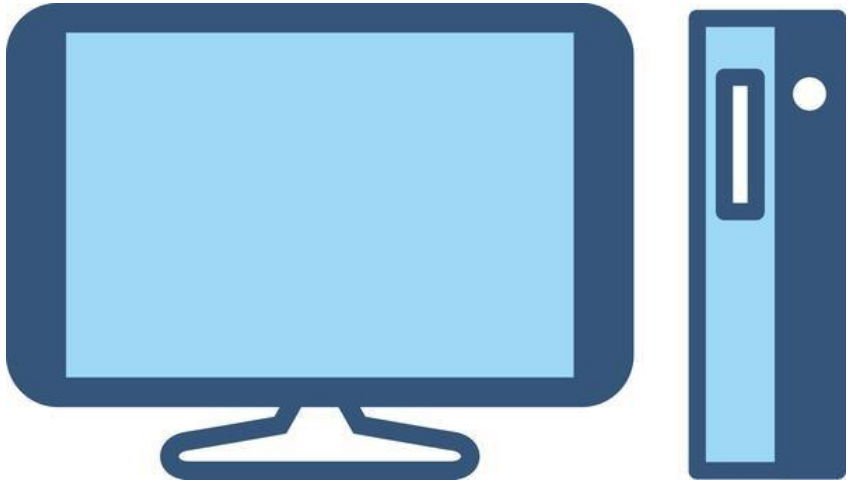
PINGPONG 3 PYTHON

Управління МАШИНОЮ

Матеріали

<Машина>

<USB-донгл>



<Комп'ютер>

Початок

- PYTHON 3.6.6

- VISUAL STUDIO CODE

https://code.visualstudio.com/

- PINGPONG API

Посилання тут

Для детального пояснення звертайтеся до керівництва

Початок

Pingpong.

Мета

Управління кубом Pingpong за допомогою Python через USB-донгл

Імпорт

Імпорт модуля для відповідного коду.

Модуль - це інструмент, як "молоток"!
Вам потрібно взяти його з собою, щоб скористатися ним.

Імпорт - це команда для виконання цієї функції.

Імпорт

Імпорт модуля для відповідного коду.

Модуль - це інструмент, як "молоток"!
Вам потрібно взяти його з собою, щоб скористатися ним.
Імпорт - це команда для виконання цієї функції.

API PingPong + Клавіатура

Нам потрібні ці два інструменти.
API - це інструмент управління кубом Pingpong .
Клавіатура - це інструмент для управління кубом Pingpong за допомогою клавіатури.

Імпорт

Autocar.py

```
from pingpongthread import PingPongThread
import keyboard
```

Екземпляр

Що таке клас?

Клас - це свого роду "шаблон", якою можна скористатись, коли вам потрібні змінні та функції. Наприклад, на заводі ми хочемо виробляти джойстики. Джойстики мають бути однакової форми, чи не так? ? "Шаблон", який задає цю саму фігуру, і є клас.

```
class Factory:

    def setData(self, title, price, author):
        self.title = title
        self.price = price
        self.maker = maker

    def printData(self):
        print '제목 : ', self.title
        print '가격 : ', self.price
        print '메이커 : ', self.maker
```

Що таке екземпляр?

Екземпляр (instance) - це джойстик, який ми виймаємо з шаблону "класу".
За допомогою цього екземпляра ми створюємо тимчасовий джойстик для управління кубом Pingpong.

Екземпляр

Давайте створимо екземпляр!

Як Pingpong може створити екземпляр?

Екземпляр

Autocar.py

Кількість кубів, необхідна для управління машиною

```
from pingpongthread import PingPongThread
import keyboard

PingPongThreadInstance = PingPongThread(number=2)
```

Підключення

Давайте підключимось до куба Pingpong

Якщо ви створили екземпляр, то можете підключитися до куба Pingpong.
Давайте підключимо його за допомогою наступного коду!
(Для підключення куба необхідний USB-донгл)

Підключення

Autocar.py

```
from pingpongthread import PingPongThread
import keyboard

PingPongThreadInstance = PingPongThread(number=2)
PingPongThreadInstance.start()
PingPongThreadInstance.wait_until_full_connect()
```

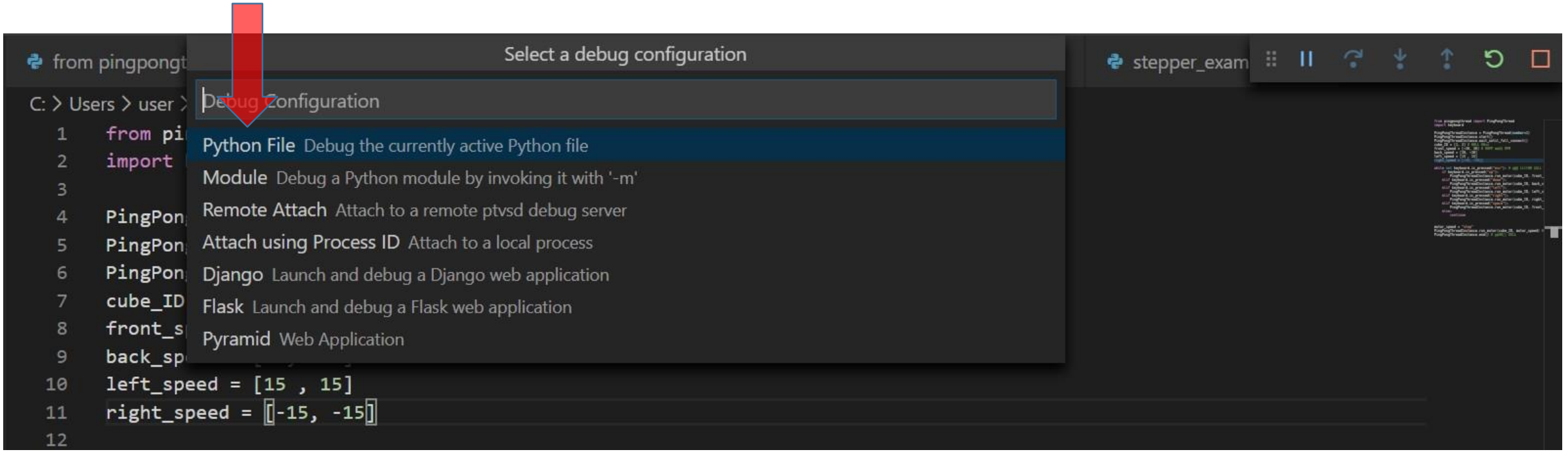
Команда для початку з'єднання

Команда, поки всі потрібні куби повністю підключаться.

Підключення

Давайте підключимось до куба Pingpong

Давайте перевіримо наявність підключення Давайте керуватимемо цим, вибравши “Visual Studio Code에서 F5 -> Python File“ після підтвердження підключення USB-донгла!

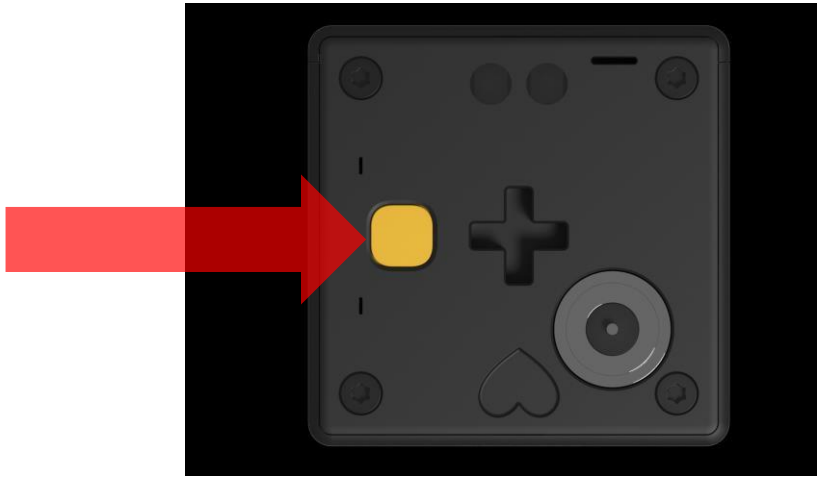


Підключення

Давайте підключимось до куба Pingpong

Коли на терміналі коду Visual Studio з'являється повідомлення про підключення, послідовно підключіть куб Pingpong, натиснувши на ньому кнопку.
Куб, підключений першим, стає керівним, потім наступний стає керованим.

При послідовному з'єднанні ідентифікатори куба нумеруються як 1, 2, 3, ...



Коли куб вимкнено, натисніть кнопку один раз, щоб увімкнути куб (ви почуєте звуковий сигнал). Після цього, коли ви ще раз натиснете кнопку, пролунає

Підключення

звук з'єднання. Після цього з’єднання буде завершено.

Двигун

Управління двигуном

Тепер вам потрібно просто запустити двигун!
Давайте надішлемо з комп'ютера сигнал про запуск двигуна!

Ми будемо користуватись наступними фукнціями.

```
def run_motor(self,
              cube_ID_list="all",
              speed_list=None,
              step_list=None,
              pause_list=False,
              time_list=None,
              discovery_group=None,
              run_option="continue"
              , speed_option="RPM",
              step_option="CYCLE",
              sync=False,
              time_option=None,
              wait=0)
```

Це виглядає складно, але насправді простіше, ніж ви думаєте!

(Для деталей звертайтеся до керівництва PingPong.)

Зверніть увагу лише на дві речі!

Двигун

Решту налаштувань можна не змінювати.

Двигун

Autocar.py

```
from pingpongthread import
PingPongThread


PingPongThreadInstance =
PingPongThread(number=2)
PingPongThreadInstance.start()


PingPongThreadInstance.run_motor(1, 2], [-30, 30] )
```

Cube_ID_list Speed_list

Якщо код виконується у терміналі,

ви можете виявити, що Куб 1 рухається зі швидкістю -30, а Куб 2 рухається зі швидкістю +30.

Структура

Створення структури

Тепер ви також можете керувати двигуном. Тож давайте побудуємо структуру, щоб переміщувати куб, як хочемо!
Ми будемо рухати машину за допомогою клавіатури.

Тут ми будемо використовувати інструмент "Клавіатура", який ми імпортували.

Клавіатура

Модуль клавіатури налаштований на виявлення комп'ютером, коли користувач натискає кнопку на клавіатурі.

Для цього ми використовуємо "keyboard_is_pressed (" гаряча клавіша ")".

Структура

Keyboard.py

```
import keyboard

keyboard.is_pressed("esc")
```

Коли ви натискаєте клавішу "esc" на клавіатурі, справжнє значення повертається.

Структура

"Поки"

Інструкція "While" ("Поки") - це команда, яка використовується багаторазового виконання. Якщо умова в операторі "While" відповідає дійсності, це твердження триває неодноразово, доки умова не стане хибною!

Ми застосуємо його разом з модулем "Клавіатура", про який ми дізналися раніше.

Структура

Keyboard.py

```
import keyboard

While not keyboard.is_pressed(“esc”):
    print(“PingPong!”)
```

Коли ви запускаєте код, велика кількість значень PingPong буде виведена на термінал.
Це буде повторюватись, поки користувач не натисне клавішу "esc", тому натисніть
"esc" для зупинки.

Структура

"Якщо"

Оператор "If" ("Якщо") - це оператор умови, який виконує наступну команду, коли виконується умова. Якщо умова не виконується, команда пропускається.

Якщо ви хочете застосувати кілька умов, додайте "elif".

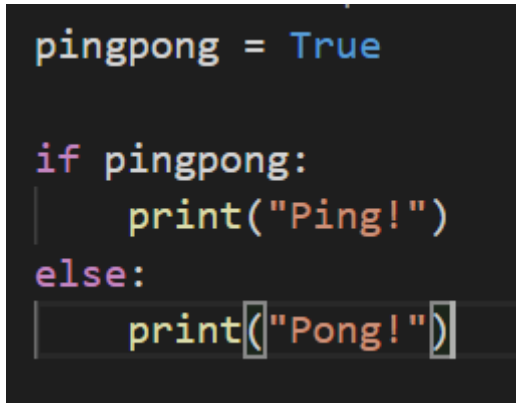
"Інакше"

Якщо ви хочете виконати певну команду, коли оператор If не виконується, ви можете додати оператор "else" ("інакше"). Тобто якщо оператор If є дійсним, він виконує команду після оператора If, а якщо оператор If не є дійсним, він виконує команду після оператора else.

Давайте застосуємо попередній приклад ще!

Структура

-> pingpong є дійсним, тому Ping відображається в терміналі. Якщо значення недійсне, відображатиметься Pong.



Структура

+ continue (продовжити) - це команда, яка використовується в циклі. Якщо твердження If недійсне, просто думайте про оператор while як про "'continue ~".

Keyboard.py

```
import keyboard

while not keyboard.is_pressed("esc"):
    if keyboard.is_pressed("up"):
        print("PingPong!")
    else:
        continue
```

Структура

PingPong буде відображено, коли користувач натискає кнопку стрілки вгору.

Якщо натиснути клавішу зі стрілкою вгору і не натискати клавішу Esc, PingPong буде відображатись безперервно.

Якщо ви провели достатню кількість тестів, натисніть клавішу esc, щоб вийти.

Створення структури

Тепер ви дізналися про основні структури для управління машиною!

Давайте створимо структуру, застосувавши ці знання.

**Натисніть стрілку вгору, щоб
рухатися вперед, натисніть стрілку
вліво, щоб повернути ліворуч,
натисніть стрілку вправо, щоб
повернути праворуч,
Натисніть стрілку назад, щоб повернути назад,**

Щоб зупинитись, натисніть пробіл.

Структура

Autocar.py

```
from pingpongthread import PingPongThread
import keyboard

PingPongThreadInstance =
PingPongThread(number=2)
PingPongThreadInstance.start()
PingPongThreadInstance.wait_until_full_connect()
    while not keyboard.is_pressed("esc"):
        if keyboard.is_pressed("up"):
            PingPongThreadInstance.run_motor([1, 2], [-30, 30])
        elif keyboard.is_pressed("down"):
            PingPongThreadInstance.run_motor([1, 2], [30, -30])
        elif keyboard.is_pressed("left"):
            PingPongThreadInstance.run_motor([1, 2], [15 ,
            15])
        elif keyboard.is_pressed("right"):
            PingPongThreadInstance.run_motor([1, 2], [-15, -15])
        elif keyboard.is_pressed("space"):
            PingPongThreadInstance.run_motor([1, 2], "stop")
```

Якщо ви запустите програму, ви побачите, як машина рухається відповідно до напрямку стрілки, яку ви натискаєте! Пробіл - зупинка.

Структура

```
        else:
            continue
```

Відключення

Ось ми і завершили. Що ж після того, як ми все зробили? Переконайтесь, що куби повністю відключені, перед виходом з терміналу Python.

Відключення

Autocar.py

```
from pingpongthread import PingPongThread
import keyboard

PingPongThreadInstance = PingPongThread(number=2)
PingPongThreadInstance.start()
PingPongThreadInstance.wait_until_full_connect()
    while not keyboard.is_pressed("esc"):
        if keyboard.is_pressed("up"):
            PingPongThreadInstance.run_motor([1, 2], [-30, 30])
        elif keyboard.is_pressed("down"):
            PingPongThreadInstance.run_motor([1, 2], [30, -30])
        elif keyboard.is_pressed("left"):
            PingPongThreadInstance.run_motor([1, 2], [15 , 15])
        elif keyboard.is_pressed("right"):
            PingPongThreadInstance.run_motor([1, 2], [-15, -
            15])
        elif keyboard.is_pressed("space"):
            PingPongThreadInstance.run_motor([1, 2], "stop")
        else:
```

Відключення

Наступний слайд

Відключення

Autocar.py

```
PingPongThreadInstance.run_motor([1, 2], "stop")
PingPongThreadInstance.end()
```

Після зупинки двигуна і перед відключенням, куб від'єднується за допомогою команди End.

Якщо ви натиснете клавішу "esc" після закінчення управління машиною, відбудеться вихід з оператора while та виконання команд вище.

Управляйте Pingpong за допомогою Python!

Ось так ми скористались Python для управління PingPong-машиною! Насправді існує багато інших методів, окрім описаних вище. Наведений вище код - лише один із них.
Ви можете створити свій власний код, використовуючи безліч різних модулів та інструментів!