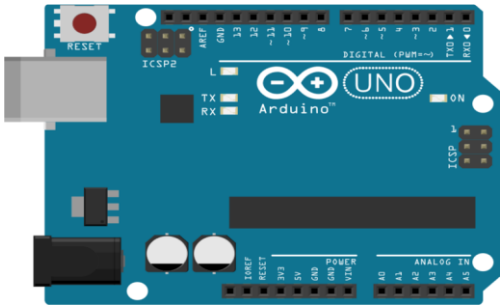
- Blank -

3 Raspberry Pi Pico & Sensors

3 Raspberry Pi Pico & Sensors

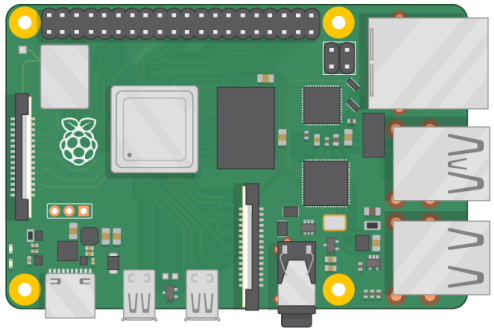
What is Raspberry Pi Pico

- Arduino Family



- Arduino UNO and similar Arduino boards is a Microcontroller Unit(MCU)
- Programming Language :
  - ✓ Arduino IDE and C/C++

- Raspberry Pi



- Raspberry Pi is a single-board computer(SBC), which is a microcontroller unit with CPU, RAM and external hard disk.
- OS : Linux
- Program language : Python + many others

- Raspberry Pi Pico

- Raspberry Pi is a Microcontroller Unit(MCU)
- Programming language :
  - ✓ Micro-Python
  - ✓ C/C++

3 Raspberry Pi Pico & Sensors

What is Raspberry Pi Pico

- Raspberry Pi Pico is a microcontroller board developed by the Raspberry Pi Foundation.
- Raspberry Pi Pico is a “downscaled” version of the original Raspberry Pi.
- Raspberry Pi Pico is more comparable with Arduino compared to the original Raspberry Pi.
- Raspberry Pi Pico is typically used for Electronics projects, IoT Applications, etc.
- Typically, ones use Micro-Python, which is a downscaled version of Python, to program it.

3 Raspberry Pi Pico & Sensors

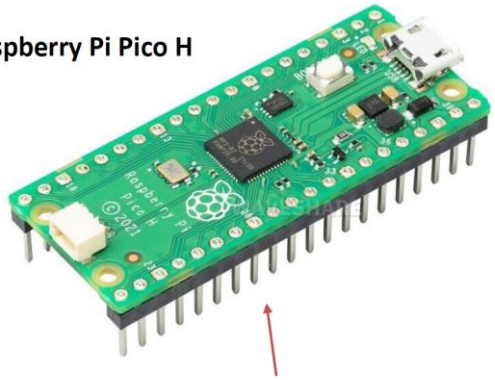
What is Raspberry Pi Pico

- 4 different types

- Raspberry Pi Pico (Org.)
- Raspberry Pi Pico H (Pre-soldered header pins included)
- Raspberry Pi Pico W (Wi-Fi included)
- **Raspberry Pi Pico WH (Wi-Fi and Pre-soldered header pins included)**

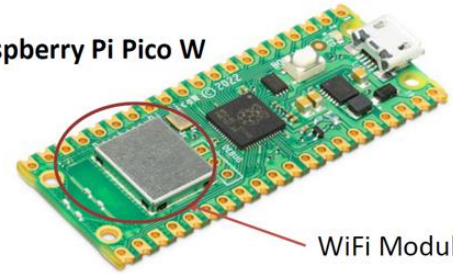
Raspberry Pi Pico (original)

Raspberry Pi Pico H



Pre-soldered header pins included

Raspberry Pi Pico W



WiFi Module and Antenna

3 Raspberry Pi Pico & Sensors

What is Raspberry Pi Pico

4 different types

- Raspberry Pi Pico includes a lot of features – all accessible using the pins around the edge of the board.
- Figure shows Raspberry Pi Pico as seen from above. If you look at the longer edges, you'll see gold-colored sections. These are the pins which provide the RP2040 microcontroller with connections to the outside, known as input/output (IO).

- The chip at the center of your Pico is a an RP2040 microcontroller. This is a custom integrated circuit(IC), designed and built specifically by Raspberry Pi's engineers to power your Pico and other microcontroller-based devices.

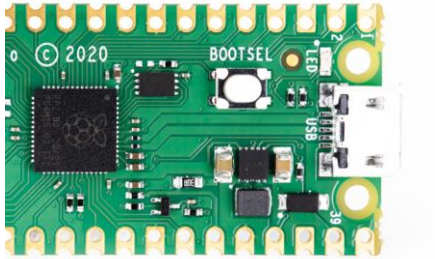
3 Raspberry Pi Pico & Sensors

What is Raspberry Pi Pico

4 different types

- At the top of your Pico is a micro-USB port. This provides power to make your Pico run and lets Pico communicate to other computer via its USB port – which is how you'll load your programs onto your Pico.

- Just below the micro-USB port is a small button marked 'BOOTSEL'. 'BOOTSEL' is short for 'boot selection', which switches your Pico between two start-up modes when it's first switched on.



3 Raspberry Pi Pico & Sensors

What is Raspberry Pi Pico

key features:

- RP2040 microcontroller with 2MByte Flash
- Micro-USB B port for power and data (and for reprogramming the Flash)
- 40 pin 21x51 'DIP' style 1mm thick PCB with 0.1" through-hole pins also with edge castellations
  - ✓ Exposes 26 multi-function 3.3V General Purpose I/O (GPIO)
  - ✓ 23 GPIO are digital-only and 3 are ADC capable
  - ✓ Can be surface mounted as a module
- 3-pin ARM Serial Wire Debug (SWD) port
- Simple yet highly flexible power supply architecture
  - ✓ Various options for easily powering the unit from micro-USB, external supplies or batteries
- High quality, low cost, high availability
- Comprehensive SDK, software examples and documentation

3 Raspberry Pi Pico & Sensors

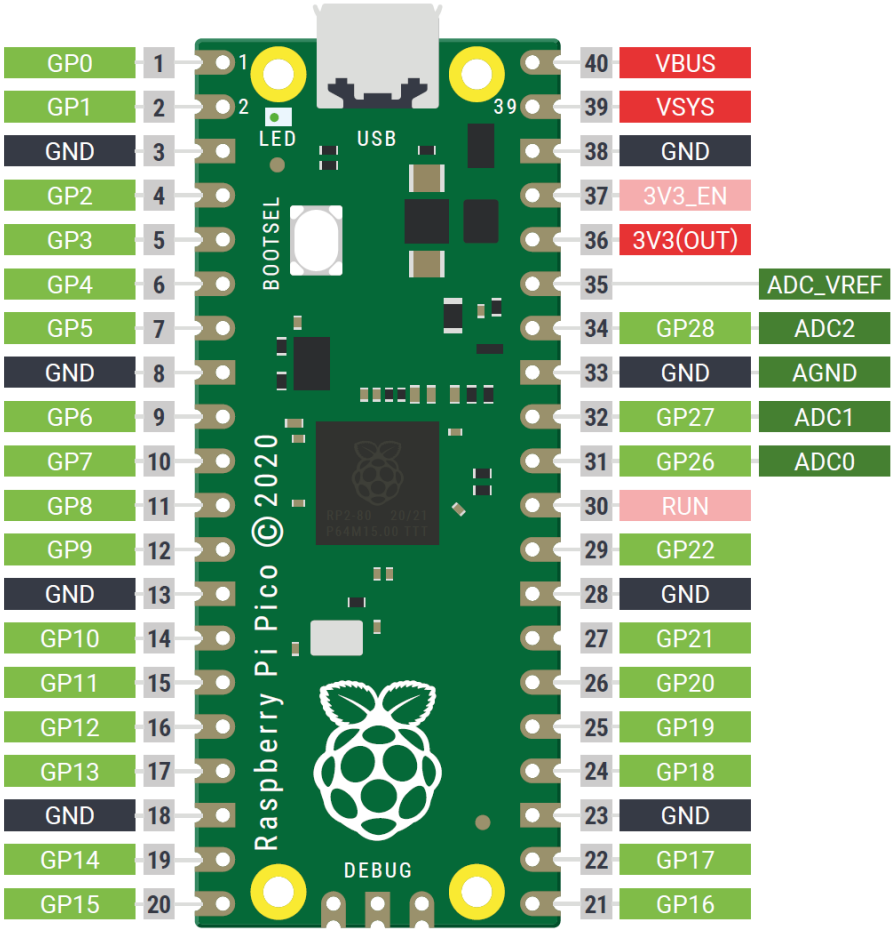
What is Raspberry Pi Pico

Pins

- Pico communicate to hardware through a series of pins along both its edges.

- Most of these pin's work as a general-purpose input/output (GPIO) pin, meaning they can be programmed to act as either an input or an output and have no fixed purpose of their own.

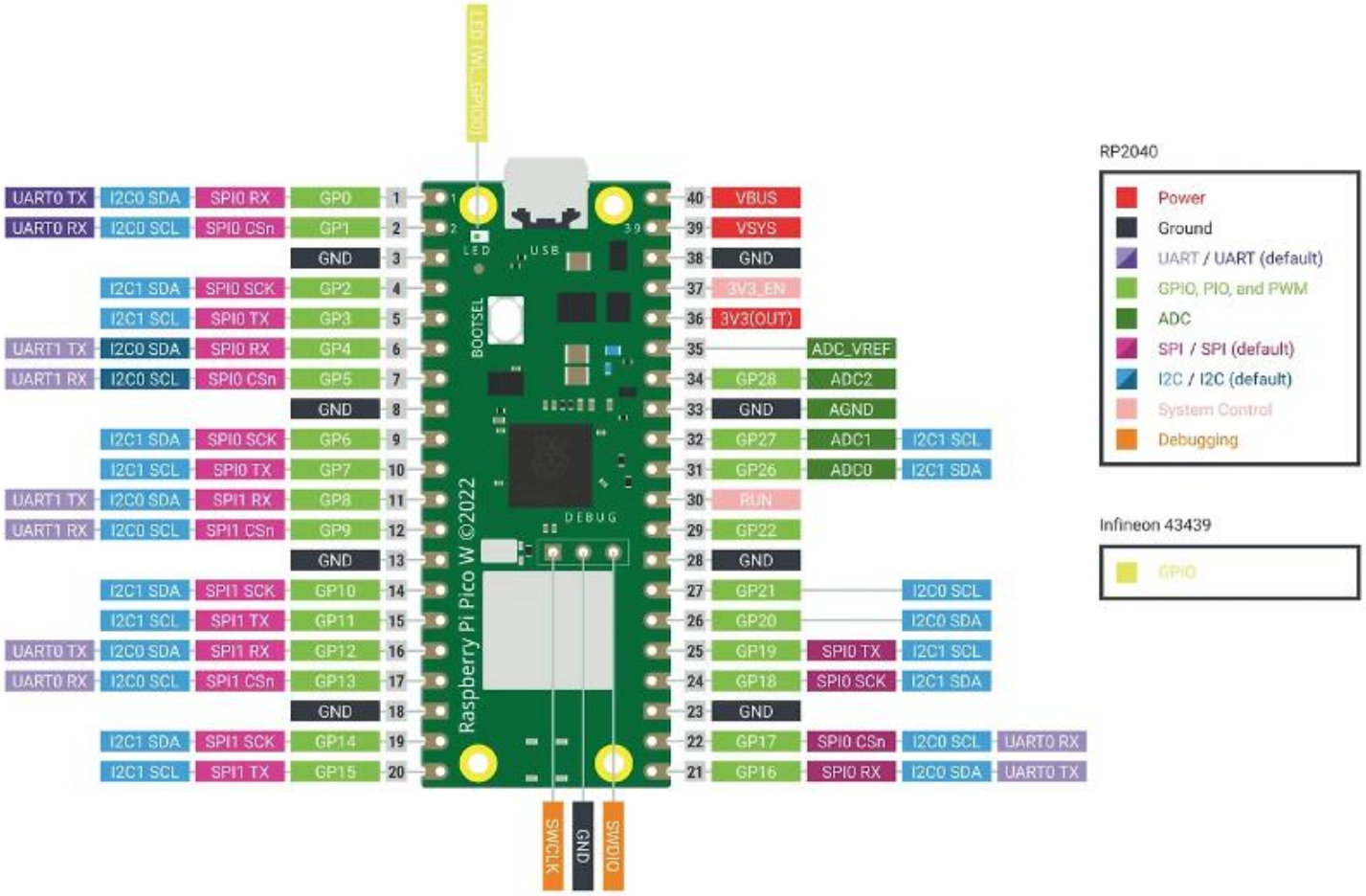
- Some pins have extra features and alternative modes for communicating with more complicated hardware; others have a fixed purpose, providing connections for things like power.



3 Raspberry Pi Pico & Sensors

What is Raspberry Pi Pico

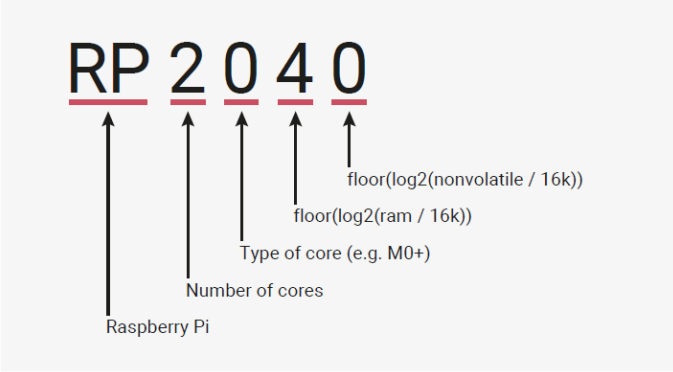
- Pinout guide



3 Raspberry Pi Pico & Sensors

What is Raspberry Pi Pico

RP2040

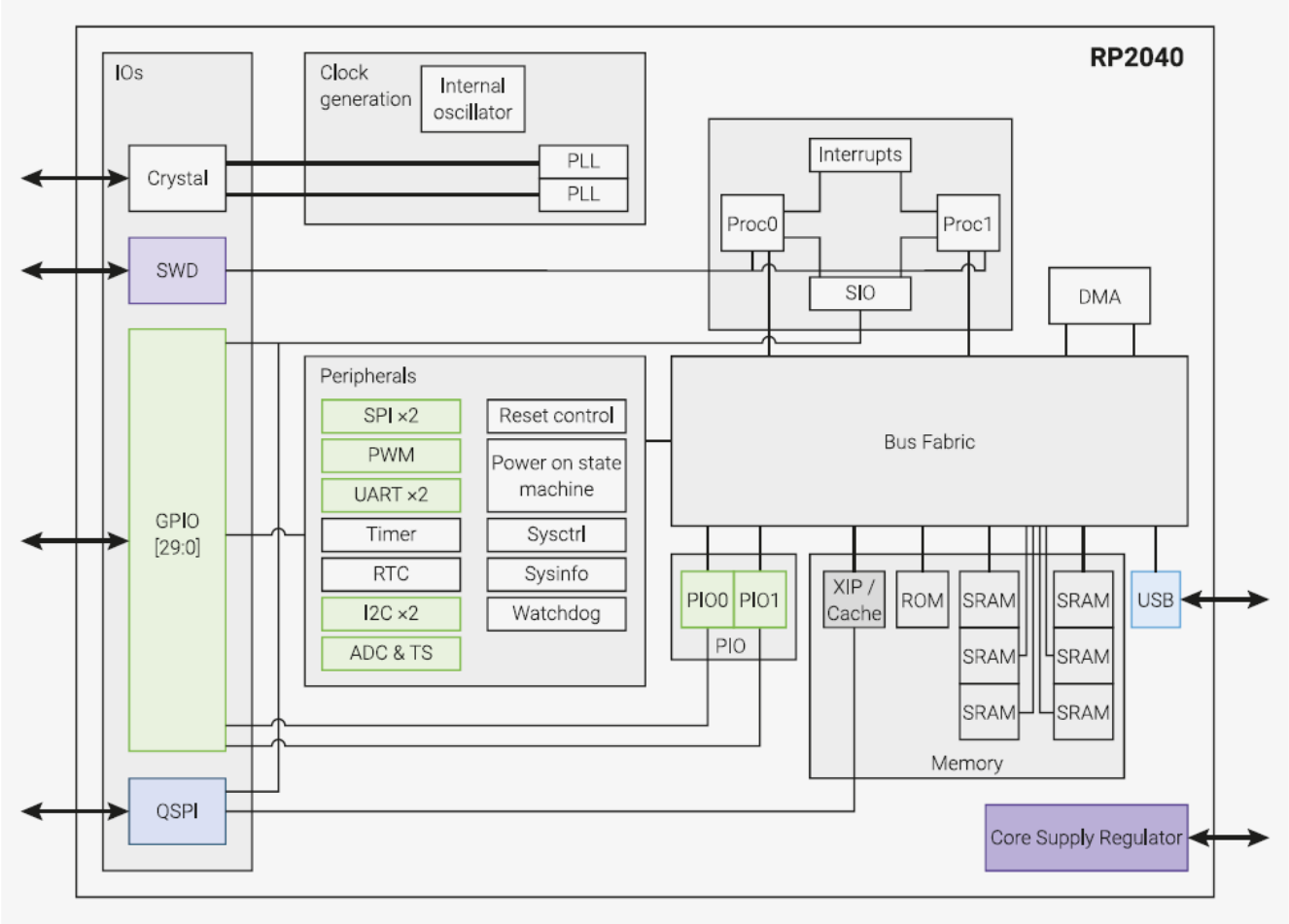


- RP2040 is a low-cost, high-performance microcontroller device with flexible digital interfaces. Key features:
  - ✓ Dual Cortex M0+ processors, up to 133 MHz
  - ✓ 264 kB of embedded SRAM in 6 banks
  - ✓ 30 multifunction GPIO
  - ✓ 6 dedicated IO for SPI Flash (supporting XIP)
  - ✓ Dedicated hardware for commonly used peripherals
  - ✓ Programmable IO for extended peripheral support
  - ✓ 4 channel ADC with internal temperature sensor, 0.5 MSa/s, 12-bit conversion
  - ✓ USB 1.1 Host/Device

3 Raspberry Pi Pico & Sensors

What is Raspberry Pi Pico

- RP2040



Raspberry Pi Pico & Sensors

Physical Computing

- When people think of ‘programming’ or ‘coding’, they’re usually – and naturally–thinking about software. Coding can be about more than just software, though: it can affect the real world through hardware. This is known as physical computing.

- As the name suggests, physical computing is all about controlling things in the real world with your programs: hardware, rather than software.

- When you set the program on your washing machine, change the temperature on your programmable thermostat, or press a button at traffic lights to cross the road safely, you’re using physical computing.

- These devices are typically controlled by a microcontroller very much like the one on your Raspberry Pi Pico.

Raspberry Pi Pico & Sensors

Pico Sensor Kit

- Yahboom Pico sensor kit is developed on the Raspberry Pi Pico.
- It contains 21 sensor modules and an expansion board tailored for the Raspberry Pi Pico.
- This Pico Sensor Kit supports examples for micro-python.

Raspberry Pi Pico & Sensors

Pico Sensor Kit

RGB light ring module

Eight programmable RGB lights form a ring, programmable RGB special effects

Color recognition module

It can identify the RGB values of different colors to determine the color

Vibration module

Contains a rotor motor, which drives the rotor to generate centrifugal force to cause vibration through the rotation of the motor

Photosensitive module

Sense the light intensity of the current environment, and make automatic switches based on this feature

Buzzer module

Receive PWM signals of different frequencies and play sounds of different tones

Button module

Used for key control, low-level effective

Ultrasonic module

Professional ultrasonic module, high precision, small blind area, fast feedback speed, very high market penetration rate

Sound sensor

When the environmental sound exceeds the threshold, the output terminal will have a level change, so that the level of the environmental sound can be detected

Human infrared module

The human body is detected by the pyroelectric principle. Once a person enters the detection area, a high-level signal will be sent.

Raspberry Pi Pico & Sensors

Pico Sensor Kit

Temperature and humidity module

Integrated DHT11 temperature and humidity sensor inside the module, stable performance, etc.

RGB three-color light

Light up three colors of red, green, and blue lights

Motor drive module

Use ULN2003 high-power Darlington chip to drive stepper motor, input voltage: 5V-12V

Stepper motor

The arduous stepping motor with internal gear inheritance, the speed is slower than the ordinary high-speed motor, but the torque is relatively large

Motor fan

DC motor small fan, with XH2.54PIN socket line, need to be used with the motor drive module

Relay module

When the relay coil has no voltage, the relay does not pull in, and the common terminal is connected to the normally closed terminal. When there is voltage, the relay pulls in, and the common terminal is connected to the normally open terminal.

Joystick module

Can output X, Y, Z axis data, so as to control the equipment in all directions

Potentiometer module

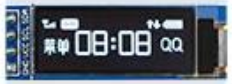
Change the tissue by adjusting the knob to change the current and voltage in the circuit

Steering gear

Change the steering gear rotation angle by PWM control pulse

Raspberry Pi Pico & Sensors

Pico Sensor Kit



OLED display

I2C command control to display the content of custom character string

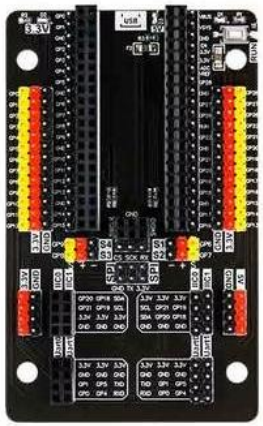
Infrared remote control module

Send infrared control signal

Infrared receiver

It can receive the data from the infrared remote control, and the control commands can be obtained by decoding in the program

Raspberry Pi Pico WH

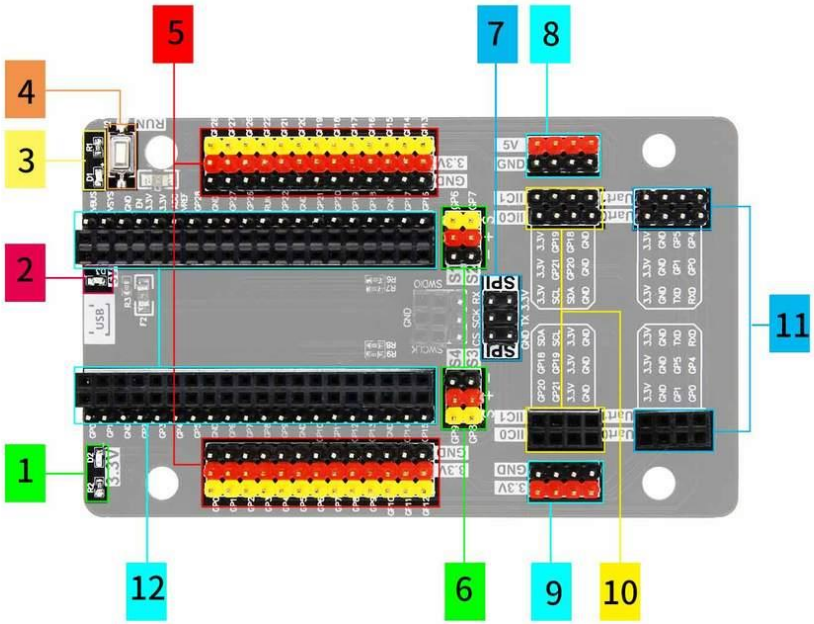


Expansion Board

Raspberry Pi Pico & Sensors

- Pico Sensor Kit (Expansion Board)

Hardware function distribution



1	3.3V indicator	2	5V indicator	3	Power Indicator
4	Reset/RUN button	5	IO interface	6	Servo interface
7	SPI interface	8	5V power interface	9	3.3V power interface
10	IIC interface	11	Serial interface	12	Raspberry Pi PICO pin socket

4 Basic Practices

Basic Practices

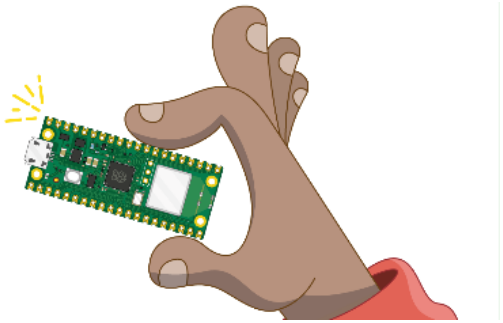
Dev. Environment

- (Step-1) Download the latest firmware version.

  https://projects.raspberrypi.org/en/projects/get-started-pico-w/1

Getting started with your Raspberry Pi Pico W

Raspberry Pi Pico | Python



Set up your Raspberry Pi Pico W

Connect your Raspberry Pi Pico W and set up MicroPython.

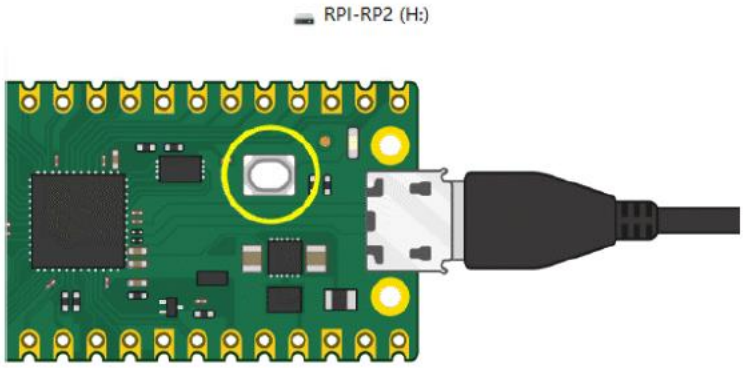
MicroPython is a version of the Python programming language for microcontrollers, such as your Raspberry Pi Pico W. MicroPython lets you use your Python knowledge to write code to interact with electronics components.

Download the latest version of Raspberry Pi Pico W firmware at **https://rpf.io/pico-w-firmware**

Basic Practices

Firmware Update

- (Step-2) Press and hold the button on the Pico board, connect the pico to the USB port of the computer through the Micro USB cable, and then release the button. After connecting, the computer will automatically recognize a removable disk (RPI-RP2).

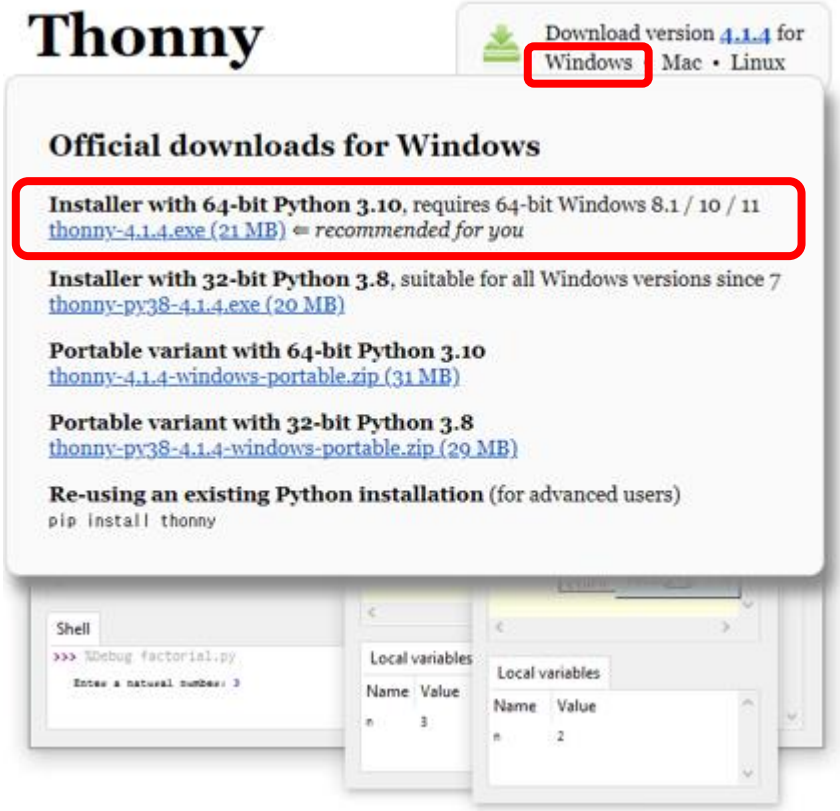


- (Step-3) Copy the latest firmware file to RPI-RP2 disk.
  - ✓ After the copy is completed, the Pico will automatically restart. After the automatic restart, the pico will be recognized as a serial port. At this time, the firmware is successfully flashed.

4 Basic Practices

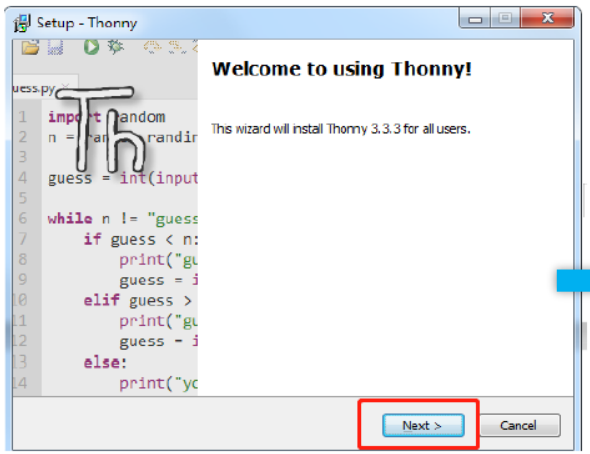
Thonny software download

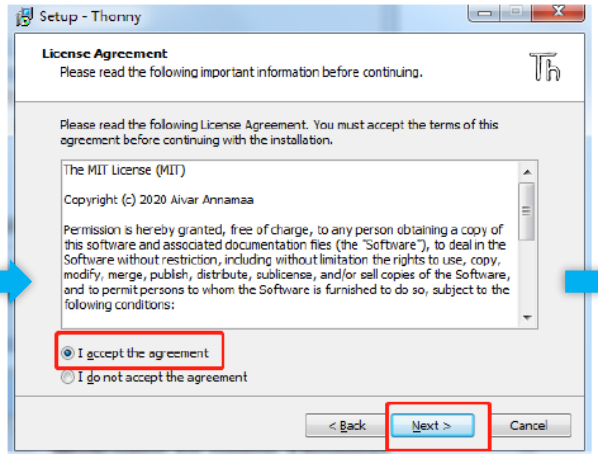
- MicroPython programing software install
- Thonny IDE(Integrated Development Environment, IDE)
- https://thonny.org/

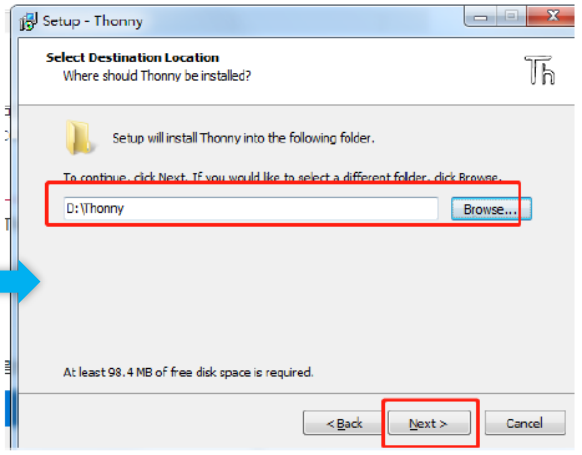


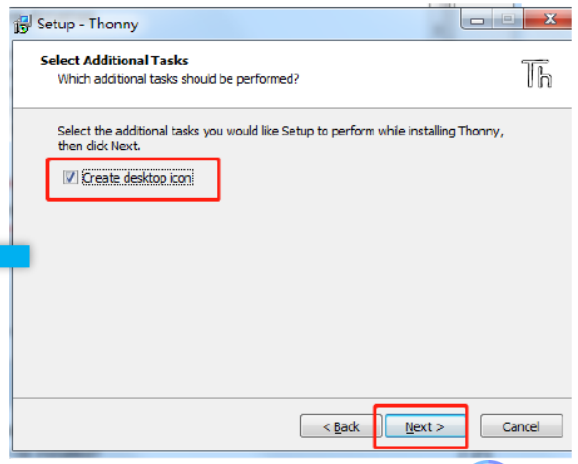
4 Basic Practices

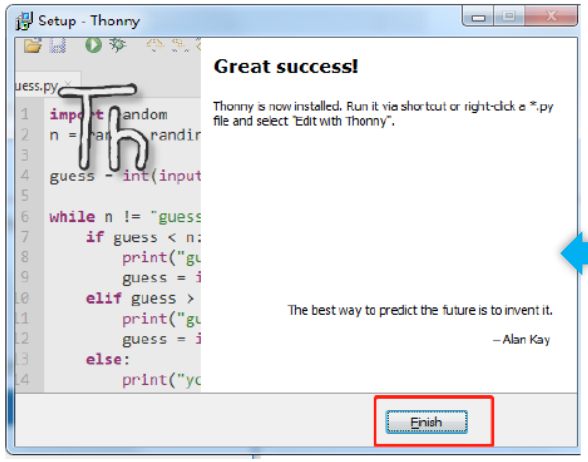
Thonny software install





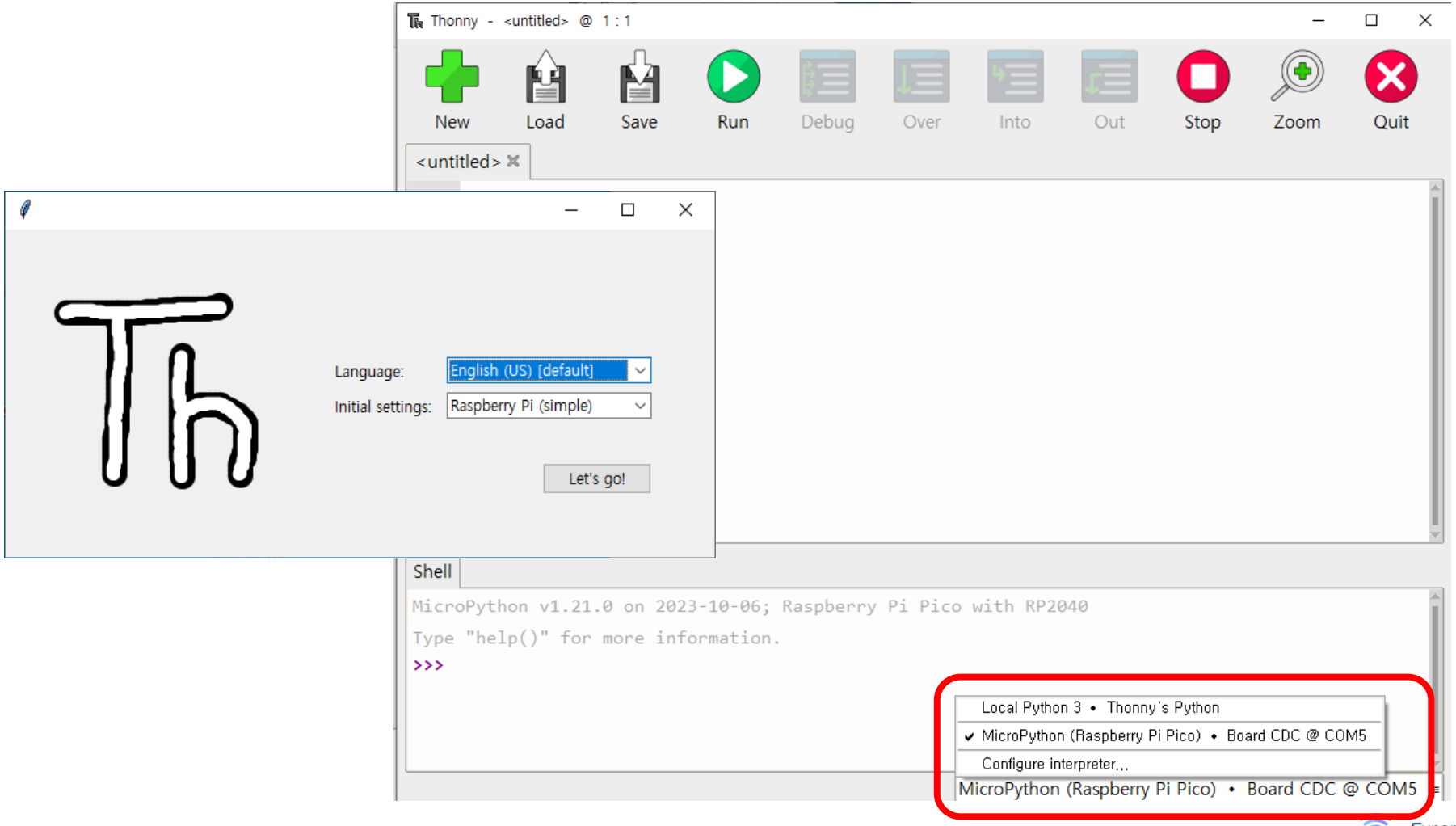






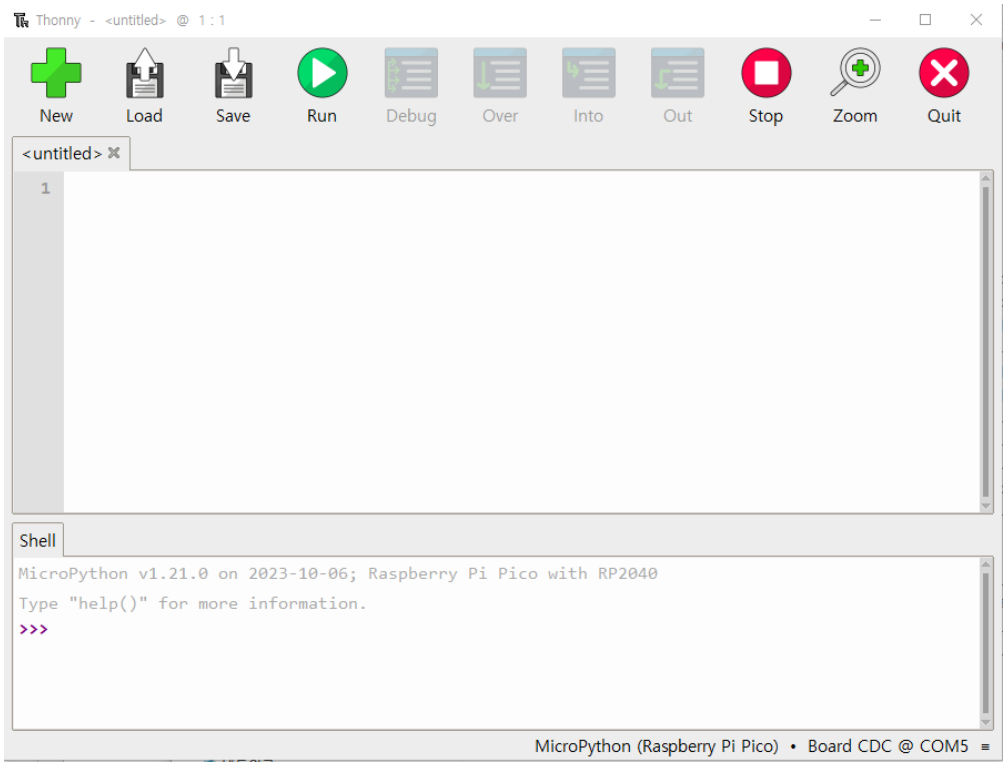
4 Basic Practices

Running Thonny



4 Basic Practices

Try example code – blink LED



Type this codes

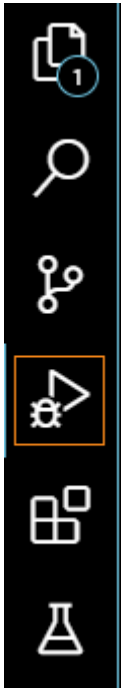
```
import machine
import time
led_onboard = machine.Pin("LED", machine.Pin.OUT)

while True:
    led_onboard.value(1)
    time.sleep(1)
    led_onboard.value(0)
    time.sleep(1)
```

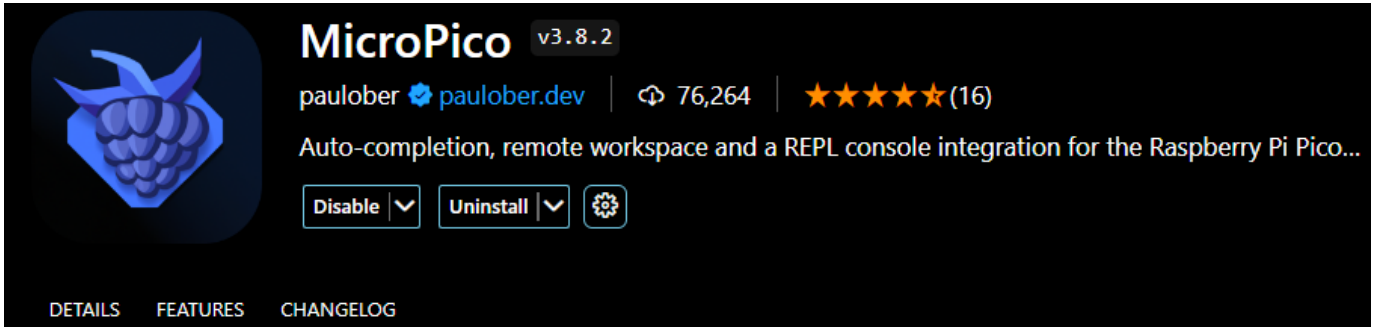
- Type codes or load *.py file
- Press 

button.

4 Basic Practices

Setup MicroPython IDE on VS Code





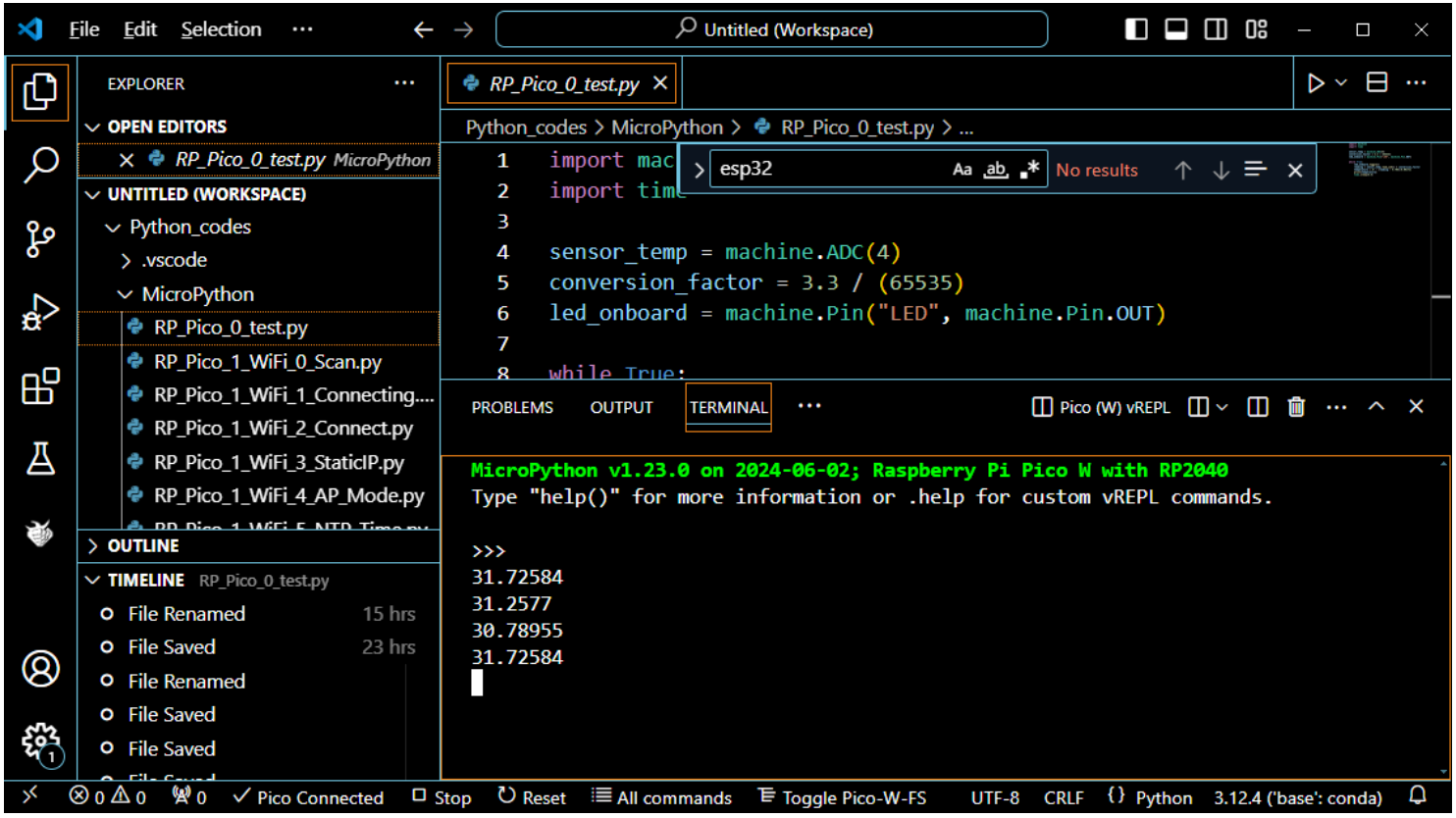
- Install MicroPico
- When installation is complete, you can see the following image at the bottom of the VS code window.

4 Basic Practices

Setup MicroPython IDE on VS Code

Execute MicorPython Code

Exploring Internal Files in Raspberry Pi Pico



IOT Expert Course

4 Basic Practices

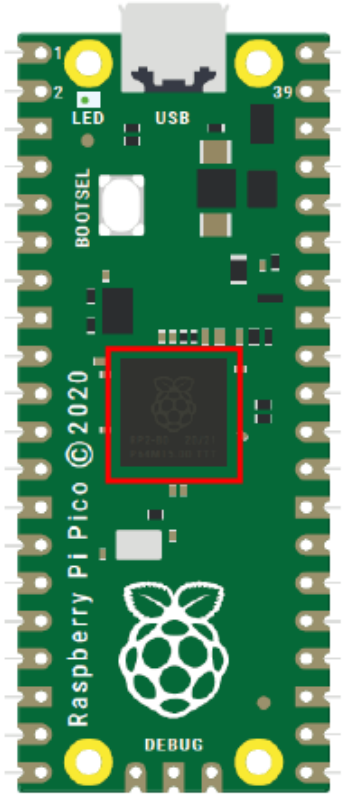
Try example code – 1. blink LED

```
import machine
import time
led_onboard = machine.Pin("LED", machine.Pin.OUT)

while True:
    led_onboard.value(1)
    time.sleep(1)
    led_onboard.value(0)
    time.sleep(1)
```

- https://docs.micropython.org/en/latest/index.html
- Import machine
  - ✓ The machine library contains all the instructions MicroPython needs to communicate with Pico and other MicroPython-compatible devices, extending the language of physical computing.
- Import time
  - ✓ The "time" library. This library handles everything time related, from measuring it to inserting delays into programs. The unit is seconds.

4 Basic Practices

0) Onboard temperature sensor



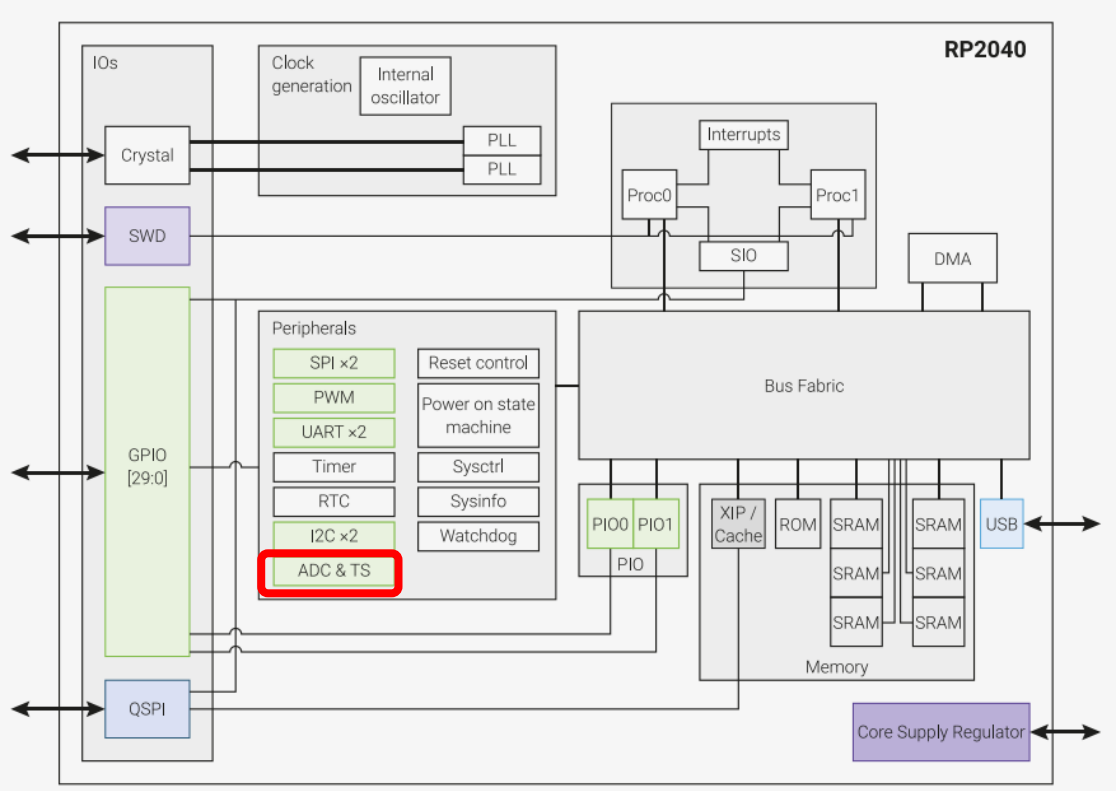
- This course requires no additional hardware and directly utilizes the temperature sensor on the Raspberry Pi Pico board.
  - ✓ The temperature sensor measures the Vbe voltage of a biased bipolar diode, connected to the fifth ADC channel 4.
  - ✓ Typically, Vbe = 0.706V at 27 degrees C, with a slope of -1.721mV per degree.
  - ✓ Therefore, the temperature can be approximated as follows:

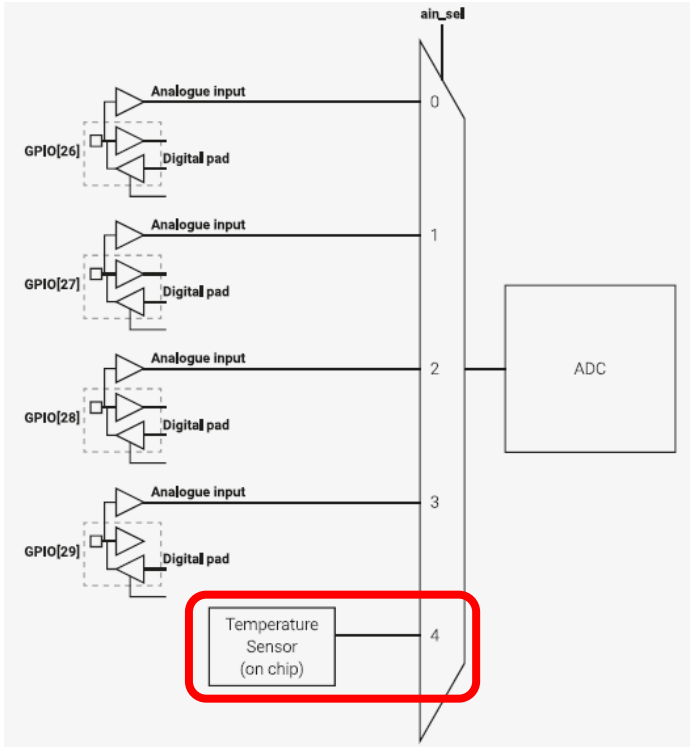
    T = 27 - (ADC_voltage - 0.706)/0.001721
  - ✓ As the Vbe and the Vbe slope can vary over the temperature range, and from device to device, some user calibration may be required if accurate measurements are required.

[Source Code] On board temperature sensor.py

4 Basic Practices

- 0) Onboard temperature sensor

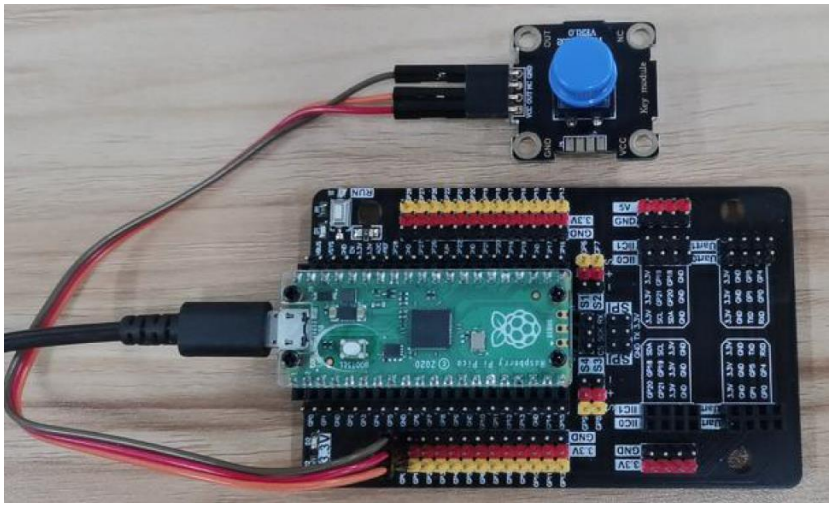




Basic Practices

1) Button Control

- When the button is pressed, the LED is on, and “press” is printed every 100 ms.
  - ✓ The button module needs to pull up the input mode. When the button is pressed, the OUT pin outputs a low level, and when it is released, it outputs a high level.



Button module

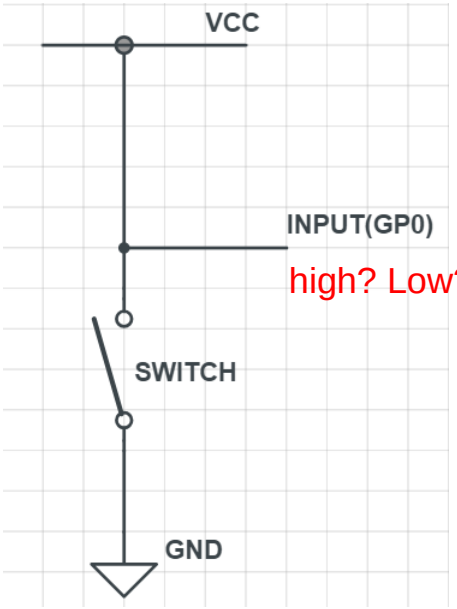
Used for key control,
low-level effective

Button module	Pico sensor expansion board
IN	GP0
VCC	3.3V
GND	GND

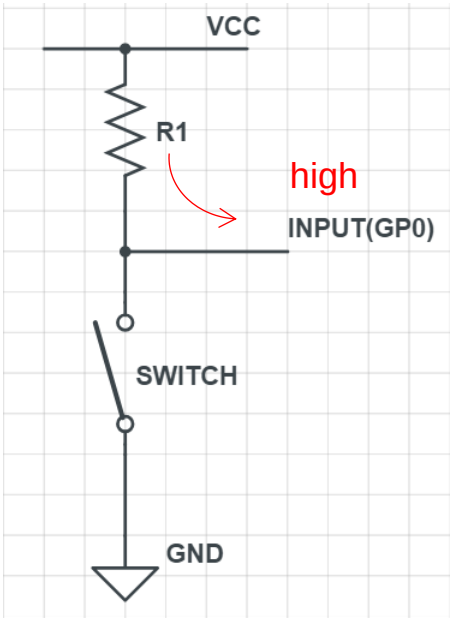
4 Basic Practices

1) Button Control

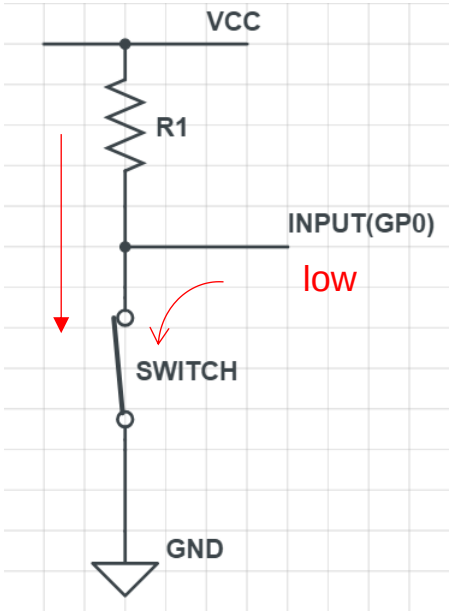
floating



high



low

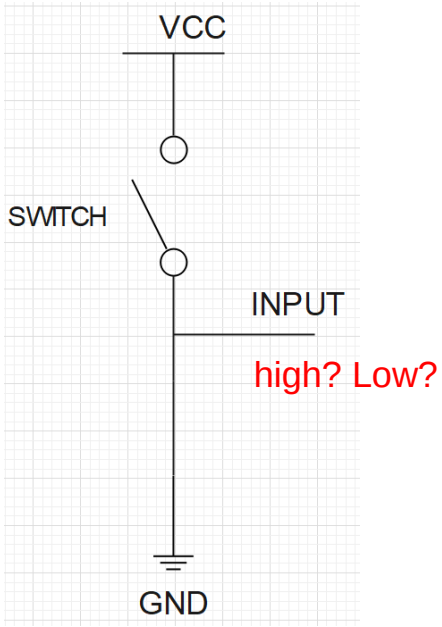


pull-up

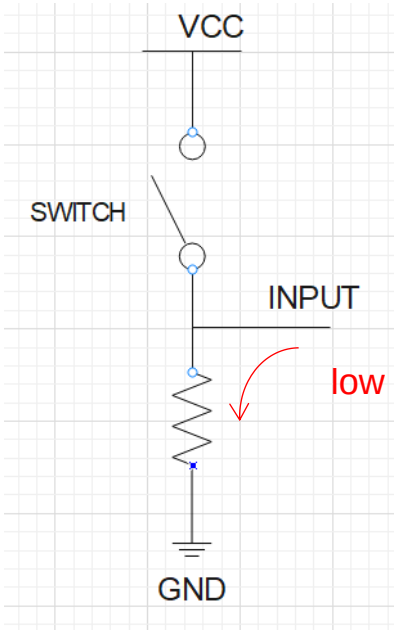
Basic Practices

1) Button Control

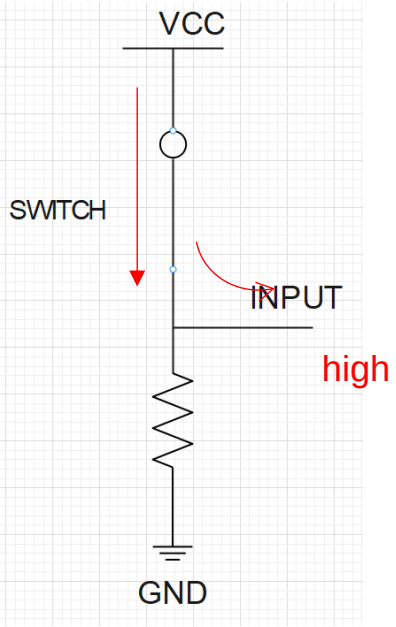
floating



low



high

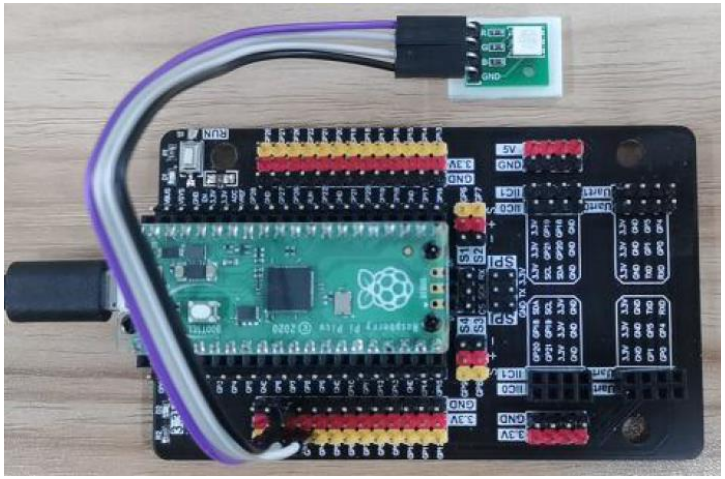


pull-down

Basic Practices

2) RGB three-color light

- Switch a color every 0.5s
  - ✓ The RGB three-color light module is composed of three LEDs: red, green, and blue. When we provide high-level voltage to the R, G, and B pins, it will light up the corresponding color, and it is low-level voltage, RGB light will turn off.



RGB three-color light

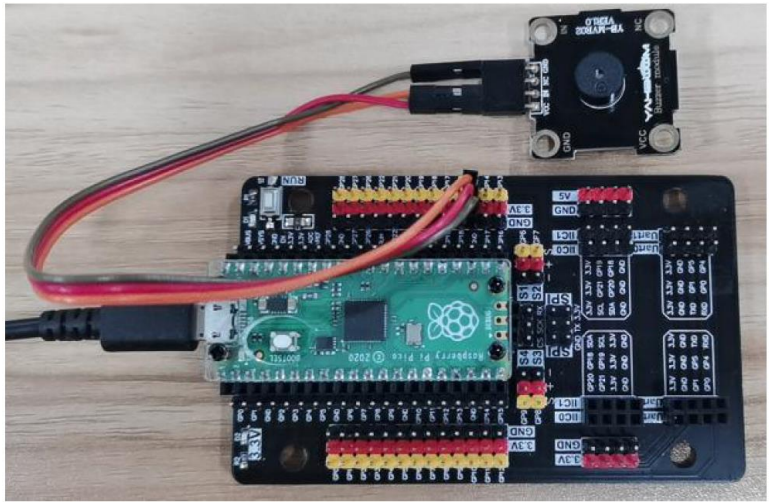
Light up three colors of red, green, and blue lights

RGB light module	Pico sensor expansion board
R	GP1
G	GP2
B	GP3
GND	GND

Basic Practices

3) Passive buzzer

- Simulate two different frequencies (500Hz, 250Hz)
  - ✓ The buzzer is divided into active buzzer and passive buzzer.
  - ✓ The active buzzer has an internal oscillation source, which can only achieve a fixed frequency sound.
  - ✓ The passive buzzer does not have an internal oscillation source, so if it is unable to make a sound with a DC signal, it needs to be driven by a square wave. The sound frequency is controllable, and square waves with different frequencies can achieve different frequencies of sound, which can realize playing music and so on.



Buzzer module

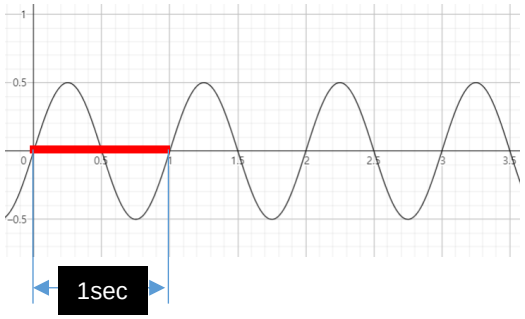
Receive PWM signals of different frequencies and play sounds of different tones

Passive buzzer	Pico sensor expansion board
IN	GP15
VCC	5V
GND	GND

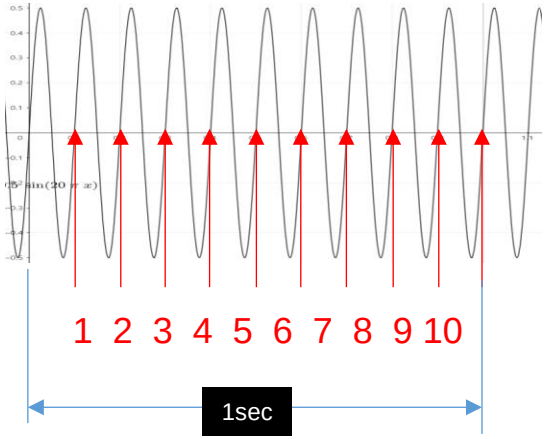
Basic Practices

3) Passive buzzer

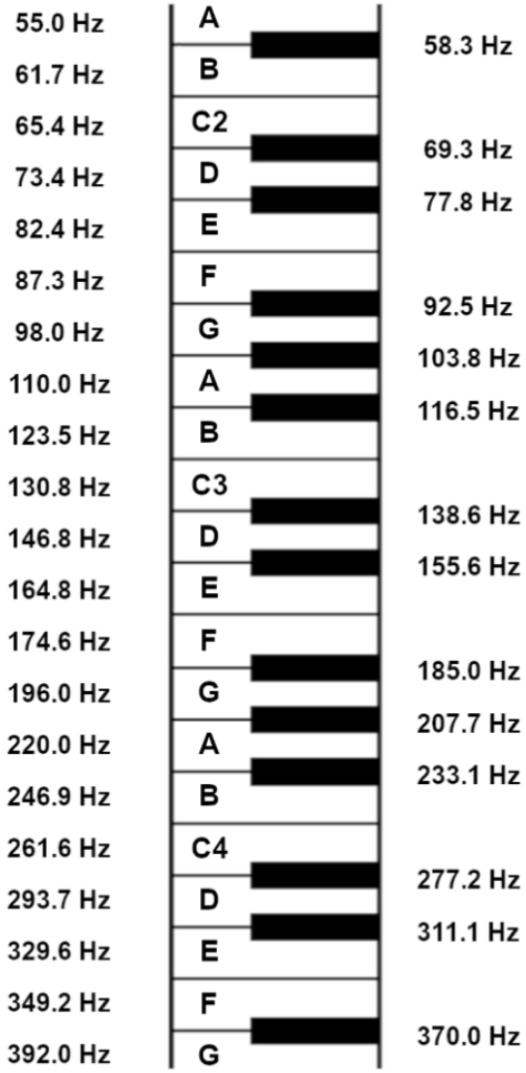
1hz =



10hz =



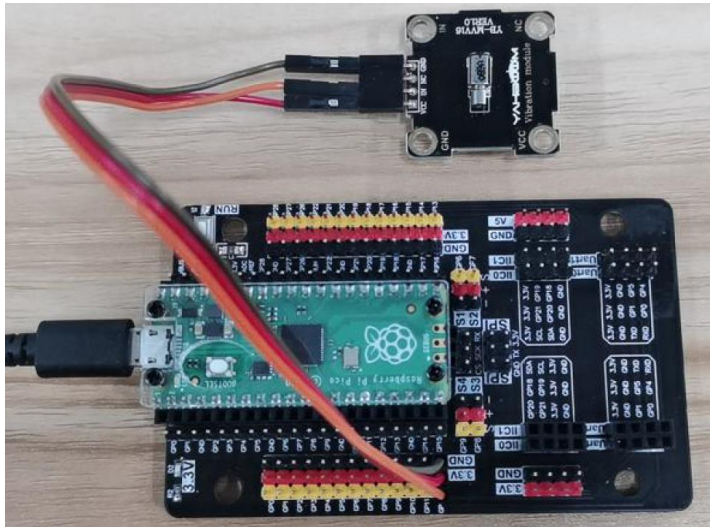
$$f = \frac{1}{T}$$



Basic Practices

4) Vibration reminder

- Make shake function, it has two arguments. One is vibration time; another is delaying time.
  - ✓ When we provide high-level to IN pin, motor will rotate.



Vibration module

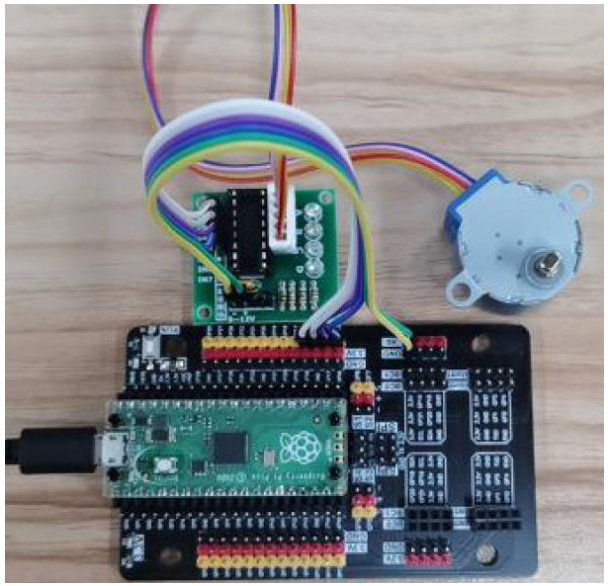
Contains a rotor motor, which drives the rotor to generate centrifugal force to cause vibration through the rotation of the motor

Passive buzzer	Pico sensor expansion board
IN	GP12
VCC	5V
GND	GND

Basic Practices

5) Drive Stepper motor

- Stepper motor first rotates clockwise, then counterclockwise, and keep the cycle



Stepper motor

The arduous stepping motor with internal gear inheritance, the speed is slower than the ordinary high-speed motor, but the torque is relatively large

Motor drive module

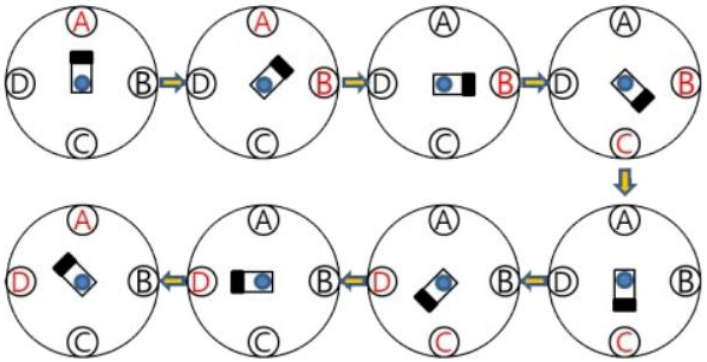
Use ULN2003 high-power Darlington chip to drive stepper motor, input voltage: 5V-12V

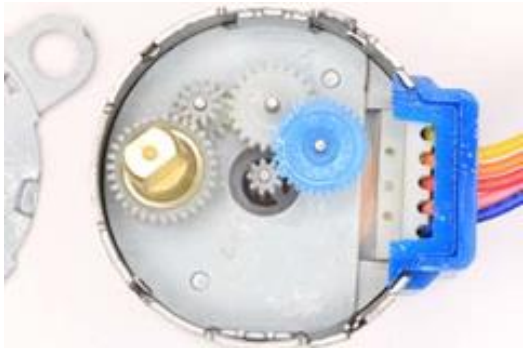
ULN2003 board	Pico sensor expansion board
IN1	GP16
IN2	GP15
IN3	GP14
IN4	GP13
Negative electrode(-)	GND
Positive electrode(-)	5V

Basic Practices

5) Drive Stepper motor

Step motor working method





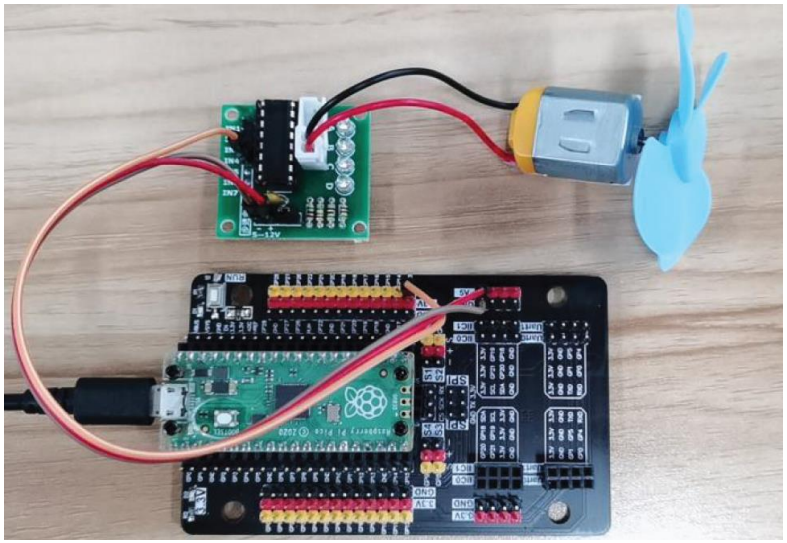
adjusted by gear ratio

	1	2	3	4	5	6	7	8
A	1	1						
B		1	1	1				
C				1	1	1		
D						1	1	1

Basic Practices

6) Fan motor

- Small motor fan starts to rotate at a speed of 50%.
  - ✓ The small motor fan needs a relatively large current, so the ULN2003 drive module is needed to drive the small motor fan, and the speed of the small motor fan can be controlled by PWM.



Motor fan

DC motor small fan, with XH2.54PIN socket line, need to be used with the motor drive module

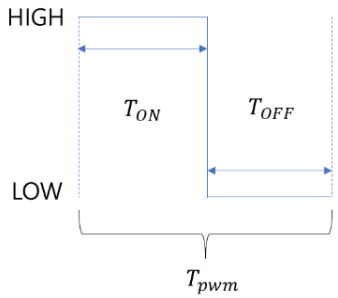
Motor drive module

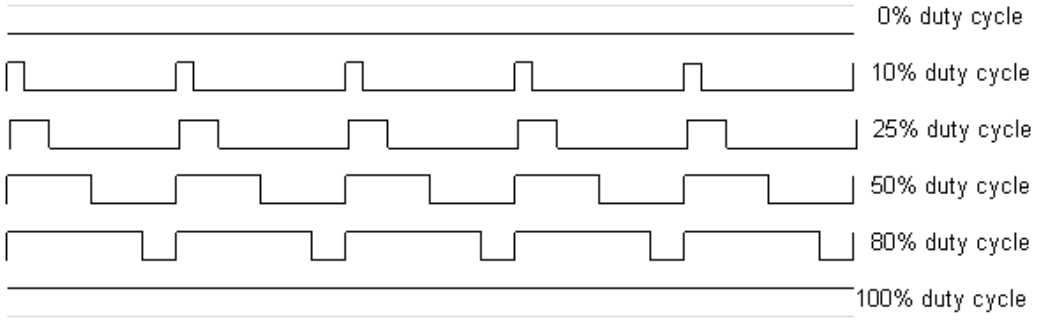
Use ULN2003 high-power Darlington chip to drive stepper motor, input voltage: 5V-12V

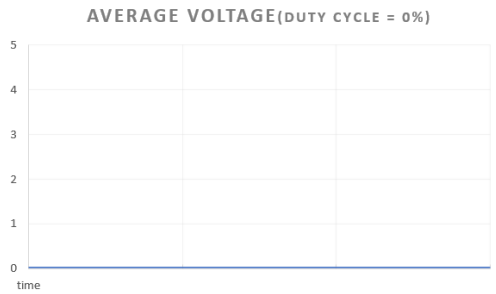
ULN2003 board	Pico sensor expansion board
IN1	GP16
IN2	GP15
IN3	GP14
IN4	GP13
Negative electrode(-)	GND
Positive electrode(-)	5V

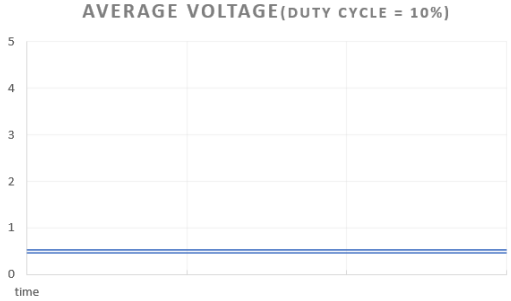
Basic Practices

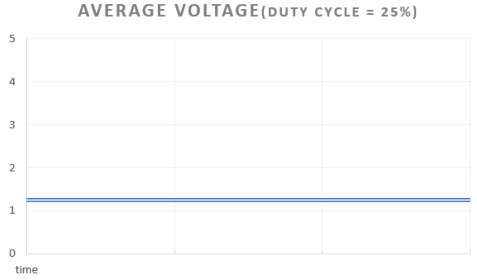
6) Fan motor

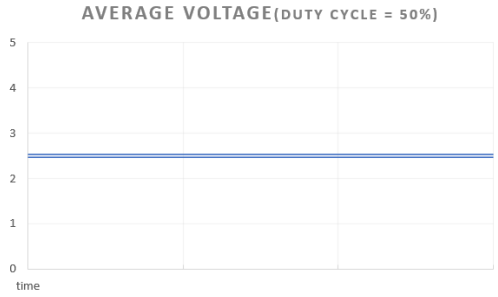


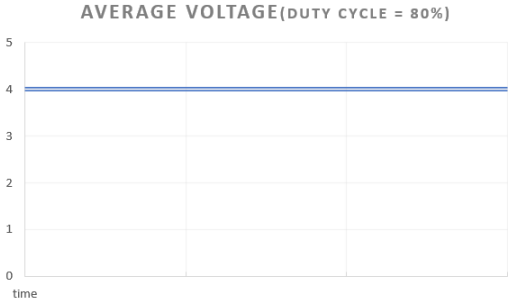


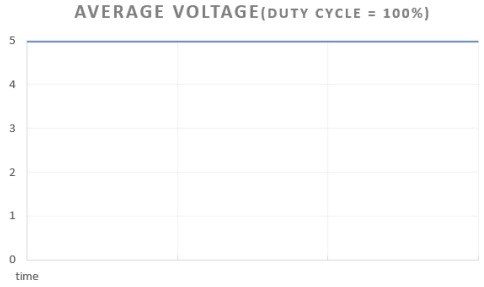








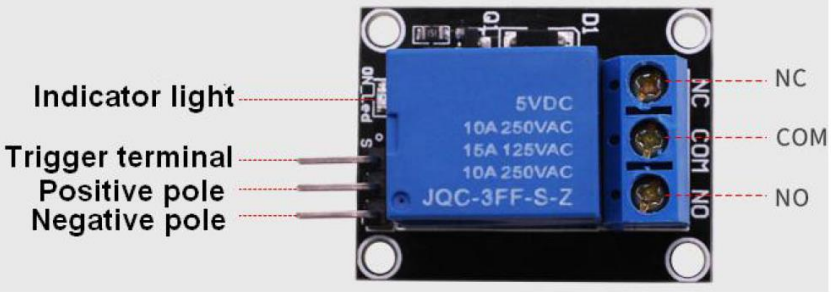




Basic Practices

7) Relay switch

- When the program is running, the relay will switch on and off cyclically, turning on for one second and turning off for another second.

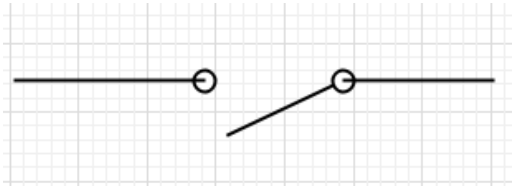


ULN2003 board	Pico sensor expansion board
S	GP4
+	3.3V
GND	GND

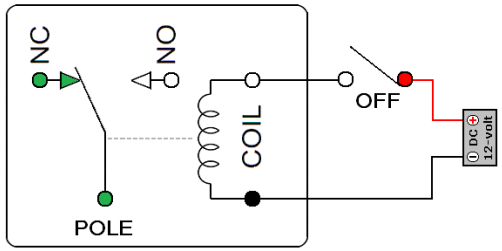
Basic Practices

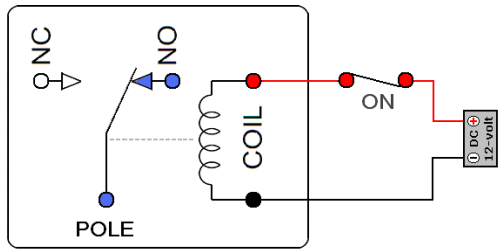
7) Relay switch

switches



relay switch

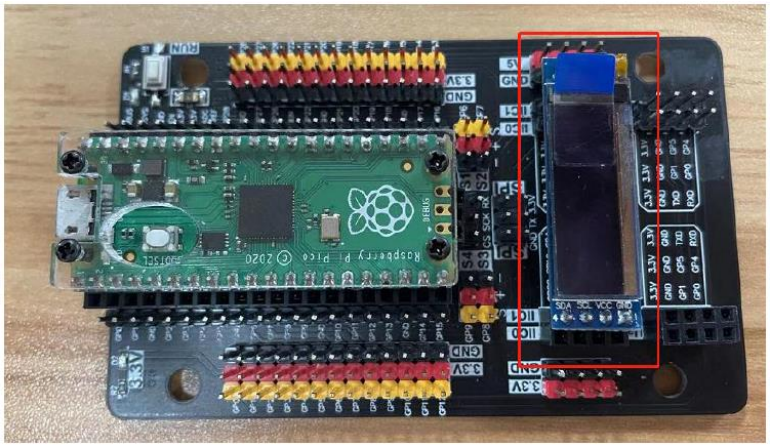


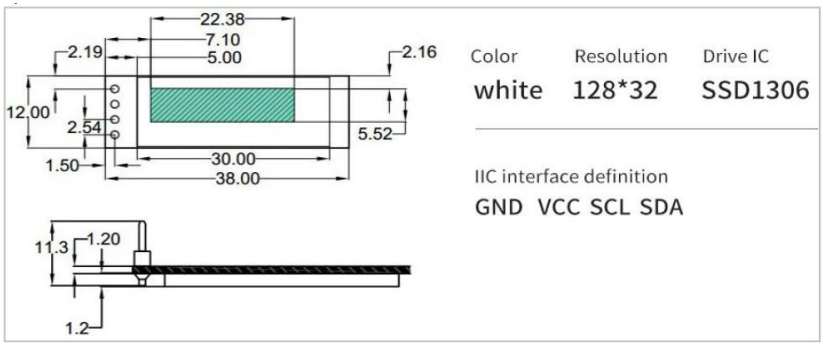


Basic Practices

8) OLED display characters

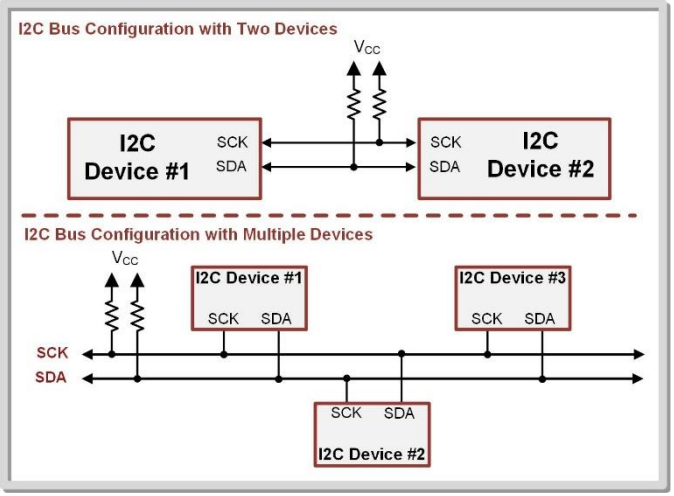
- When the program is running, OLED will display “Hello Guys!”.



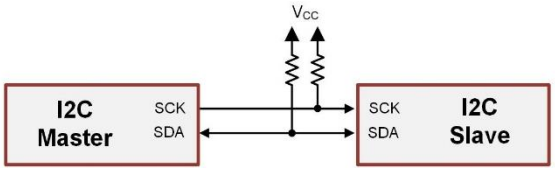


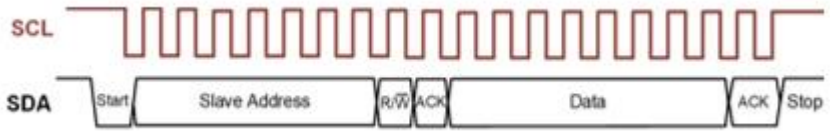
4 Basic Practices

8) OLED display characters



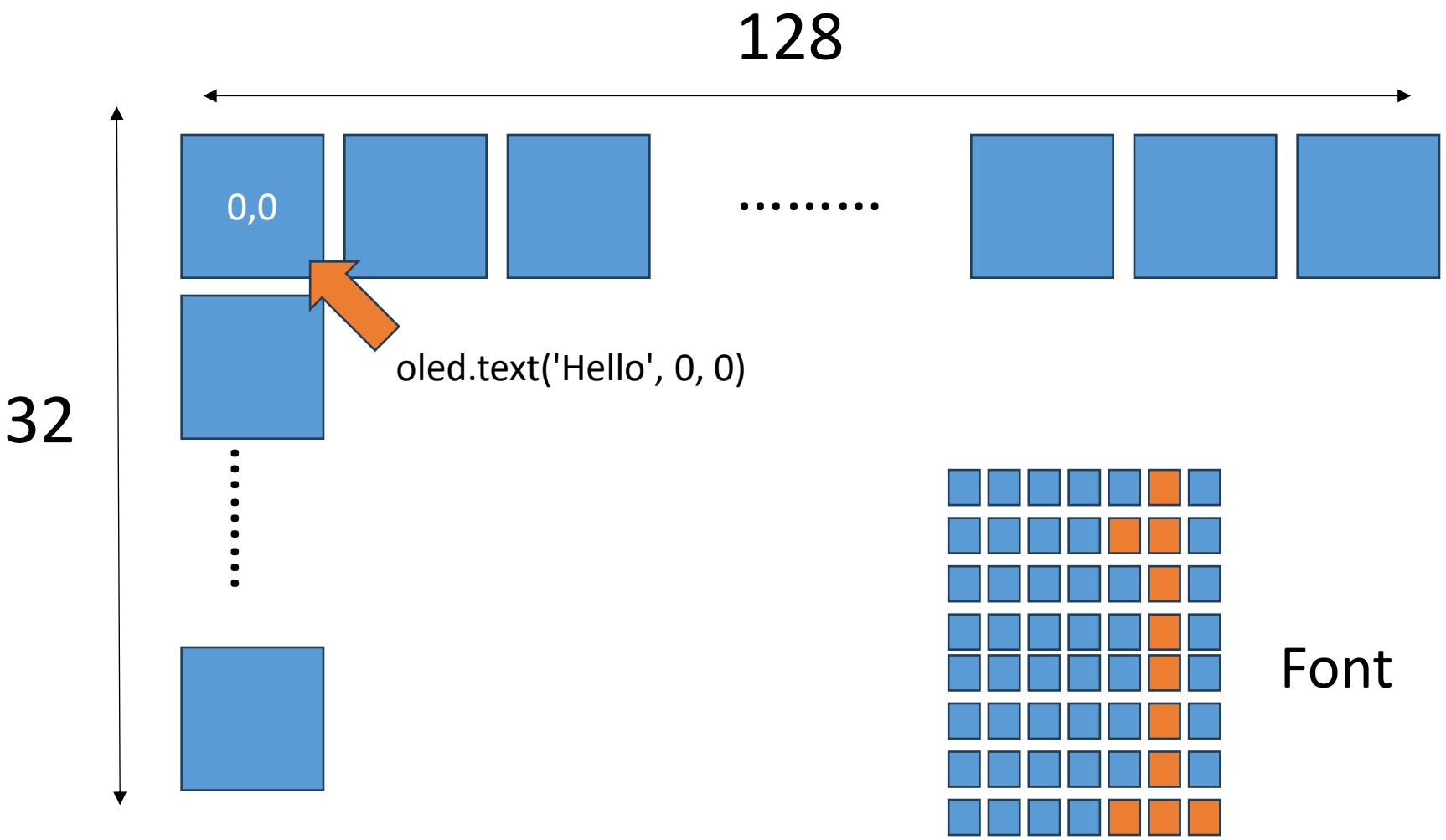
- OLED screen we use uses I2C communication
  - ✓ The Inter-Integrated Circuit (I2C) standard is a serial interface implement with a two-wire link that can support multiple masters and multiple slaves.
  - ✓ An I2C bus contains a clock line (SCK) and data line (SDA).
  - ✓ Master – the device that initiates communication and controls the clock.
  - ✓ Slave – a device on the bus that is read or written to but does not initiate transmission or provide a clock.





4 Basic Practices

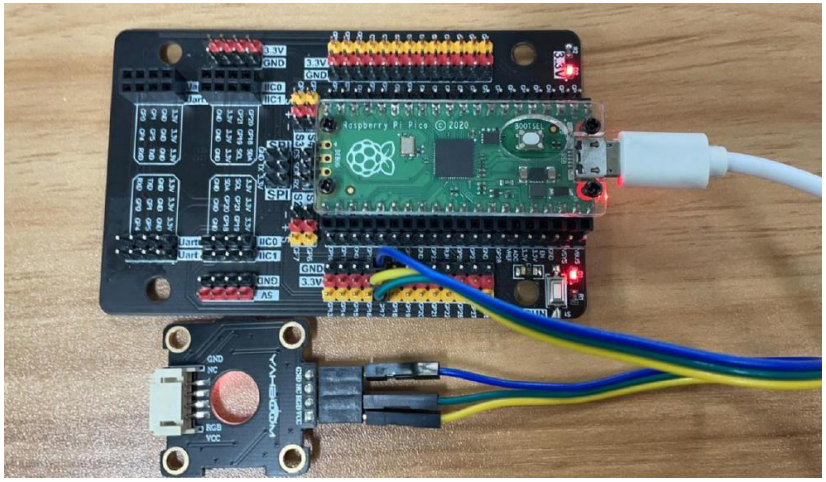
- 8) OLED display characters



Basic Practices

9) Marquee

- When the program is running, all RGB lights on the module will be turned on to achieve the effect of marquee.



RGB light ring module

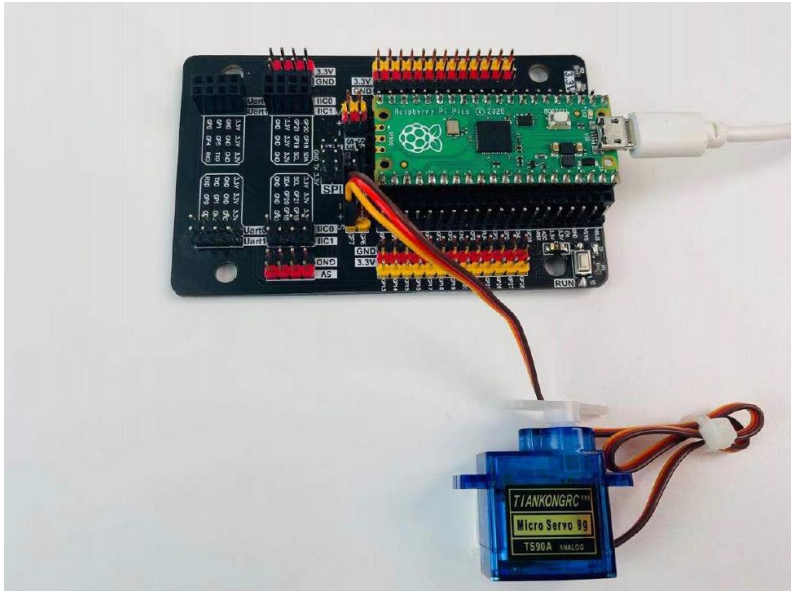
Eight programmable RGB lights form a ring, programmable RGB special effects

RGB halo module	Pico sensor expansion board
RGB	GP17
GND	GND
VCC	3.3V

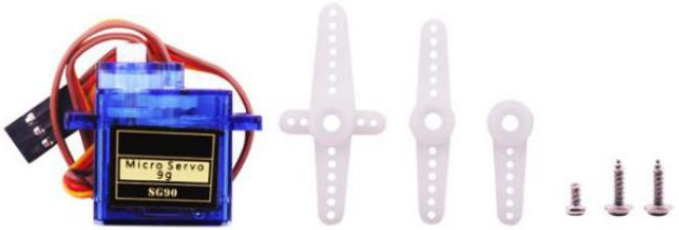
Basic Practices

10) PWM control Servo

- When the program is running, the servo will rotate 0˚ → 180˚ → 0˚ → 180˚. And keep in loop with this status.



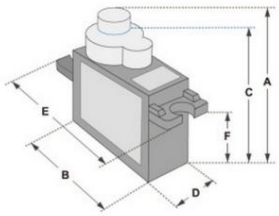
0.5ms----------0°
1.0ms----------45°
1.5ms----------90°
2.0ms----------135°
2.5ms----------180°



Servo	Pico sensor expansion board
Yellow line	GP7
Brown line	GND
VCC line	3.3V

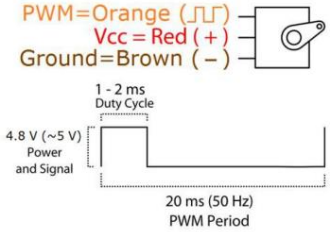
Basic Practices

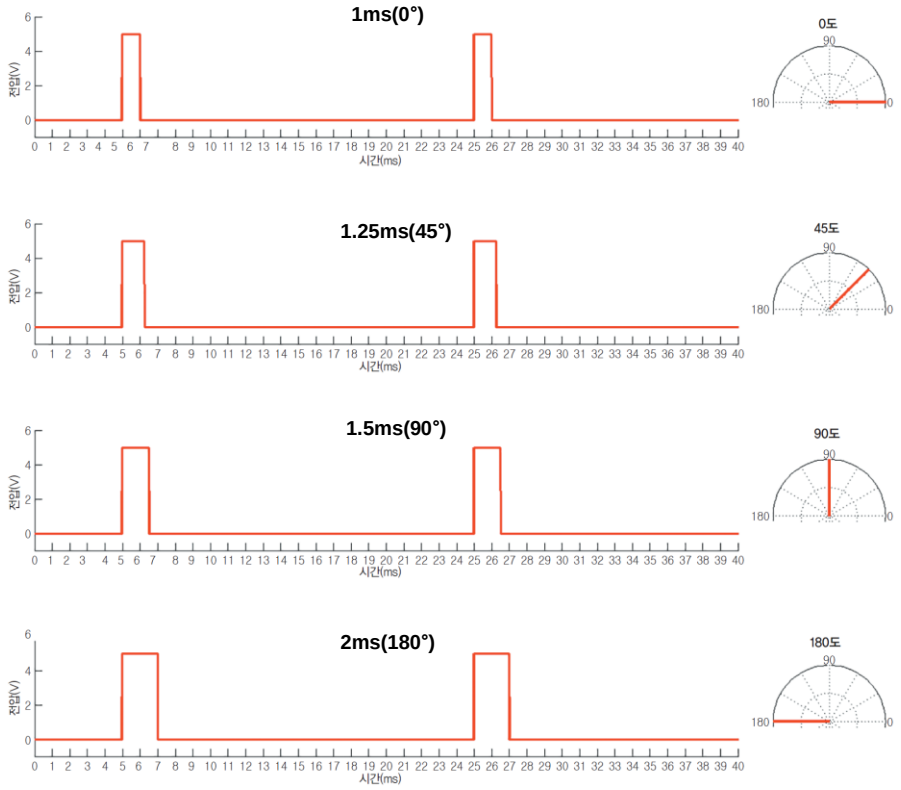
10) PWM control Servo



Dimensions & Specifications
A (mm) : 32
B (mm) : 23
C (mm) : 28.5
D (mm) : 12
E (mm) : 32
F (mm) : 19.5
Speed (sec) : 0.1
Torque (kg-cm) : 2.5
Weight (g) : 14.7
Voltage : 4.8 - 6

Position "0" (1.5 ms pulse) is middle, "90" (~2ms pulse) is middle, is all the way to the right, "-90" (~1ms pulse) is all the way to the left.

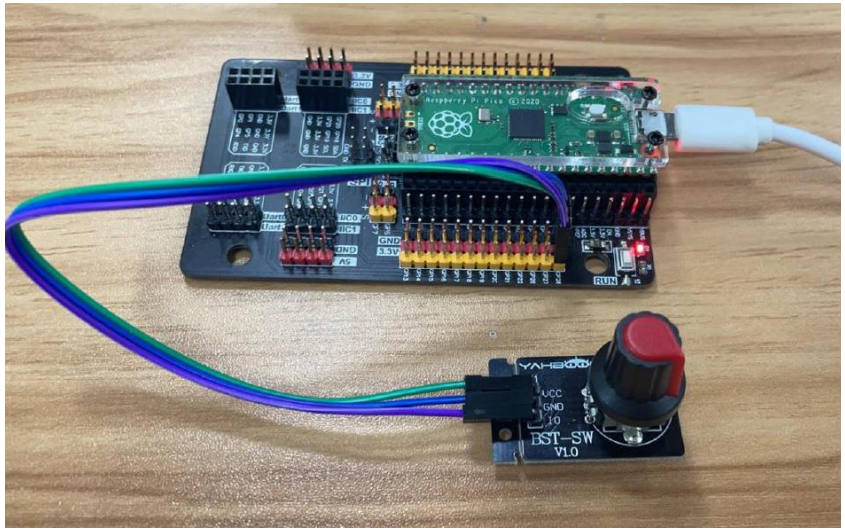




Basic Practices

11) Potentiometer output analog value

- When the program is running, the shell window will print the current analog value of the potentiometer and adjust the knob to see that the analog value becomes larger or smaller.

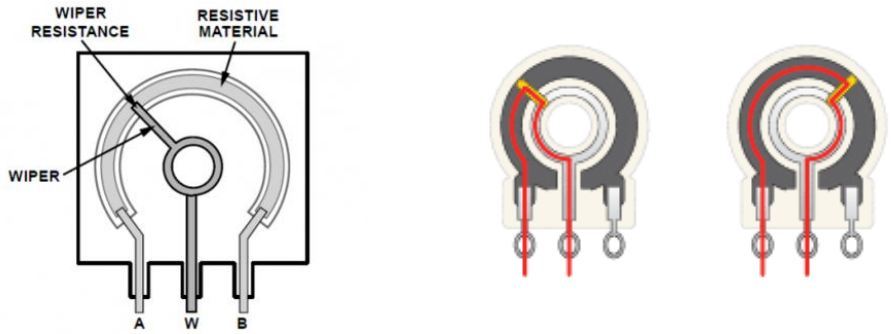


Potentiometer module	Pico sensor expansion board
OUT	GP28
GND	GND
VCC	3.3V

Basic Practices

11) Potentiometer output analog value

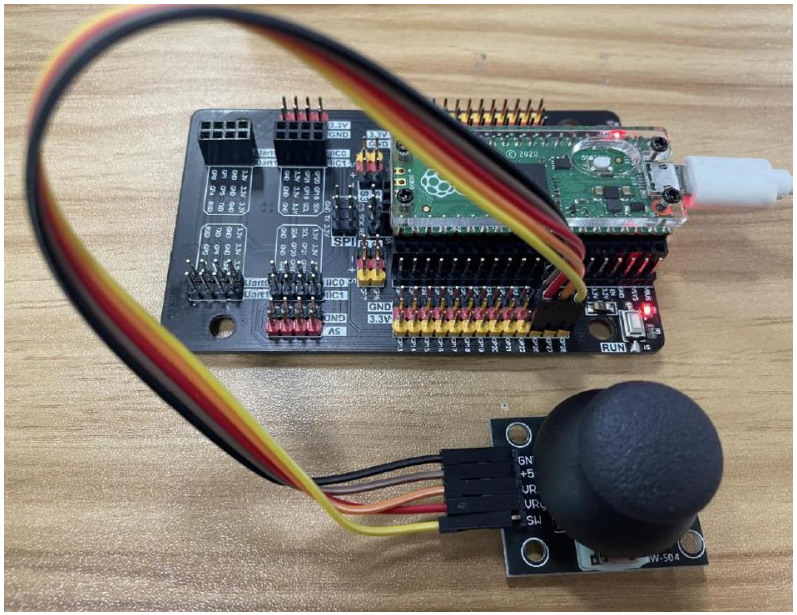
- A resistor whose resistance value can be arbitrarily changed in an electronic circuit.
- In other words, a resistance that can be adjusted with an arbitrary resistance value, not a fixed resistance.
- When the resistance is changed using a variable resistor, the magnitude of the current also changes.
- It is structured so that the resistance value can be adjusted by adjusting the length of the resistance component composed of carbon band inside with a wiper.

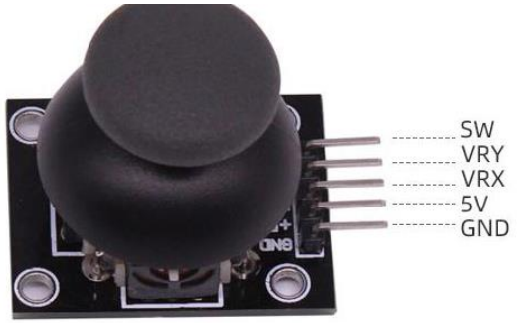


Basic Practices

12) Rocker print data

- When the program is running, the shell window will print the current analog value of the rocker.



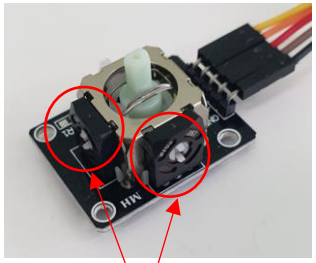


Rocker module	Pico sensor expansion board
SW	GP28
VRX	GP27
VRY	GP26
GND	GND
+5V	3.3V

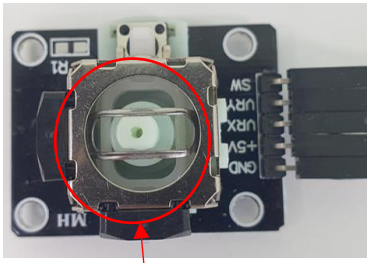
Basic Practices

12) Rocker print data

- using 2 variable resistors.(potentiometer)
- The value changes according to the change of the x-axis resistance value
  - ✓ When moving to the left, the smaller the X value of the output. When moving to the right, the larger the X value of the output.
- Change the value according to the change of the y-axis resistance value.
  - ✓ When moving to the up, the larger the Y value of the output. When moving to the down, the smaller the Y value of the output.
- This module can be regarded as a potentiometer with two channels, which can output the analog values of the X and Y axes.
- The output analog range of the X and Y axes is from 0 to 1023



potentiometer

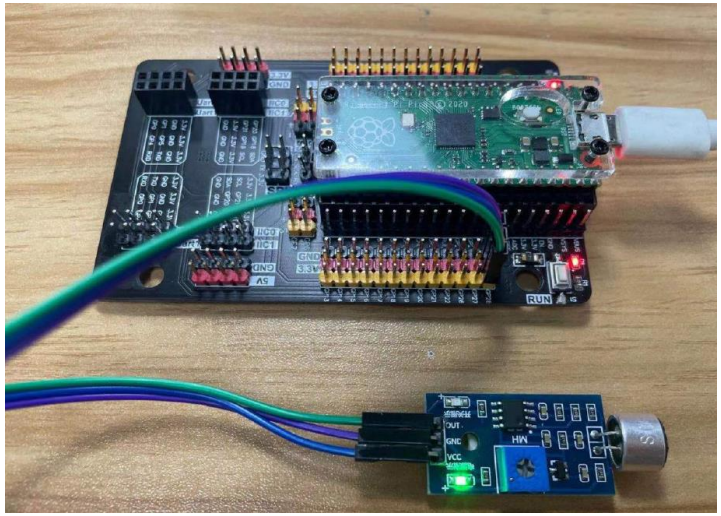


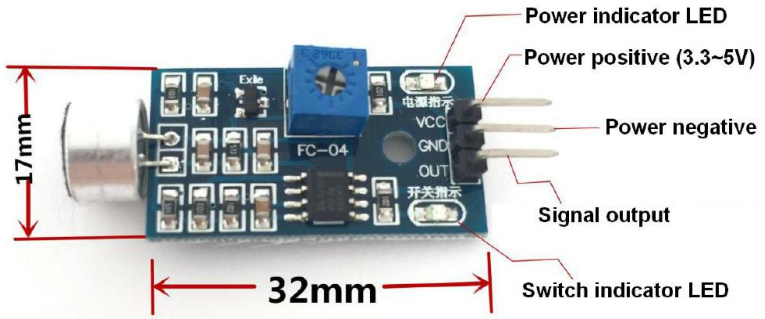
controller

Basic Practices

13) Sound control LED

- When the program is running, when the ambient sound exceeds the threshold, the indicator light on the Pico board will light on.



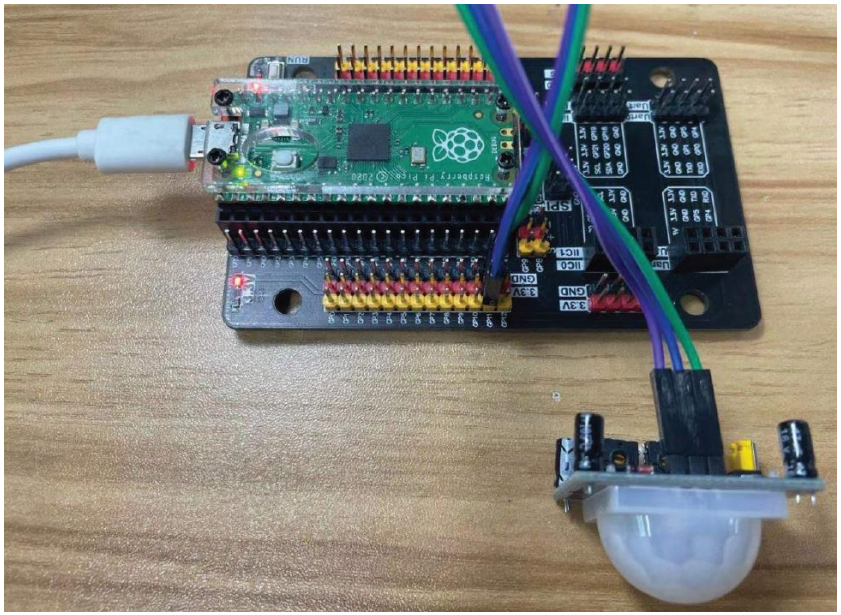


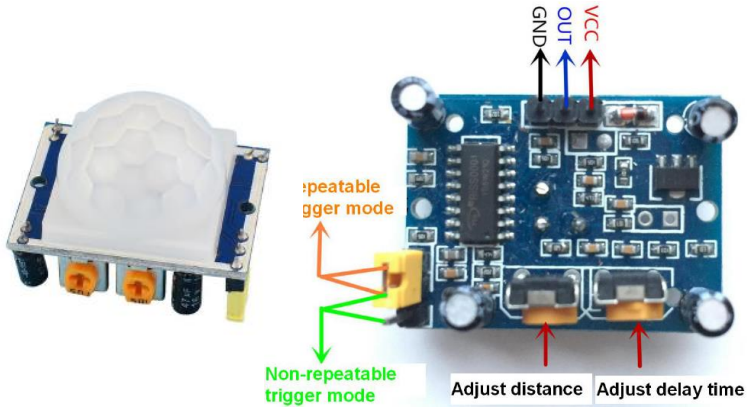
Sound module	Pico sensor expansion board
VCC	GP28
GND	GND
OUT	GP28

Basic Practices

14) Human infrared detector

- When the program is running, if the sensor detects that a person enters the sensing range, the indicator light on the Pico board will light on.



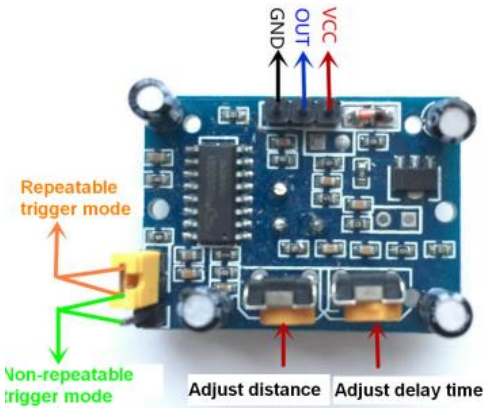


Human infrared module	Pico sensor expansion board
OUT	GP11
GND	GND
VCC	3.3V

Basic Practices

14) Human infrared detector

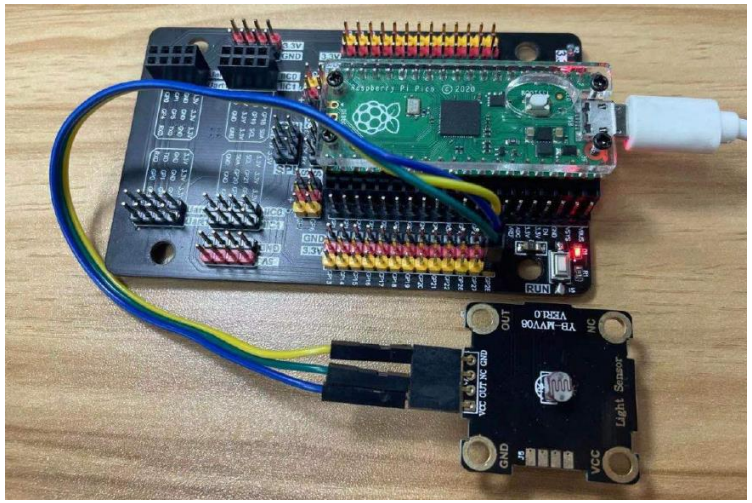
- Infrared sensor module, which works by detecting the infrared rays of about 10 um emitted by the human body.
- A Fresnel lens filter is attached to the surface to reduce the interference of the external environment on the detection.
- Once a person enters the detection area, the infrared radiation of the human body is focused by a part of the mirror, and an alarm signal is generated after processing, and the pin outputs a high level, otherwise it outputs a low level.
- After the module is triggered, it will generate a high level. After a period of delay, the output signal will automatically change from high level to low level.
  - ✓ Adjust the distance potentiometer to rotate clockwise, the sensing distance will increase, otherwise, the sensing distance will decrease.
  - ✓ Adjust the delay potentiometer to rotate clockwise, the induction delay will be longer, otherwise, the induction delay will be shortened.



Basic Practices

15) Detect light intensity

- When the program is running, the shell window will print the current analog value of the photosensitive sensor. The stronger the light intensity, the smaller the read analog value.



Photosensitive module

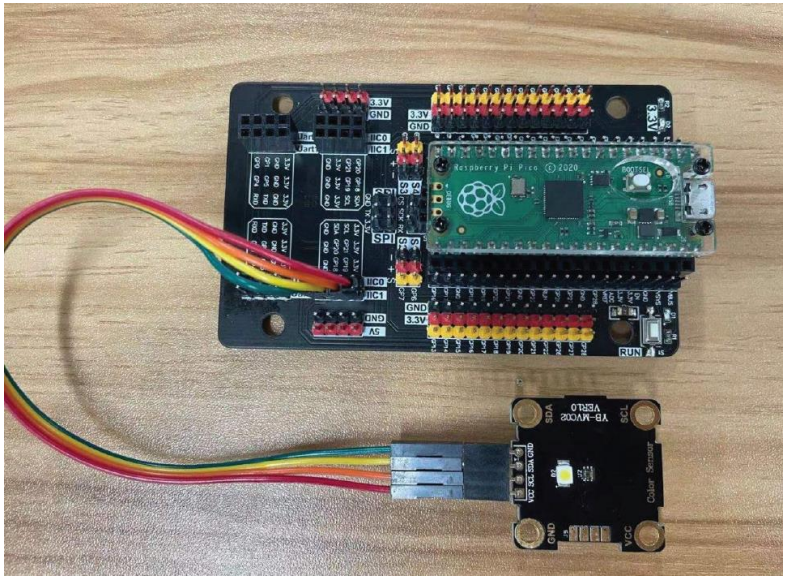
Sense the light intensity of the current environment, and make automatic switches based on this feature

Human infrared module	Pico sensor expansion board
OUT	GP28
GND	GND
VCC	3.3V

Basic Practices

16) Identify color

- When the program is running, the shell window will print the RGB value of the current environment.



Color recognition module

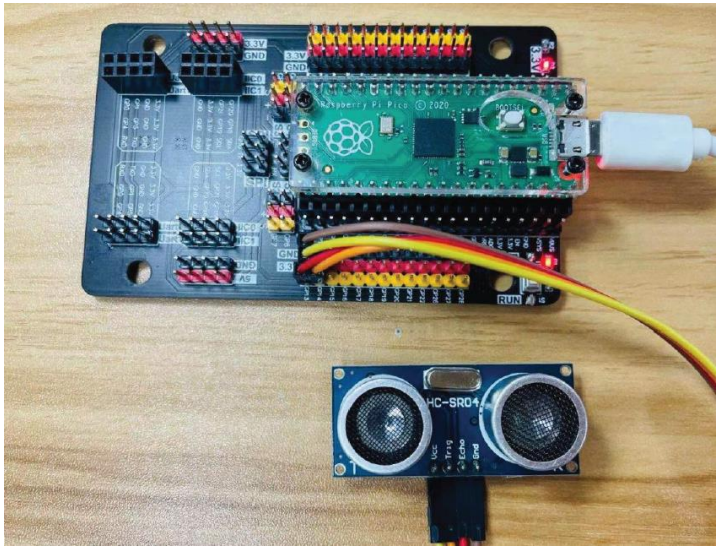
It can identify the RGB values of different colors to determine the color

Color recognition module	Pico sensor expansion board
OUT	GP28
GND	GND
SCL	GP19
SDA	GP18

Basic Practices

17) Ultrasonic ranging

- When the program is running, the shell window will print the distance value of the current environment.



Ultrasonic module

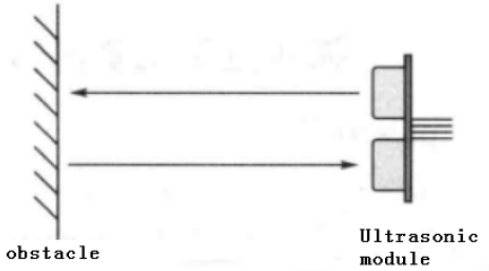
Professional ultrasonic module, high precision, small blind area, fast feedback speed, very high market penetration rate

Ultrasonic module	Pico sensor expansion board
Trig	GP13
Echo	GP14
VCC	5V
GND	GND

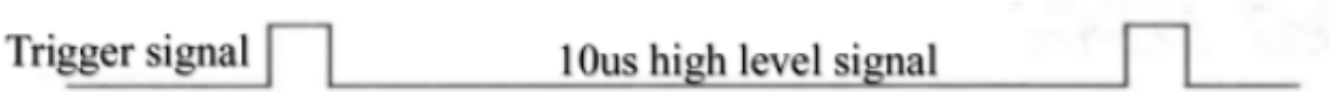
Basic Practices

17) Ultrasonic ranging

- The ultrasonic module has two ultrasonic probes for transmitting and receiving ultrasonic waves. (The range of measurement is 3-450 cm)



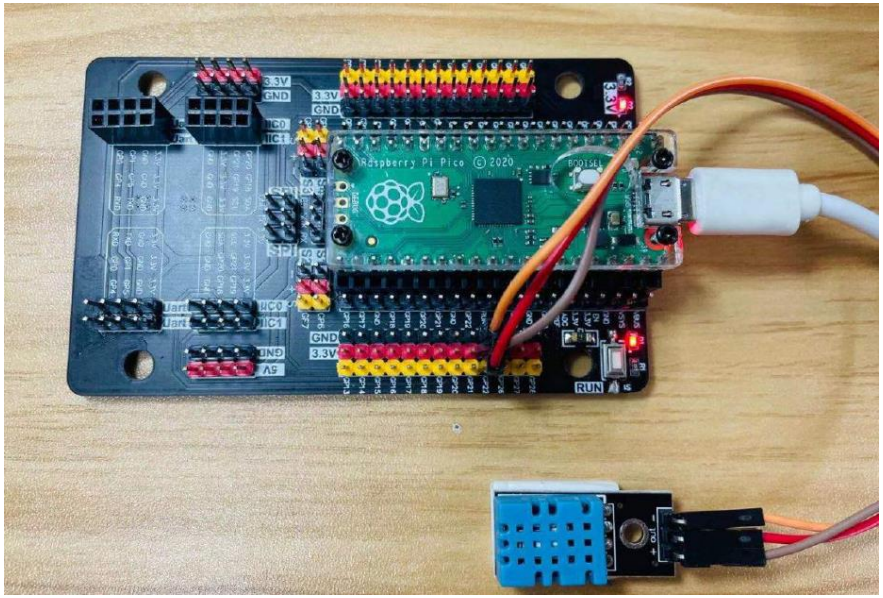
- ✓ (1) You need to input a high-level signal of at least 10us to the Trig pin to trigger the ranging function of the ultrasonic module.
- ✓ (2) After the ranging function is triggered, the module will automatically send out 8 ultrasonic pulses with 40 kHz and automatically detect whether there is a signal return.
- ✓ (3) When the module detects an echo signal, the ECHO pin will output a high level. The high-level duration is the time from when the ultrasonic wave is sent to when it returns. You can calculate the distance by using the time function to calculate the high-level duration.

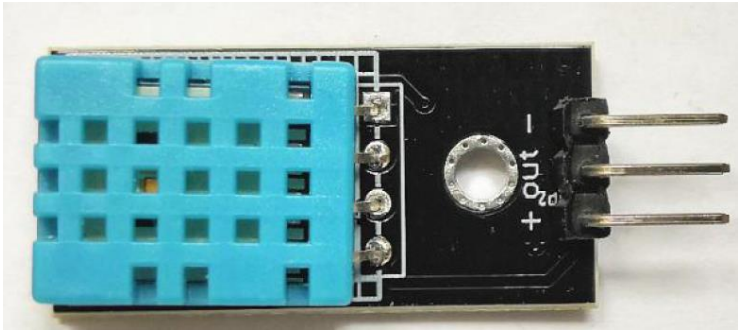


Basic Practices

18) Detect temperature humidity

- When the program is running, the shell window will print the temperature and humidity value of the current environment.

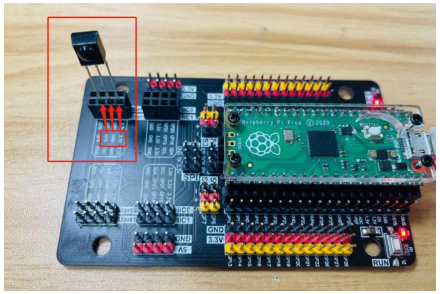


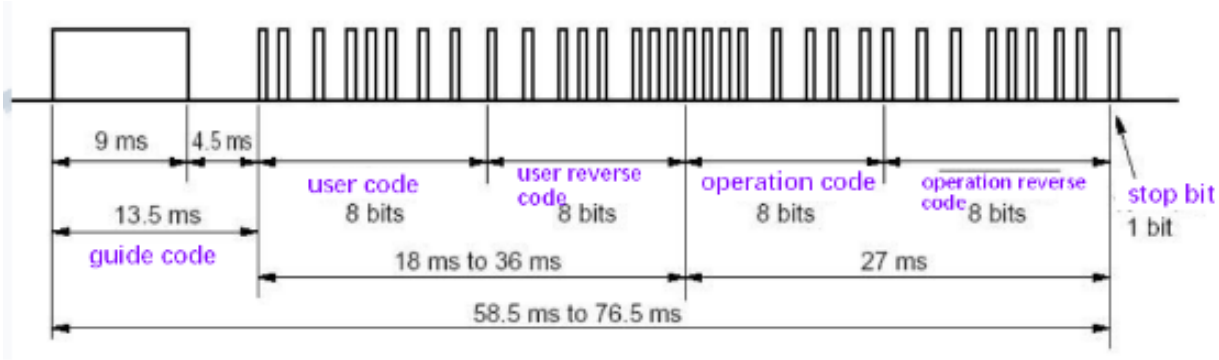


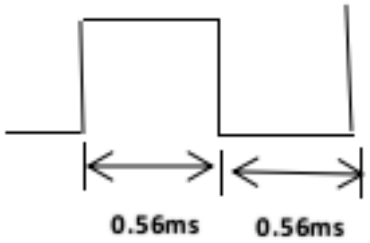
DHT11 module	Pico sensor expansion board
OUT	GP22
VCC	3.3V
GND	GND

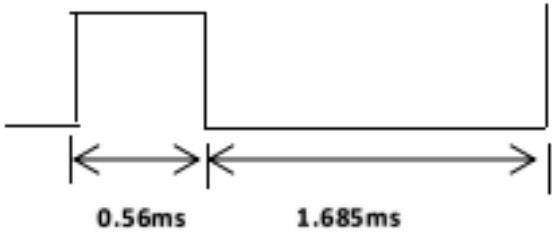
Basic Practices

19) IR receive data





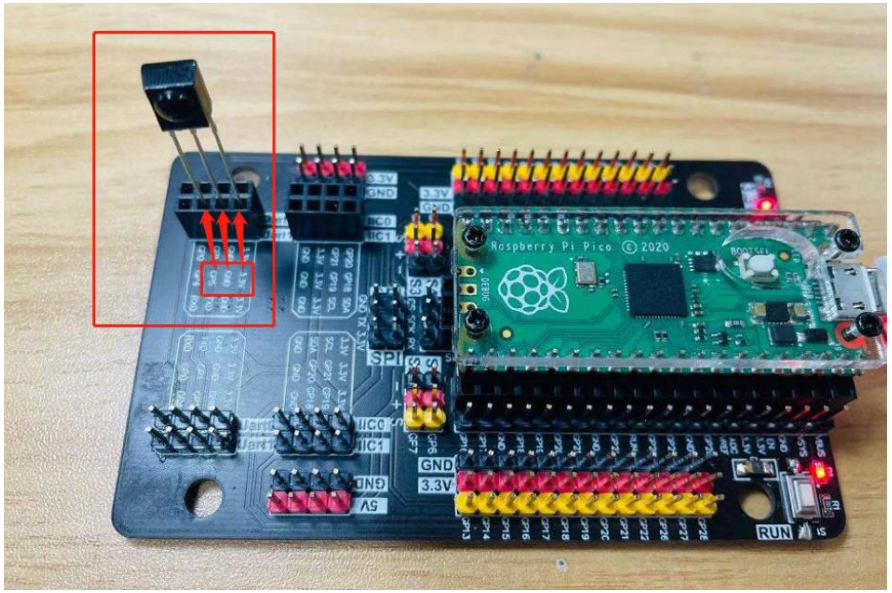


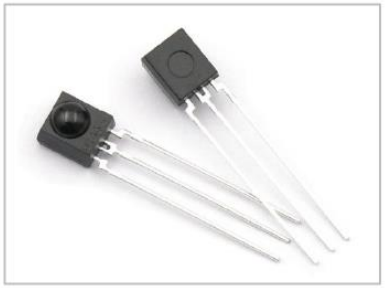


Basic Practices

19) IR receive data

- When the program is running, point the transmitter of the infrared remote control at the infrared receiving head, press the keys on IR controller, the shell window will print the currently received key code value.





5 Network controls

Network Controls

Wi-Fi Control

Wi-Fi Scanning

- When you run the provided Python code, it will display the name, Mac Address, and signal strength of the Wi-Fi router detected on the Raspberry Pi Pico W board.

```
* Mffice_1
 ~ BSSID: 44:48:c1:fc:8f:20
 ~ Channel : 11
 ~ RSSI : -47
--------------------------------
* U+NetC0AB
 ~ BSSID: 08:5d:dd:fa:c0:a9
 ~ Channel : 1
 ~ RSSI : -81
--------------------------------
* iptime.
 ~ BSSID: 88:36:6c:b2:e6:24
 ~ Channel : 10
 ~ RSSI : -71
```

5 Network Controls

Wi-Fi Control

Wi-Fi Connecting

- Modify your python code.
  - ✓ ssid = '___Router_name___'
  - ✓ password = '_____Pass_Word____'

- When attempting to connect Wi-Fi, the LED blinks
- When connected, the LED will turn on and the following message will be displayed.

```
MicroPython v1.23.0 on 2024-06-02; Raspberry Pi Pico W with RP2040
Type "help()" for more information or .help for custom vREPL commands.

>>>
Connected to Mffice_1 successfully!
Network Config: ('192.168.0.227', '255.255.255.0', '192.168.0.1', '192.168.0.1')
```

5 Network Controls

Wi-Fi Control

Wi-Fi Static IP setting

- How to set a static IP when connecting to Wi-Fi.
- You can connect to the network using a specific IP address on a MicroPython-based device.

```
>>>
Connecting to network...
Connected to network: ('192.168.137.150', '255.255.255.0', '192.168.137.1', '8.8.8.8')
```

5 Network Controls

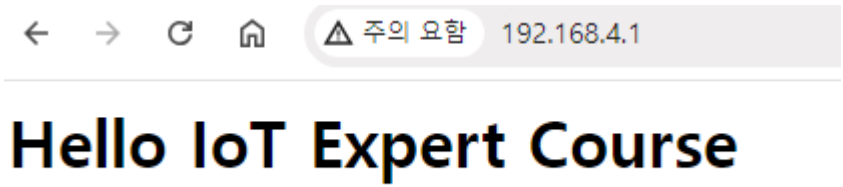
Wi-Fi Control

❖ Raspberry Pi Pico AP(Access Point) Mode

- Set up your Raspberry Pi Pico as an AP so external devices can connect to that network.
- You can connect using the SSID and password you set.

```
>>>
AP Mode Is Active, You can Now Connect
IP Address To Connect to:: 192.168.4.1
```

```
ap_mode('PICO_AP', '123456789')
```



5 Network Controls

Wi-Fi Control

NTP(Network Time Protocol)

- NTP is a standard protocol for accurate time synchronization.
- It receives accurate information from a time server over the Internet and adjusts the time.
- The time provided is standard time and needs to be corrected for each region.

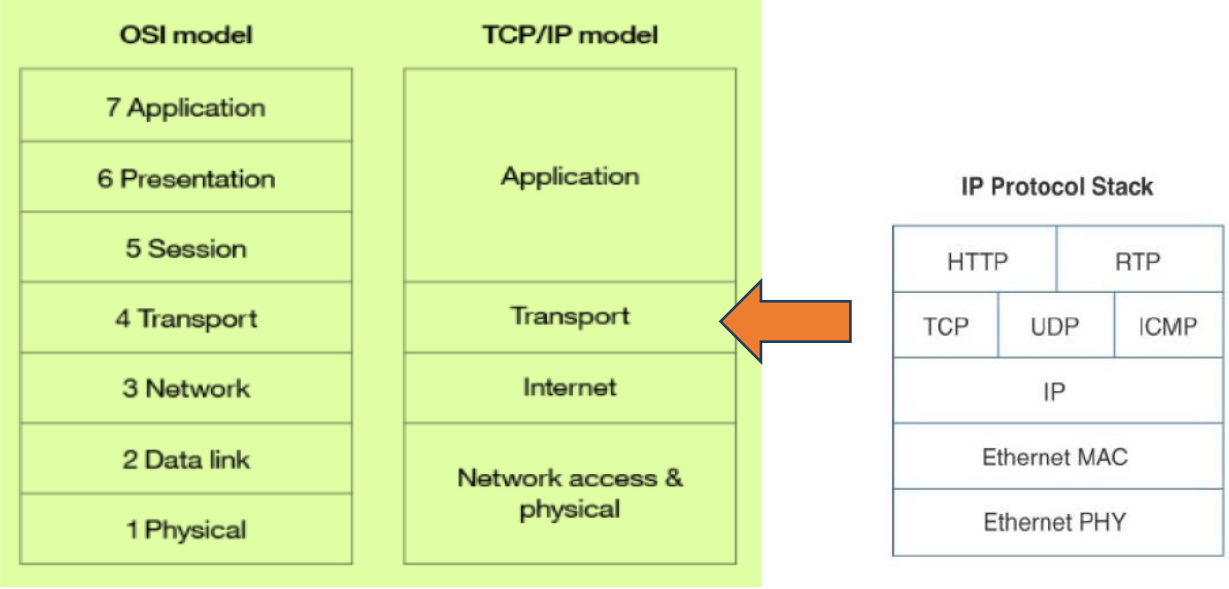
[RP_Pico_1_WiFi_5_NTP_Time.py]

Network Controls

TCP/UDP Basics

Networking standards and technologies

- The Open Systems Interconnection (OSI) model is an ISO-standard abstract model is a stack of seven protocol layers.
- From the top down, they are application, presentation, session, transport, network, data link and physical. TCP/IP, or the Internet Protocol suite, underpins the internet, and it provides a simplified concrete implementation of these layers in the OSI model.



5 Network Controls

TCP/UDP Basics

Transport service overview

- Provide service to application layer by using the service provided by network layer
- Hide physical network
  - ✓ Hide processing complexity
  - ✓ Hide different network technologies and architectures
- Provide reliable, host-to-host transport

Transport layer design issues

- Addressing
- Connection Establishment
- Connection Release
- Flow Control
- Error Detection and Crash Recovery

5 Network Controls

TCP/UDP Basics

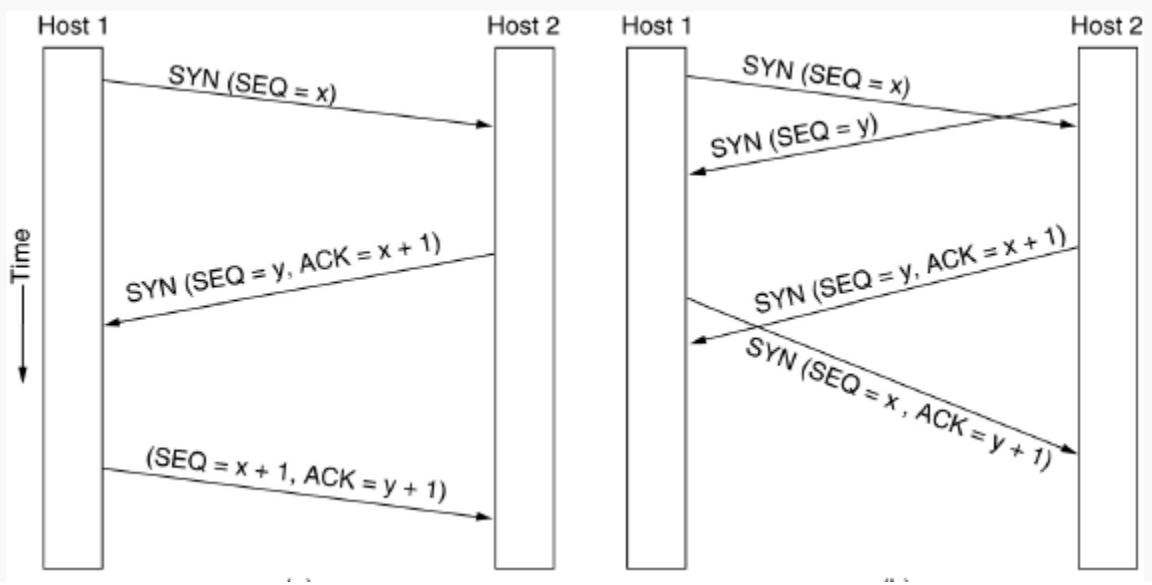
TCP - Addressing

- There are many network applications running on a host. When a packet arrive at network layer, how to know which application to send to?
  - ✓ Port: there are 216 = 65536 ports (0-65535) on one machine
  - ✓ One port is linked to only one application
  - ✓ One application may use many ports for different purposes
    - e.g. FTP: 20, 21
- How a client knows which service uses which port?
  - ✓ Permanent, well-known: often used service
    - 0-1023: well-known ports
    - 1024-49151: registered ports
    - 49152-65535: private ports
  - ✓ Process server proxy and create service on-the-fly: temporary service
  - ✓ Name server: for file service

Network Controls

TCP/UDP Basics

- TCP – Connection Establishment (three-way handshake)



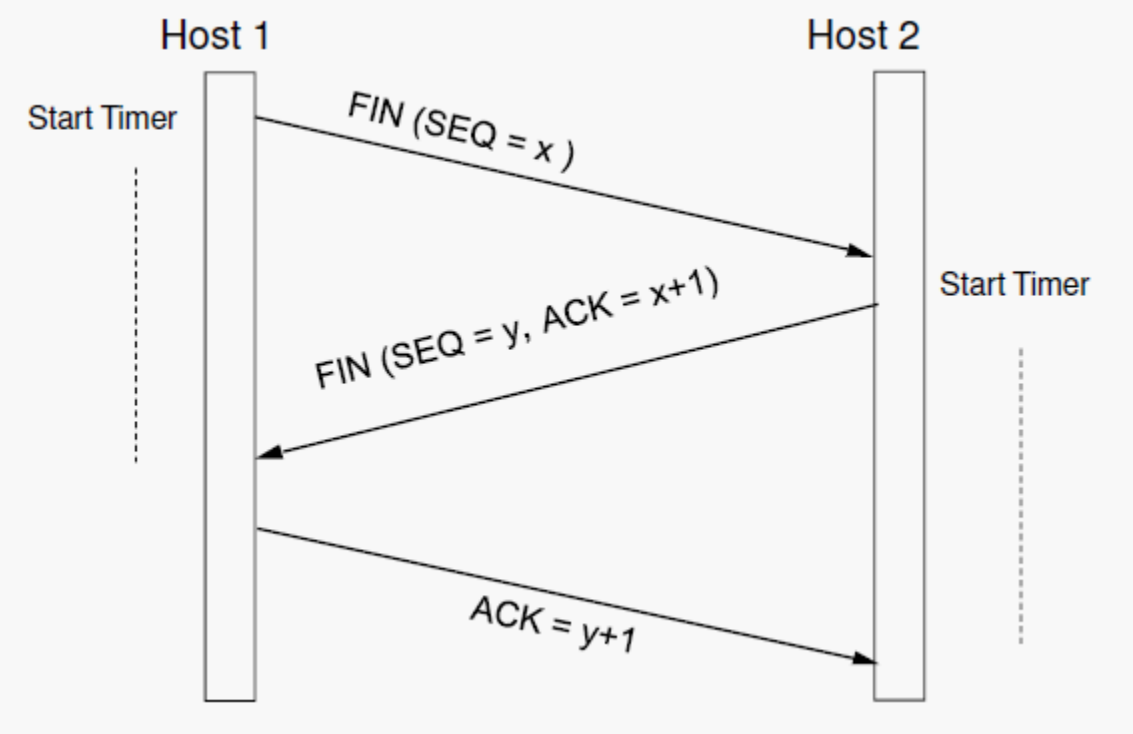
TCP connection establishment in the normal case

Call collision

Network Controls

TCP/UDP Basics

- TCP – Connection Release (Three-way handshake + timeout)



Network Controls

TCP/UDP Basics

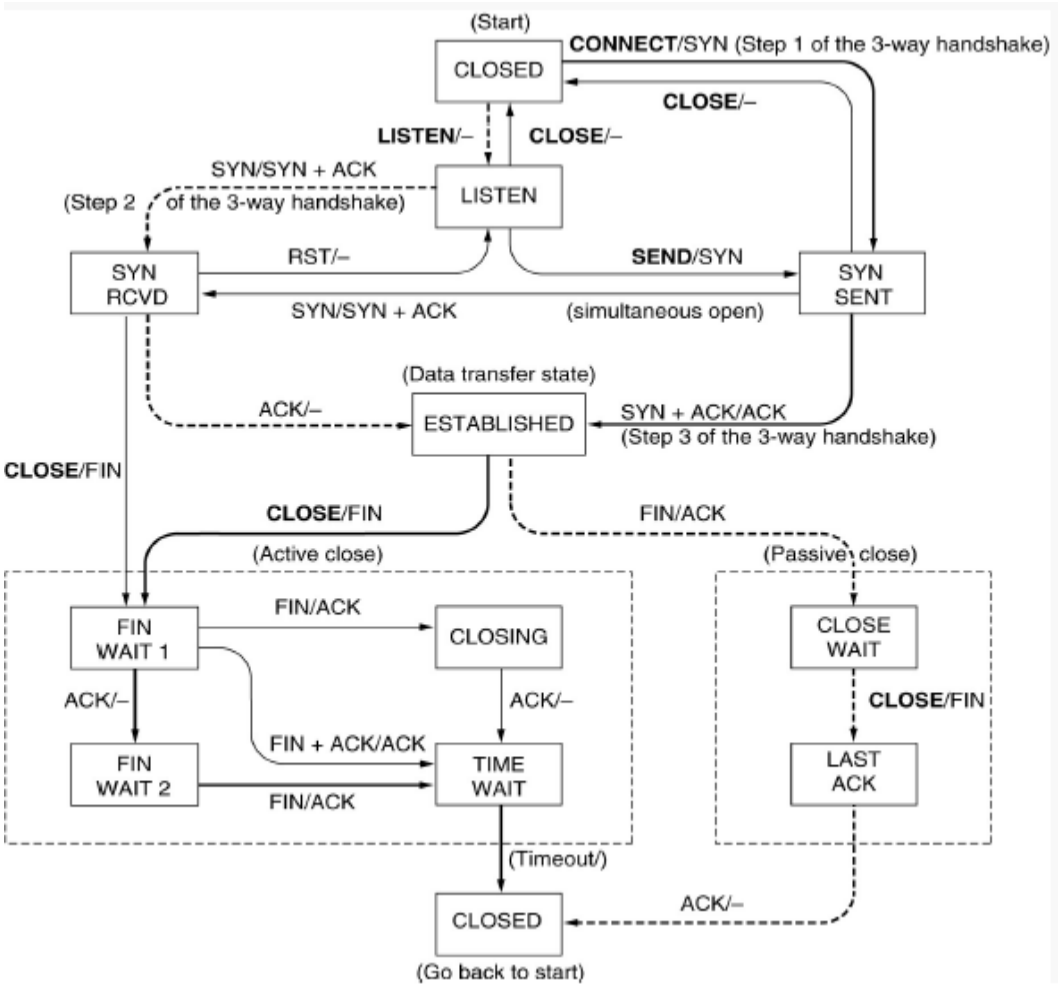
TCP – Service Primitives

Primitive	Meaning
SOCKET	Create a new communication end point
BIND	Attach a local address to a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Block the caller until a connection attempt arrives
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

Network Controls

TCP/UDP Basics

- TCP – Flow Control (TCP Finite State Machine)



Network Controls

TCP/UDP Basics

TCP – Application examples

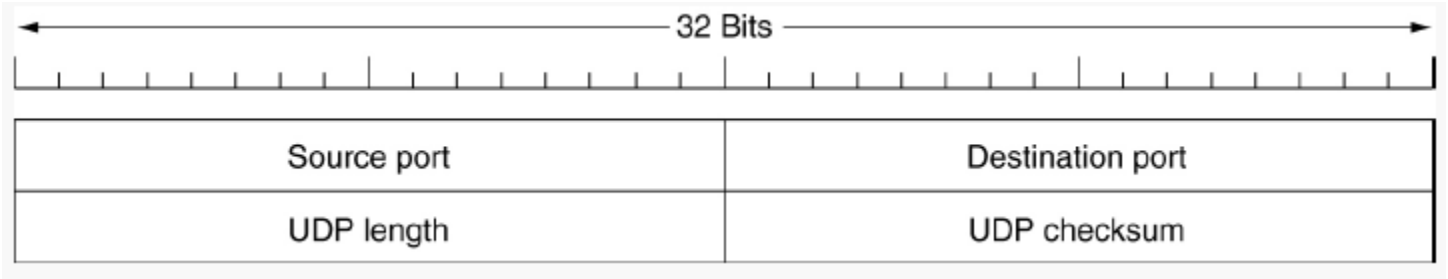
- When to use TCP:
  - ✓ When an application need a reliable transport

- Examples
  - ✓ File Transfer Protocol : FTP (21)
  - ✓ Secure Shell: SSH (22)
  - ✓ Teletype Network: TELNET (23)
  - ✓ Simple Mail Transfer Protocol: SMTP (25)
  - ✓ Hypertext Transfer Protocol: HTTP (80)

Network Controls

TCP/UDP Basics

UDP – Header



- UDP Destination Port: identifies destination process
- UDP Source Port: optional – identifies source process for replies, or zero
- Message Length: length of datagram in bytes, including header and data
- Checksum: optional -- 16-bit checksum over header and data, or zero

Network Controls

TCP/UDP Basics

UDP – Properties

- UDP provides an unreliable datagram service
  - ✓ Packets may be lost or delivered out of order
  - ✓ Message split into datagrams, user sends datagrams as packets on network layer
  - ✓ No buffer at either sending or receiving side
  - ✓ Unreliable but fast
  - ✓ Full duplex
  - ✓ Application must deal with lost packets

Network Controls

TCP/UDP Basics

UDP – Application Examples

- When to use UDP
  - ✓ Reduce the requirement of computer resources
  - ✓ The checking scheme has provided completely by the application program
  - ✓ When using the Multicast or Broadcast to transfer
  - ✓ The transmission of Real-time packets

- Examples
  - ✓ Trivial File Transfer Protocol , TFTP
  - ✓ Simple Network Management Protocol , SNMP
  - ✓ Dynamic Host Configuration Protocol , DHCP
  - ✓ Domain Name System , DNS
  - ✓ Routing Information Protocol , RIP
  - ✓ Real-Time Transport Protocol , RTP

Network Controls

TCP/UDP Basics

- TCP vs UDP

TCP	UDP
connection-oriented	connectionless
confirmed service	unconfirmed service
high overhead (header 20 bytes)	low overhead (header 8 bytes)
flow control	no flow control

Network Controls

Download Hercules utility

- Hercules utility https://www.hw-group.com/software/hercules-setup-utility

Hercules SETUP utility

Hercules SETUP utility is useful serial port terminal (RS-485 or RS-232 terminal), UDP/IP terminal and TCP/IP Client Server terminal. It was created for HW group internal use only, but today it's includes many functions in one utility and it's Freeware! With our original devices (Serial/Ethernet Converter, RS-232/Ethernet Buffer or I/O Controller) it can be used for the UDP Config.

Licence type: Freeware

SW version: Hercules

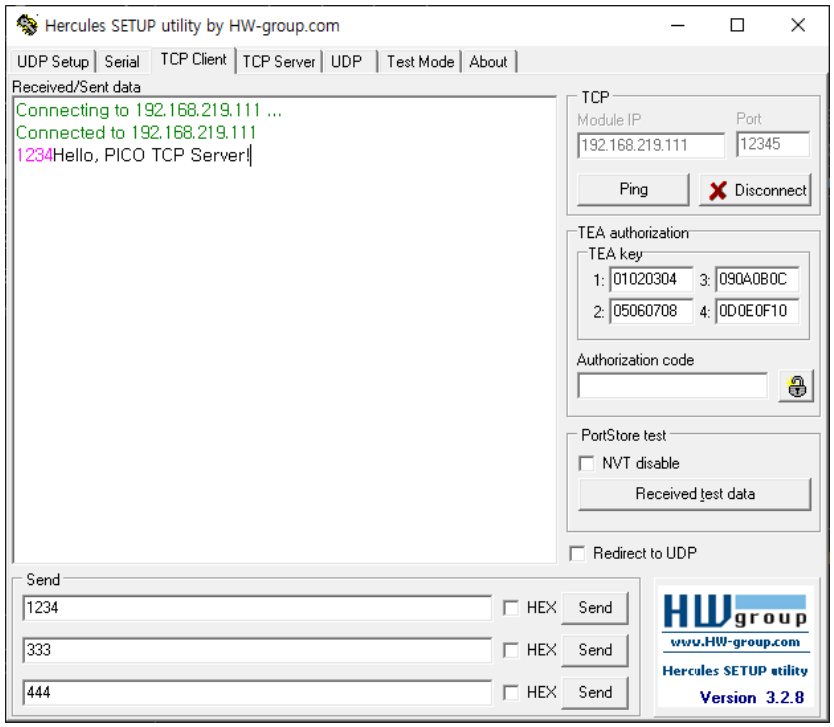
Download latest version of **Hercules** | ver. 3.2.8

5 Network Controls

TCP

TCP Server Test

```
>>>
Connecting to network...
Connected to network: ('192.168.219.111', '255.255.255.0', '192.168.219.1', '1.214.68.2')
TCP server started on port 12345
Accepted connection from ('192.168.219.110', 26500)
Received data: 1234
```



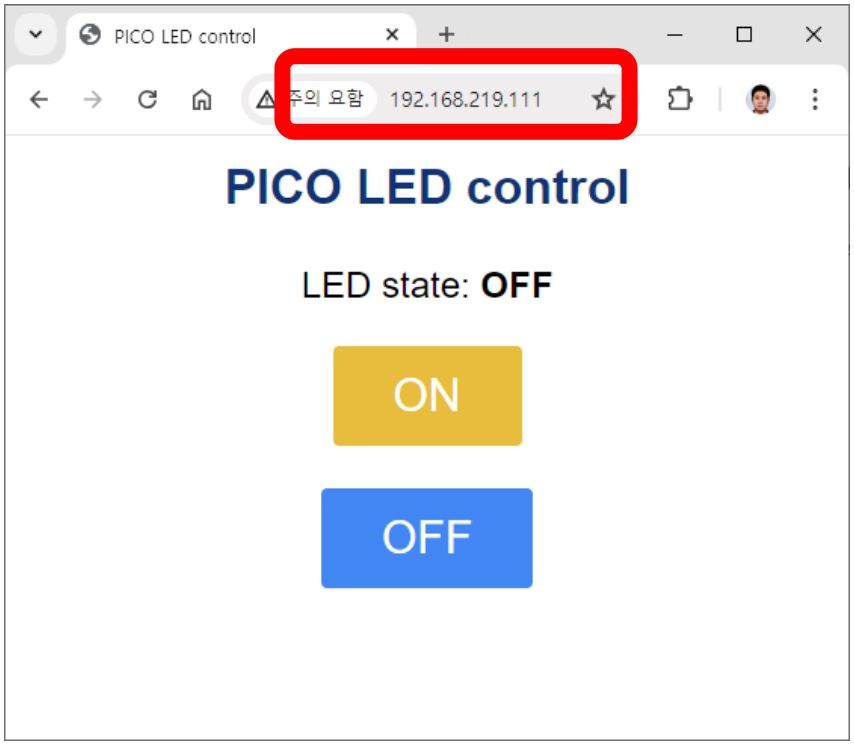
- Connect to Internet Router
- Raspberry Pi Pico operates as a TCP Server.
- Test with the TCP Client of the Hecules utility with the assigned IP and the port number set in the source code.

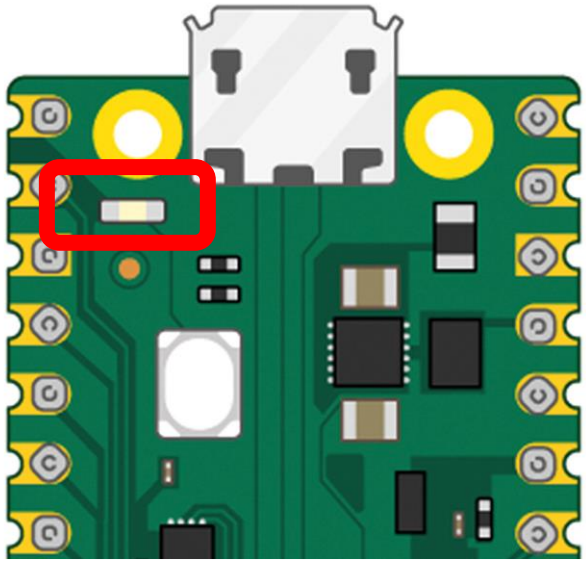
5 Network Controls

Web-Server

Web-Server LED Control

```
>>>
Connecting to network...
Connected to network: ('192.168.219.111', '255.255.255.0', '192.168.219.1', '1.214.68.2')
```



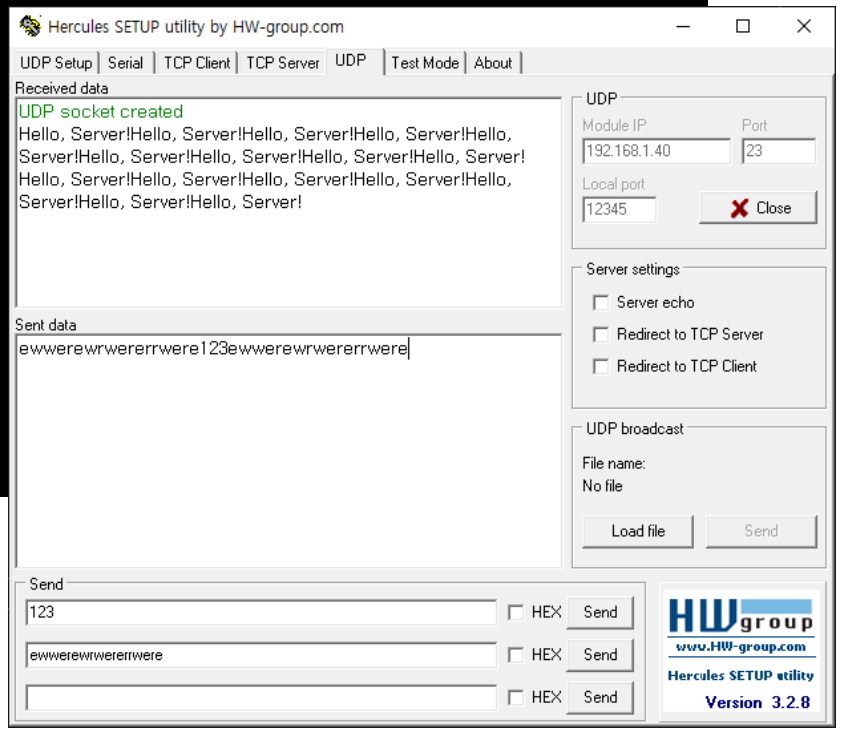


Network Controls

UDP

- UDP Protocol Test

```
Connecting to network...
Connected to N-517 successfully!
Network Config : ('192.168.1.40', '255.255.255.0', '192.168.1.1', '192.168.1.1')
Connected to server
Sent data: Hello, Server!
Sent data: Hello, Server!
Sent data: Hello, Server!
Sent data: Hello, Server!
Sent data: Hello, Server!
Sent data: Hello, Server!
Sent data: Hello, Server!
Sent data: Hello, Server!
Received data: ewwerewrwererrwere
Sent data: Hello, Server!
Received data: 123
Sent data: Hello, Server!
```

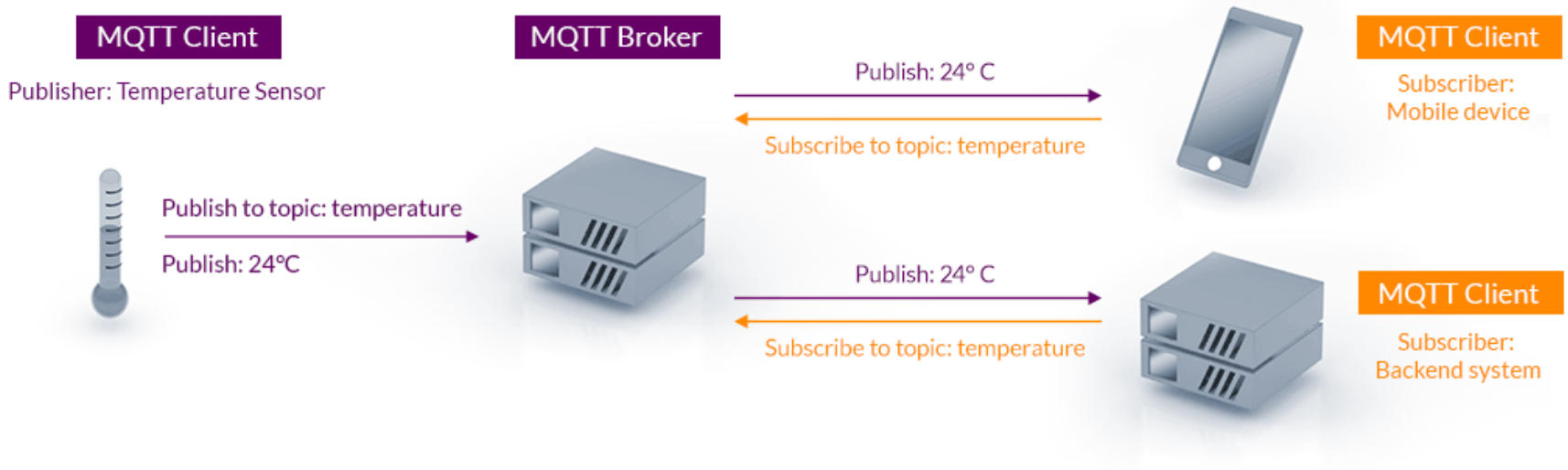


Network Controls

MQTT

Message Queuing Telemetry Transport

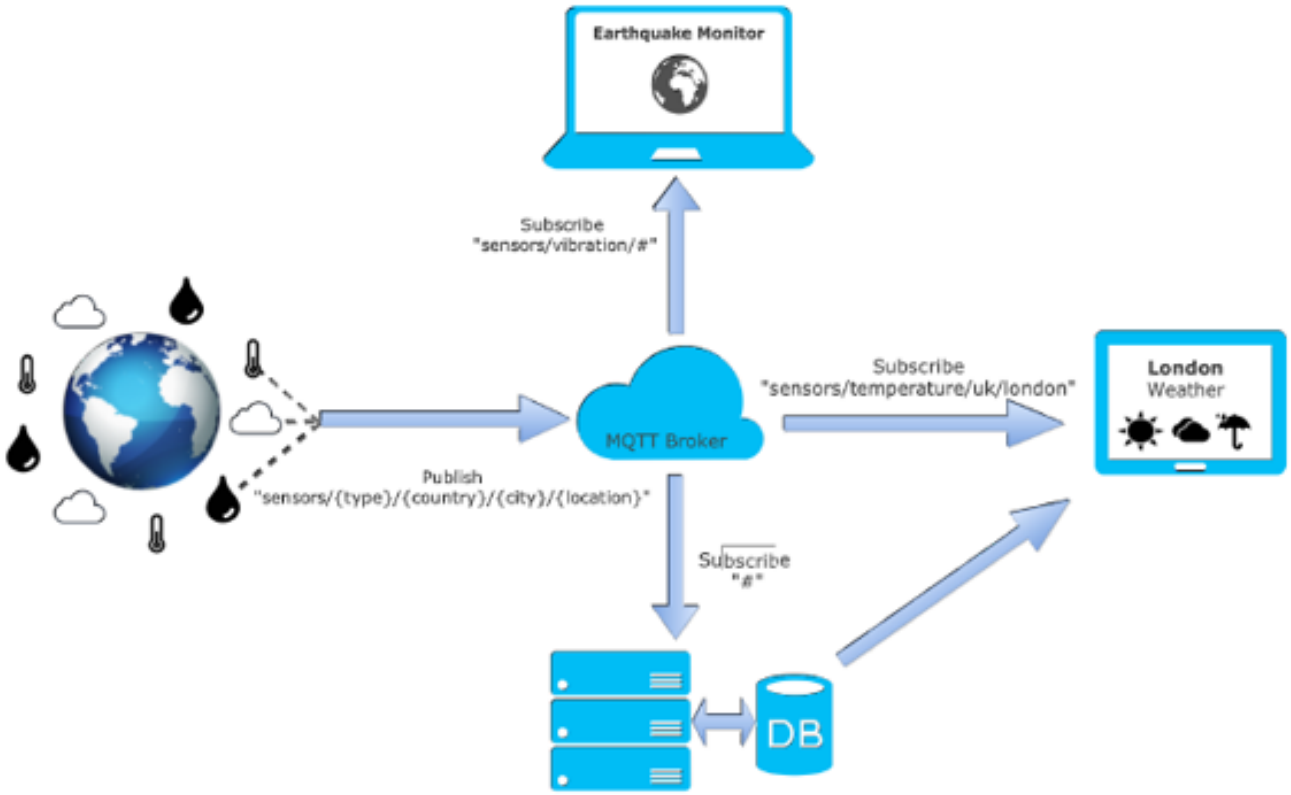
- MQTT (Message Queuing Telemetry Transport) is a lightweight, open-source messaging protocol that allows devices to communicate with each other using a publish-subscribe model. It's designed for devices with limited resources, such as those in the Internet of Things (IoT), and for networks with low bandwidth, high latency, or unreliability.



Network Controls

MQTT

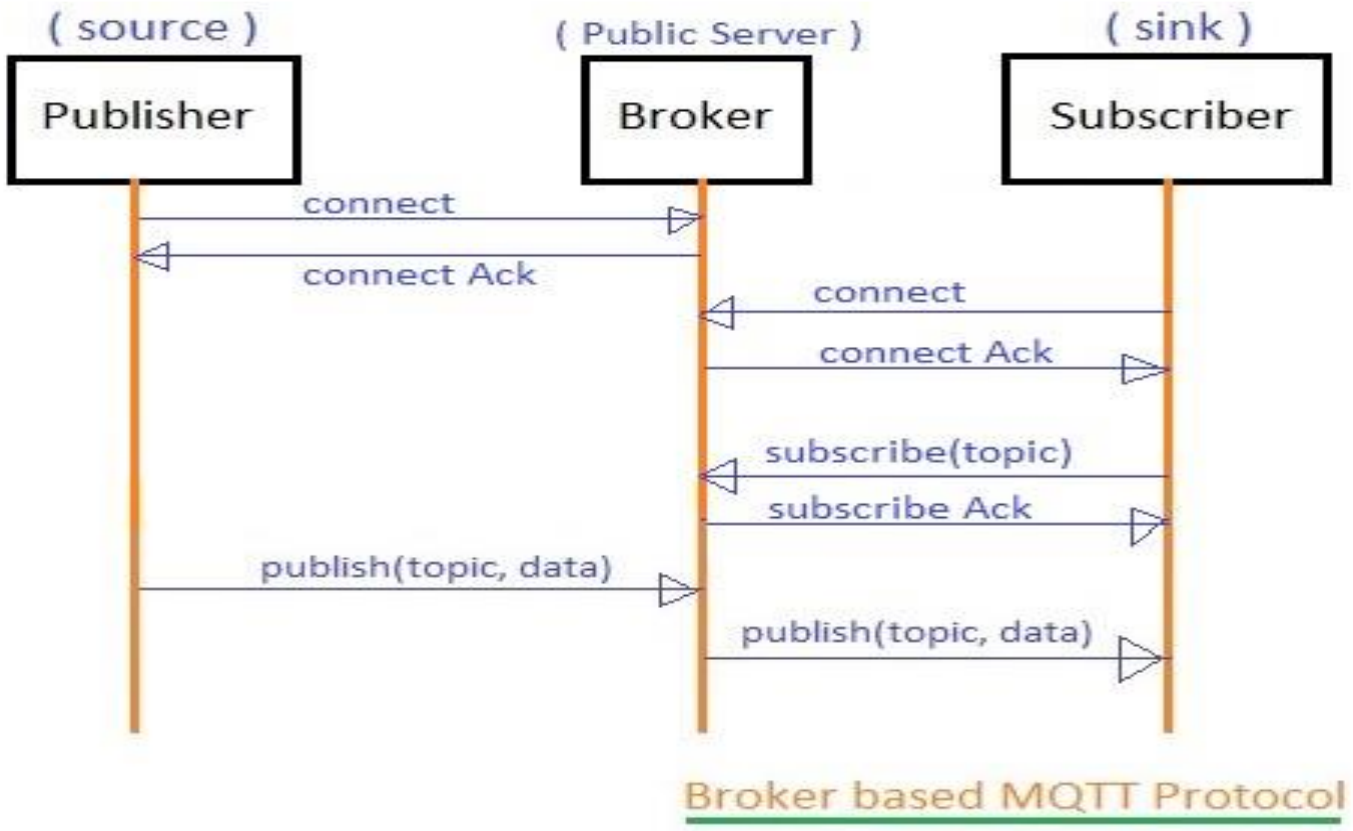
- Message Queuing Telemetry Transport



Network Controls

MQTT

- Message Queuing Telemetry Transport

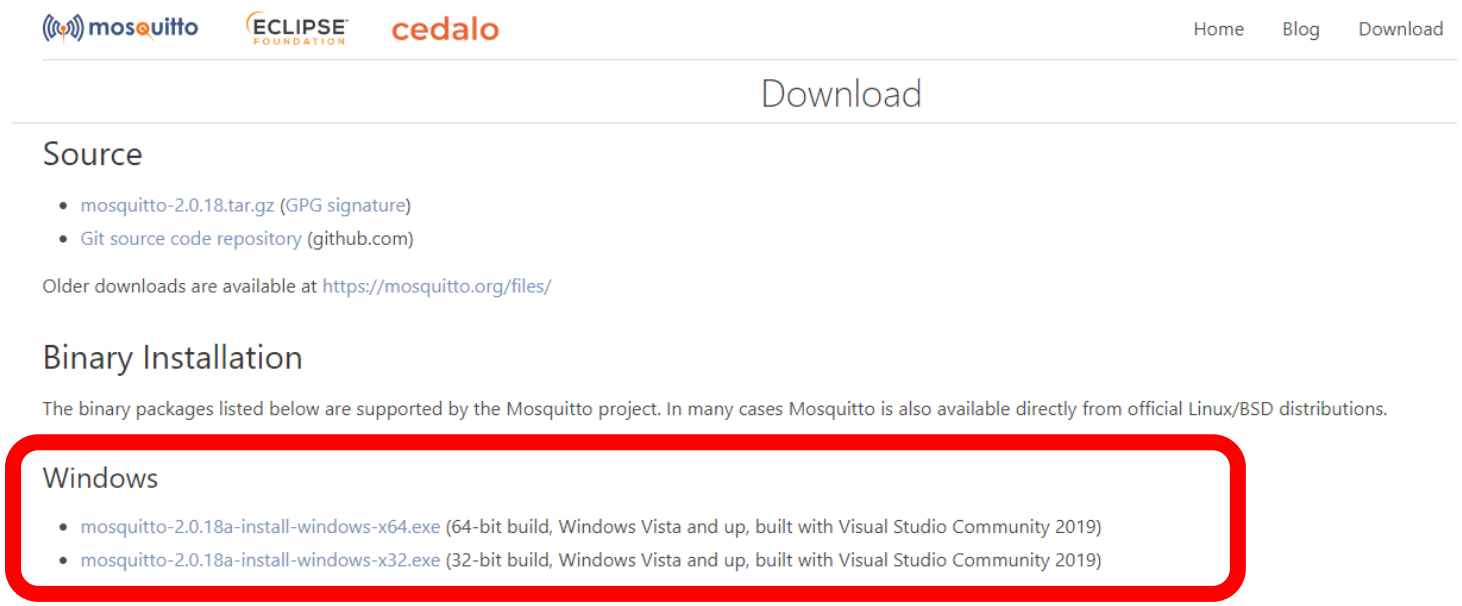


5 Network Controls

MQTT

- Message Queuing Telemetry Transport

https://mosquitto.org/download/

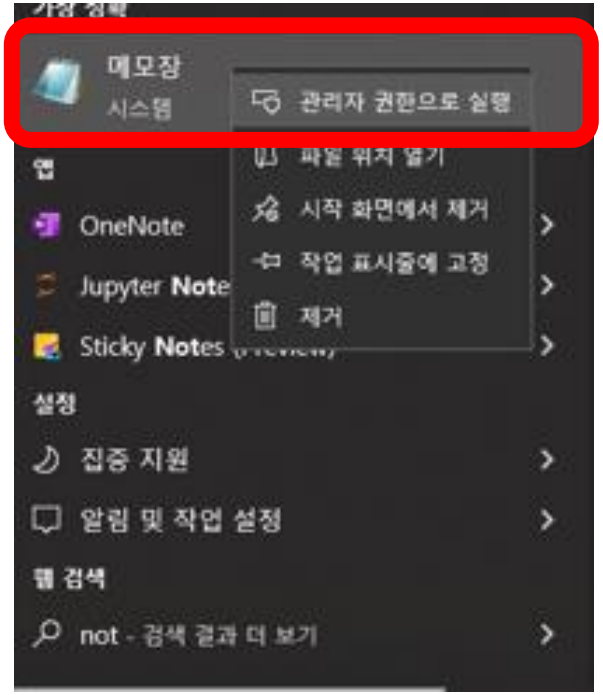


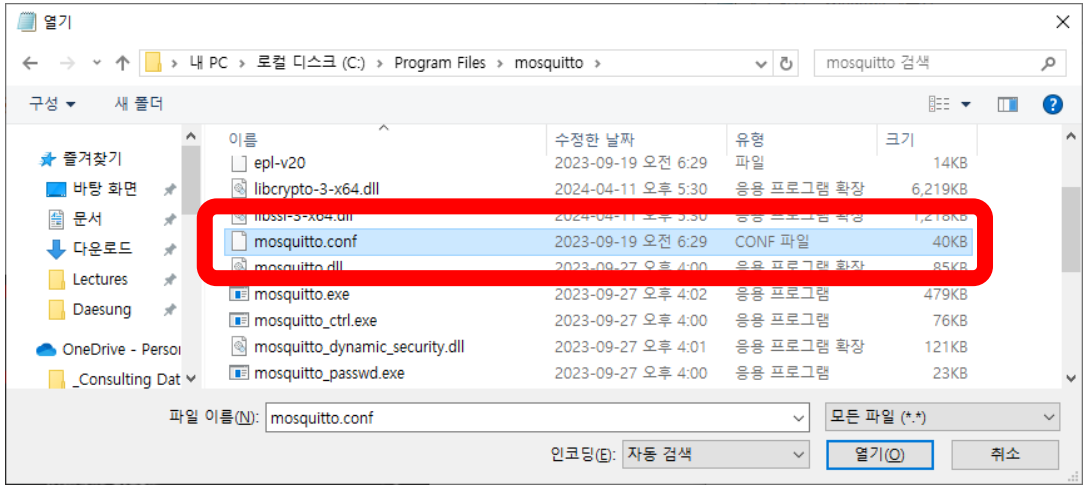
Network Controls

MQTT

Message Queuing Telemetry Transport

- Notepad runs as administrator.
- Open mosquito.conf





5 Network Controls

MQTT

Message Queuing Telemetry Transport

- Find #allow_anonymous (false → true)

```
# Defaults to false, unless there are no listeners defined in the configuration
# file, in which case it is set to true, but connections are only allowed from
# the local machine.
allow_anonymous true
```

- Find # listener port-number [ip address/host name/unix socket path] (blank → 1883)

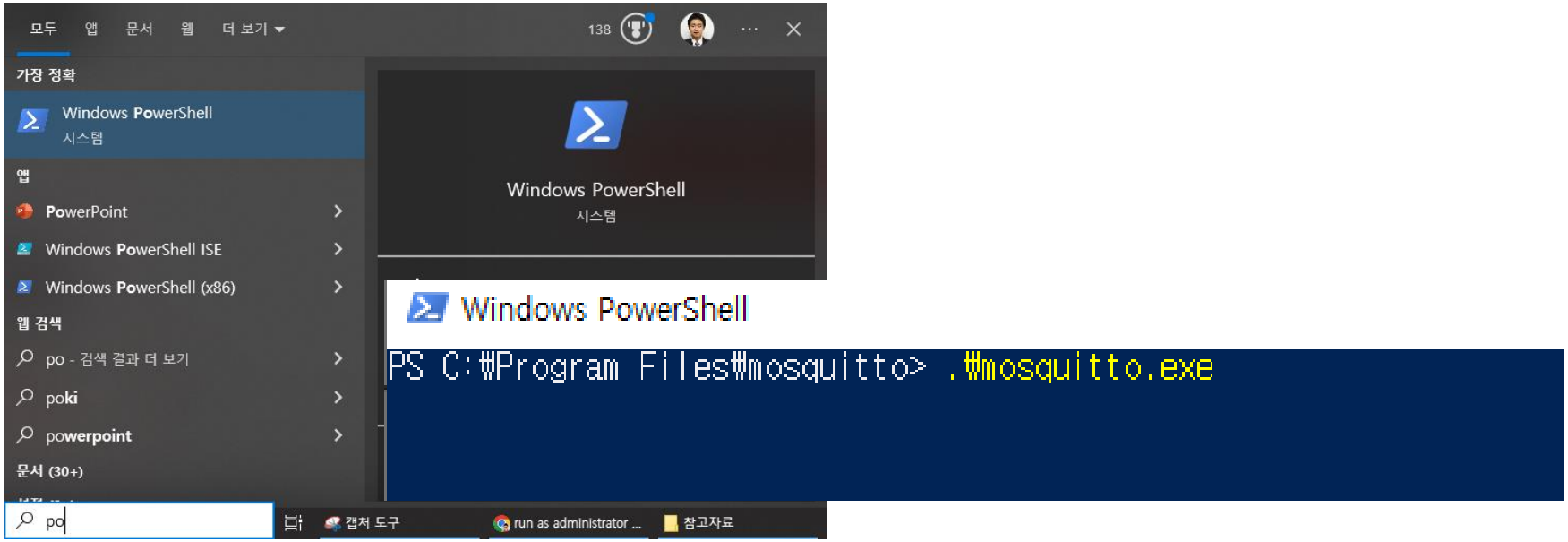
```
#
# listener port-number [ip address/host name/unix socket path]
listener 1883
```

Network Controls

MQTT

Message Queuing Telemetry Transport

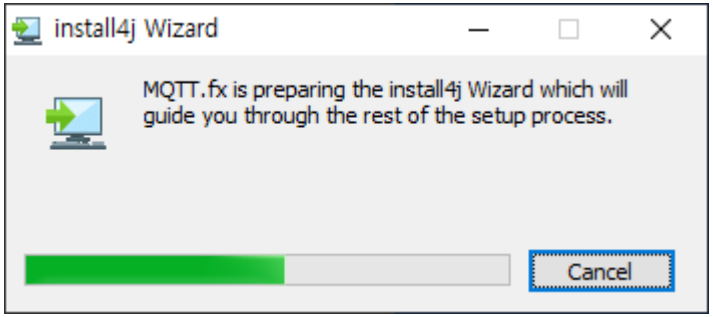
- Run windows PowerShell and execute mosquitto.exe

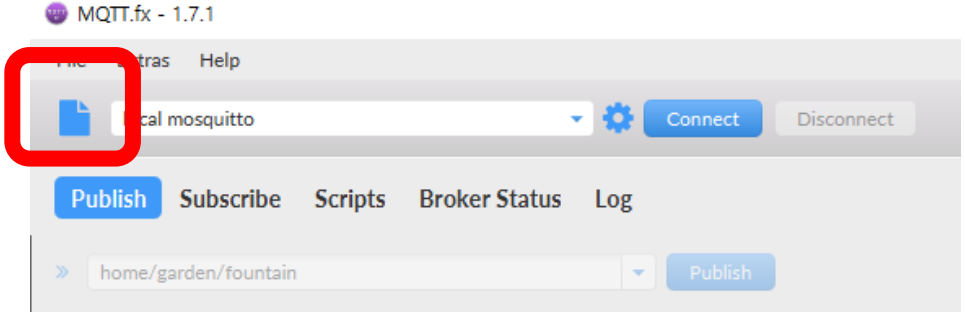


5 Network Controls

MQTT

- Message Queuing Telemetry Transport
  - Download free licensed version MQTT.FX 1.7.1 and install, run



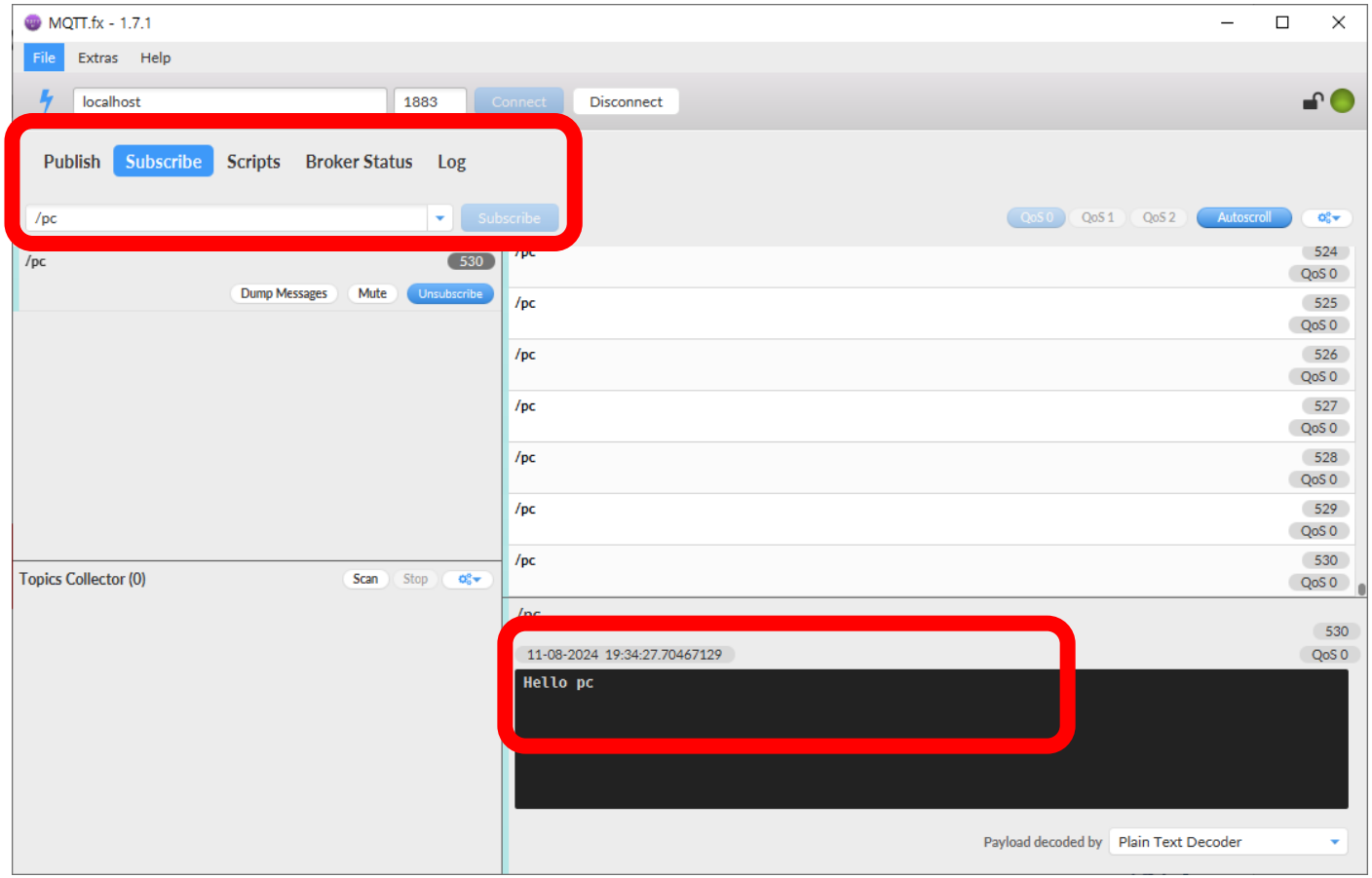


5 Network Controls

MQTT

MQTT Subscribe Topic

- RP_Pico → MQTT Fx

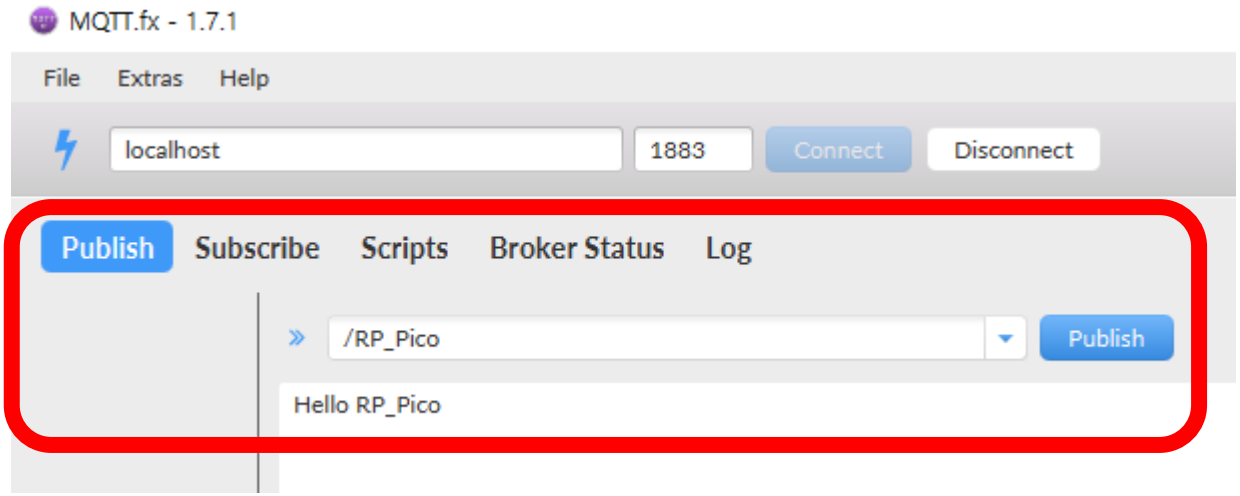


5 Network Controls

MQTT

MQTT Publish

- MQTT Fx → RP_Pico



Network Controls

Data Representation

- Java Script Object Notation(JSON)

```
{"menu": {
  "id": "file",
  "value": "File",
  "popup": {
      "menuitem": [
         {"value": "New", "onclick": "NewDoc()"},
         {"value": "Open", "onclick": "OpenDoc()"},
         {"value": "Close", "onclick": "CloseDoc()"}
      ]
    }
  }
}
```

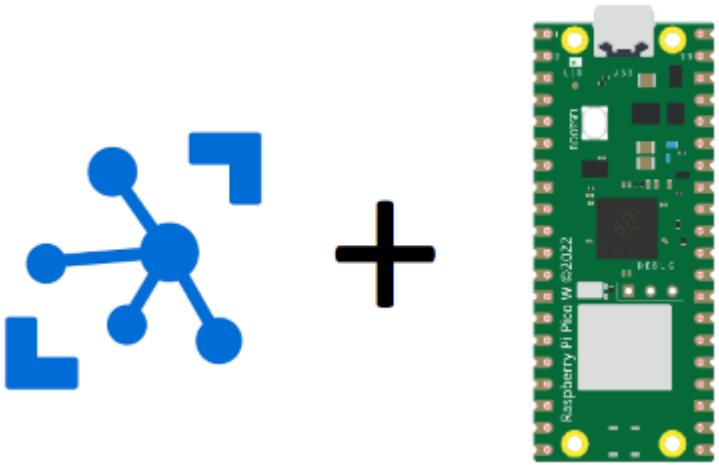
```
>>> import json

>>> d = {'sensorId': 'temp1', 'Value': 25}

>>> d
{'sensorId': 'temp1', 'Value': 25}
>>> d['sensorId']
'temp1'

>>> dj = json.dumps(d)
>>> dj
'{"sensorId": "temp1", "Value": 25}'

>>> nd = json.loads(dj)
>>> nd
{u'sensorId': u'temp1', u'Value': 25}
>>> nd['sensorId']
u'temp1'
```

6 RP Pi Pico W with Azure IoT Hub

6 RP Pi Pico W with Azure IoT Hub

Connecting a Raspberry Pi Pico W to IoT Hub

- Connecting a Raspberry Pi Pico W to Microsoft Azure IoT Hub using MicroPython and MQTT



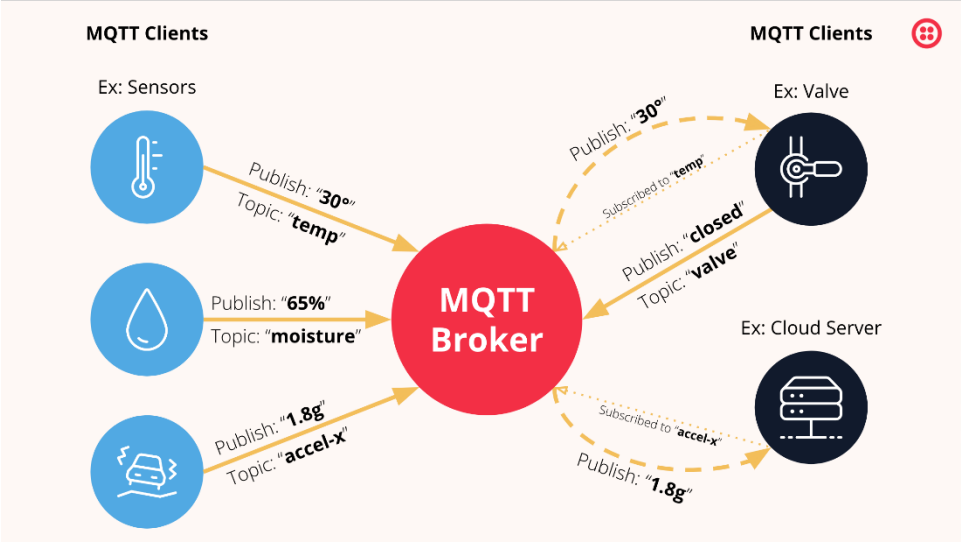
RP Pi Pico W with Azure IoT Hub

MQTT

What is MATT?

- MQTT stands for MQ (Message Queue) Telemetry Transport and is one of the most popular network protocols for IoT Applications due to it's light-weight nature.

- MQTT is a Publish-Subscribe protocol, where Devices are able to publish telemetry messages using a Topic and receiving devices can Subscribe to that topic to receive those messages.



6 RP Pi Pico W with Azure IoT Hub

Azure

Microsoft Azure IoT Hub

- The Microsoft Azure IoT Hub is a service which sits at the edge of Azure, allowing millions of devices to connect to the IoT Hub and then route those messages to other Azure services along the line.

- The IoT Hub has the concept of a Device Registry, where each connecting device has its own set of credentials to connect to the IoT Hub.

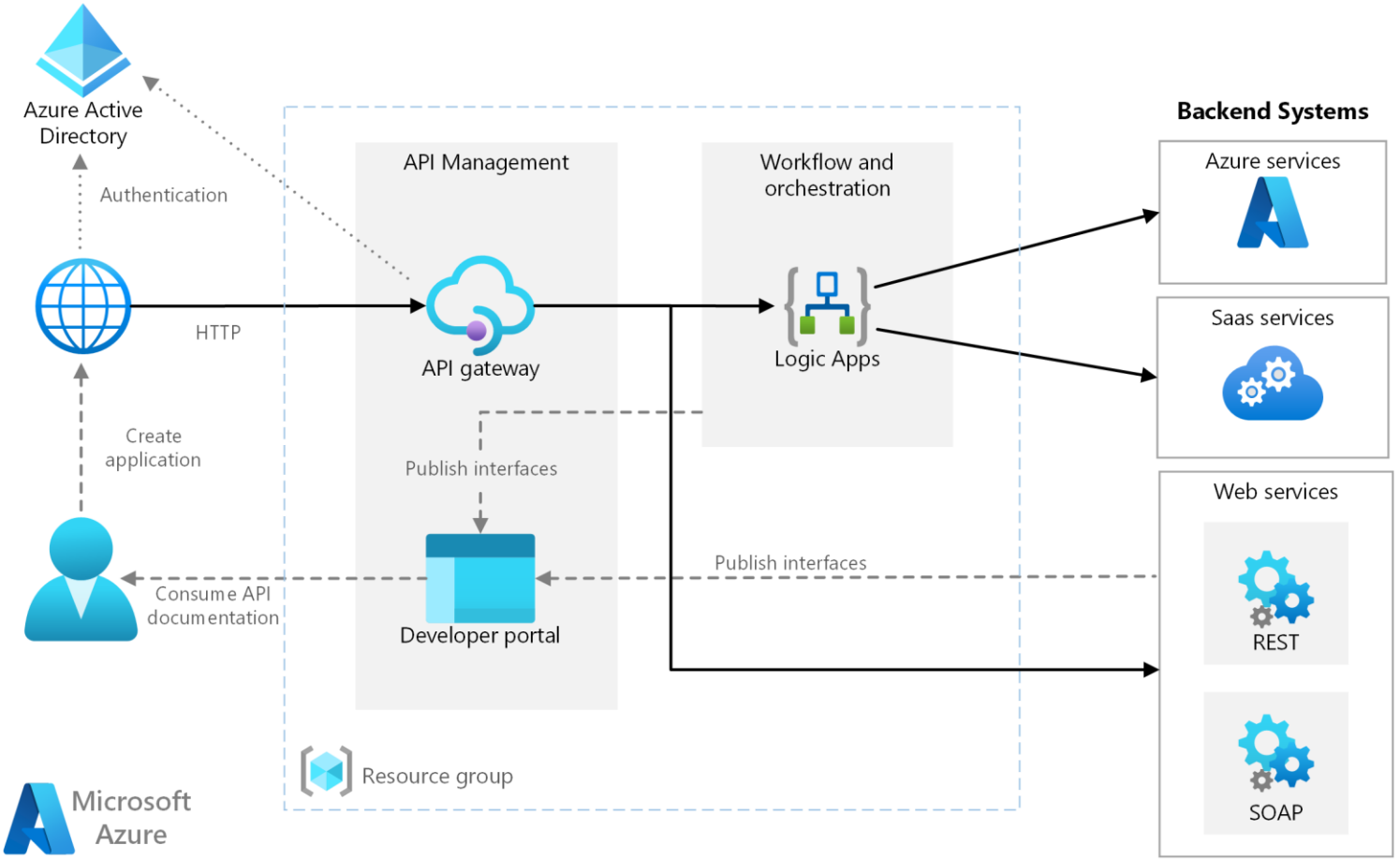
- IoT Hub is based on Azure Event Hubs, however where Event Hub communication is one way from Sender to Receiver, IoT hubs allows bi-directional communication between the IoT Hub and the connected device.

- IoT Hub supports communication using both AMQP (Advanced Message Queueing Protocol) and MQTT.

6 RP Pi Pico W with Azure IoT Hub

Azure

- Microsoft Azure IoT Hub



RP Pi Pico W with Azure IoT Hub

- Connecting a Raspberry Pi Pico W to IoT Hub Practice

- Blank -


Q & A