

API & Python 實作

API

Application Programming Interface

應用程式介面

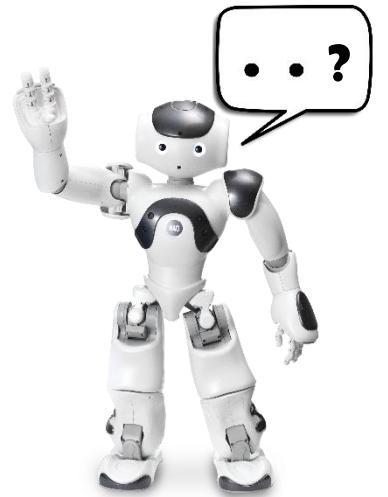
API 是什麼？

大部分是由系統廠商

為了使第三方的開發者可以額外開發應用程式
進而來強化使用廠商的產品，所推出可以與他們系統溝通的介面

如果從介面二字去思考，會有點難理解其概念

換句話說，**API** 是廠商開發出可以讓使用者使用的一個函式庫
而使用者可呼叫使用者想要使用的函式，並給予相對應所需的參數
函式便回傳給使用者結果，流程便是如此



API 是什麼？



影片來源 [API 到底是什麼？用白話文帶你認識](#)

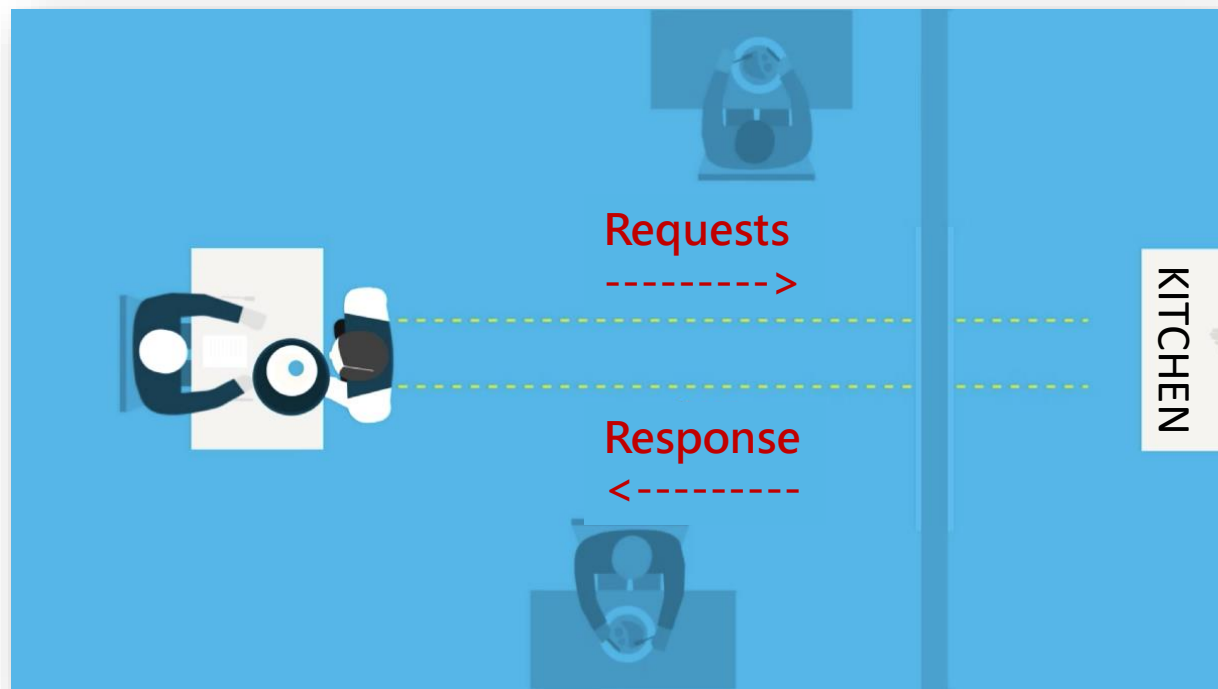
API 是什麼？

即使完全是從頭開始寫的程式
一個漂亮的應用程式也會有供內部使用的 **API** 來幫助組織你的程式碼
除了方便維護程式碼，也能讓特定的功能重複使用性提高，增加效率

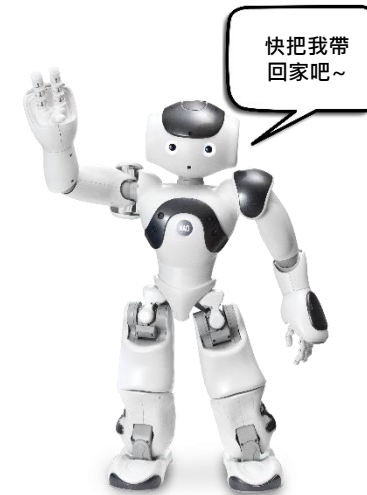
API 是什麼？

最重要的是
使用 API 的過程時
你不需要知道其內部程式運作的
邏輯或演算法

你只要告訴 API 它需要知道的事
它就會把你想知道的結果帶給你



再看一次 API 是什麼？



服務生

大部分是由系統廠商
為了使第三方的開發者可以額外開發應用程式
進而來強化使用廠商的產品，所推出可以與他們系統溝通的介面
如果從介面二字去思考，會有點難理解其概念

廚房

換句話說，**API** 是廠商開發出可以讓使用者使用的一個**函式庫**
而**使用者**可呼叫使用者**想要使用的函式**，並給予相對應所需的**參數**
函式便回傳給使用者**結果**，流程便是如此

顧客

想吃的餐點

上菜囉

餐點名稱

Choregraphe的API在哪裡？

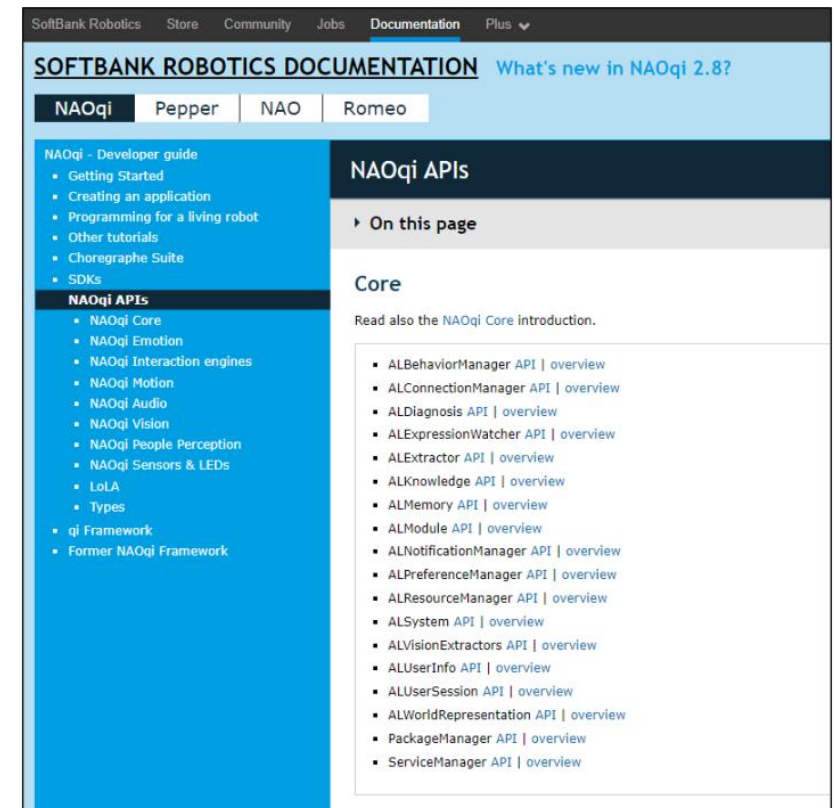
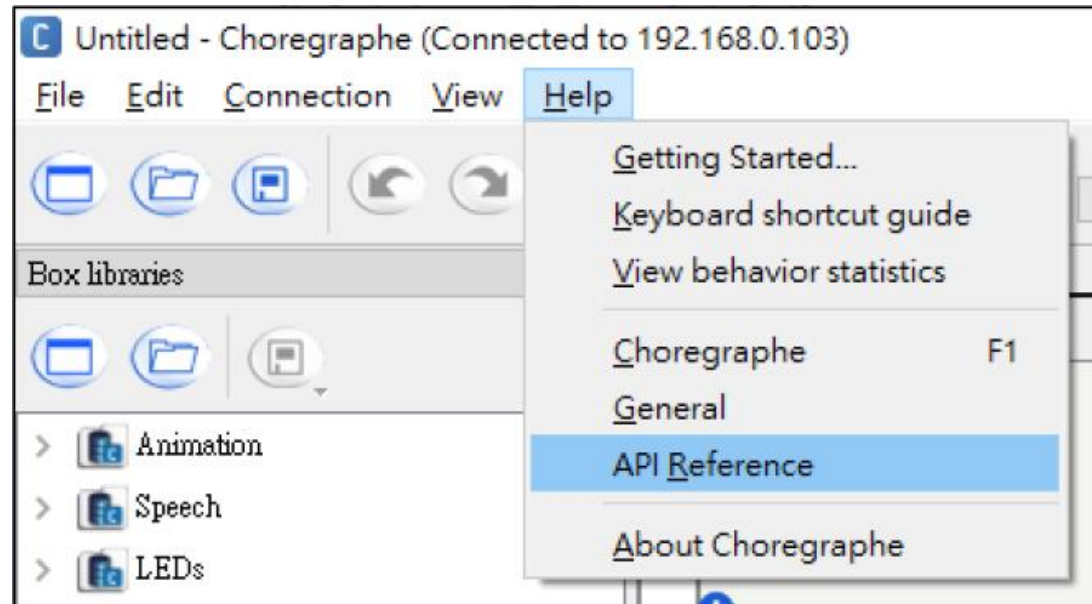
[Help] 選 單



選[API Reference]



[SOFTBANK ROBOTICS
DOCUMENTATION]網頁

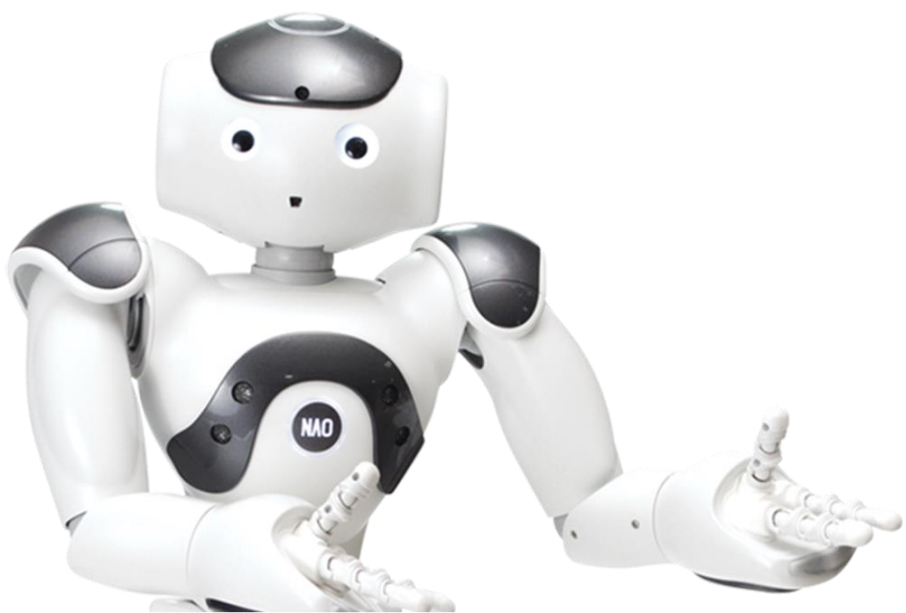


Let's TRY IT ...?

修但幾咧
hold up



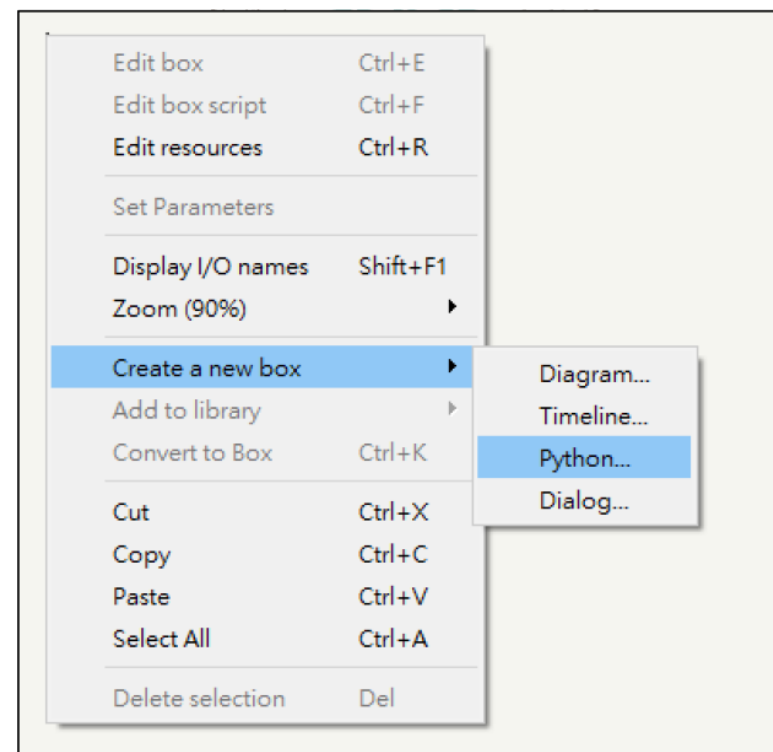
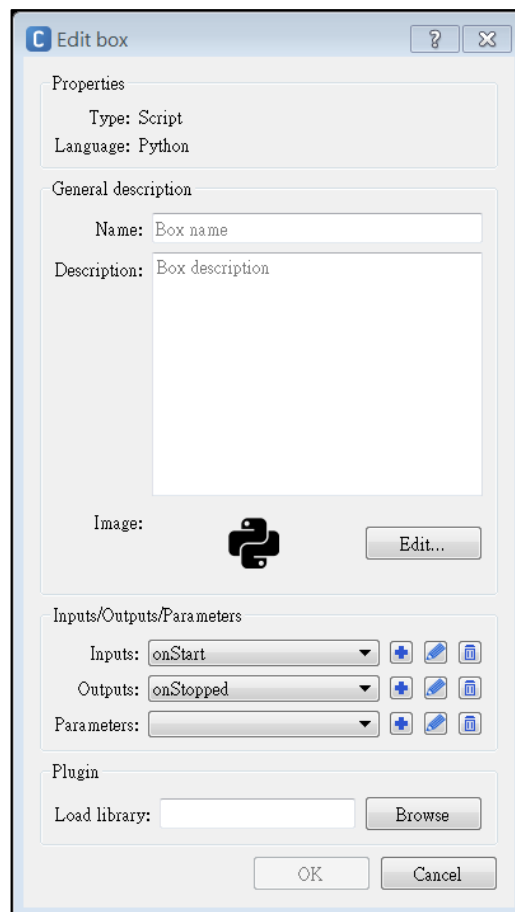
開始前先認識 Python Box



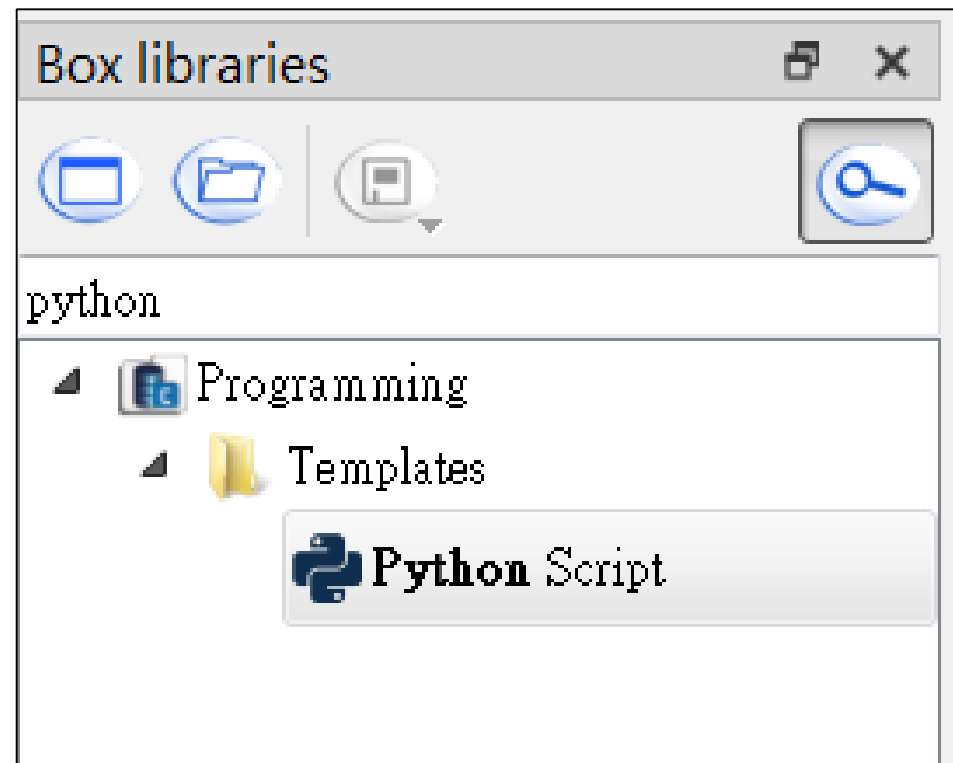
工作區點選右鍵
[Create a new box]

選擇[Python]

命名指令盒和與設置參數→完成

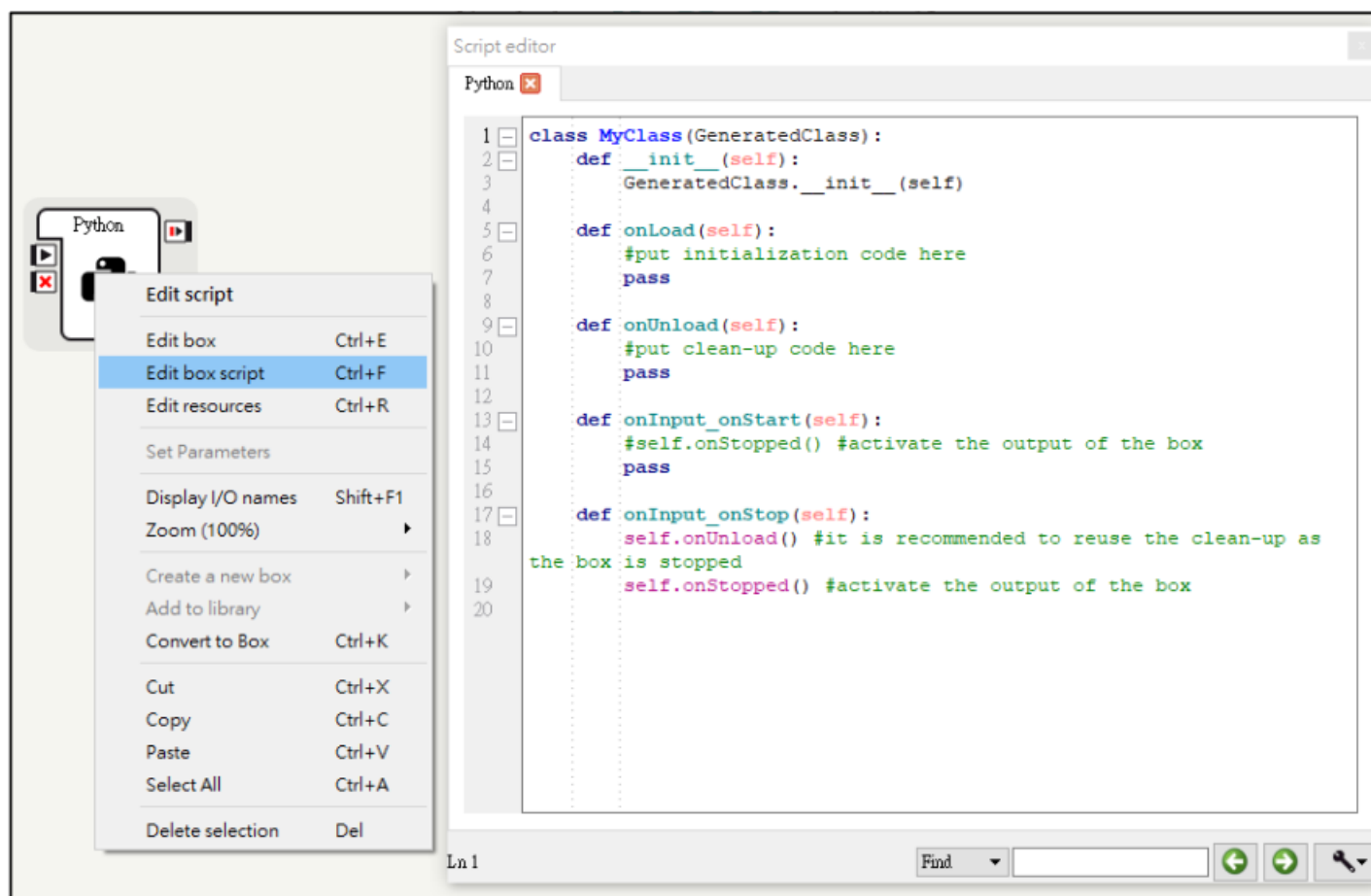


當然 也可以直接在
Box libraries
搜尋Python Script



Python程式碼在哪編輯呢？

- 指令盒上點選右鍵[Edit box script]
- 指令盒上快點兩下



Ok !
Let's TRY?

修但幾咧
hold up



修但幾咧
hold up



修但幾咧
hold up



修但幾咧
hold up



修但幾咧
hold up



API 介紹



ALTextToSpeech API介紹

ALTextToSpeech 的 API → 表示與說話相關的函式庫

SOFTBANK ROBOTICS DOCUMENTATION What's new in NAOqi 2.8?

NAOqi | Pepper | NAO | Romeo

NAOqi - Developer guide

- Getting Started
- Creating an application
- Programming for a living robot
- Other tutorials
- Choregraphe Suite
- SDKs
- NAOqi APIs
 - NAOqi Core
 - NAOqi Emotion
 - NAOqi Interaction engines
 - NAOqi Motion
 - NAOqi Audio
 - ALAnimatedSpeech
 - ALAudioDevice
 - ALAudioPlayer
 - ALAudioRecorder
 - ALSoundDetection
 - ALSoundLocalization
 - ALSpeechRecognition
 - ALTextToSpeech

ALTextToSpeech API

- ALTextToSpeech Tutorial
- ALVoiceEmotionAnalysis
- ALAudioSourceLocalization
- NAOqi Vision
- NAOqi People Perception
- NAOqi Sensors & LEDs
- LoLA
- Types
- qi Framework
- Former NAOqi Framework

ALTextToSpeech API

On this page

NAOqi Audio - Overview | API | Tutorial

Namespace : AL

```
#include <alproxies/altexttospeechproxy.h>
```

Method list

As any module, this module inherits methods from [ALModule API](#). It also has the following specific methods:

class ALTextToSpeechProxy

- ALTextToSpeechProxy::addToDictionary
- ALTextToSpeechProxy::deleteFromDictionary
- ALTextToSpeechProxy::getAvailableLanguages
- ALTextToSpeechProxy::getAvailableVoices
- ALTextToSpeechProxy::getLanguage
- ALTextToSpeechProxy::getParameter
- ALTextToSpeechProxy::getSupportedLanguages
- ALTextToSpeechProxy::getVoice
- ALTextToSpeechProxy::getVolume
- ALTextToSpeechProxy::getLocale
- ALTextToSpeechProxy::resetSpeed
- ALTextToSpeechProxy::say
- ALTextToSpeechProxy::sayToFile
- ALTextToSpeechProxy::setLanguage
- ALTextToSpeechProxy::setLanguageDefaultVoice

左側選單
[NAOqi APIs]

[NAOqi Audio]

[ALTextToSpeech]

[ALTextToSpeech API]

ALTextToSpeech使用方法

匯入 ALTextToSpeech 使用



在 `def onInput_onStart(self):`內
加入 `ttsProxy = ALProxy("ALTextToSpeech")`
建立 `ttsProxy` 物件→以便後續呼叫使用

```
def onInput_onStart(self):  
  
    #self.onStopped() #activate the output of the box  
  
    ttsProxy=ALProxy("ALTextToSpeech") //15 將 ALTextToSpeech 之 API 建立  
    為 ttsProxy 物件  
  
    pass
```


Script editor

Python 

```
1 class MyClass(GeneratedClass):
2     def __init__(self):
3         GeneratedClass.__init__(self)
4
5     def onLoad(self):
6         #put initialization code here
7         pass
8
9     def onUnload(self):
10        #put clean-up code here
11        pass
12
13    def onInput_onStart(self):
14        #self.onStopped() #activate the output of the box
15        ttsProxy=ALProxy("ALTextToSpeech")
16
17
18    def onInput_onStop(self):
19        self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
20        self.onStopped() #activate the output of the box
```

Let's TRY IT !

用Python使Nao說話

參數內容 `ALTextToSpeechProxy::say(const std::string& stringToSay)`
使Nao 說話之函式，第一個參數值表示：說話的內容

任務

讓 NAO 說" Hello World"

Let's TRY IT !

任務

讓 NAO 說" Hello World"

```
def onInput_onStart(self):
```

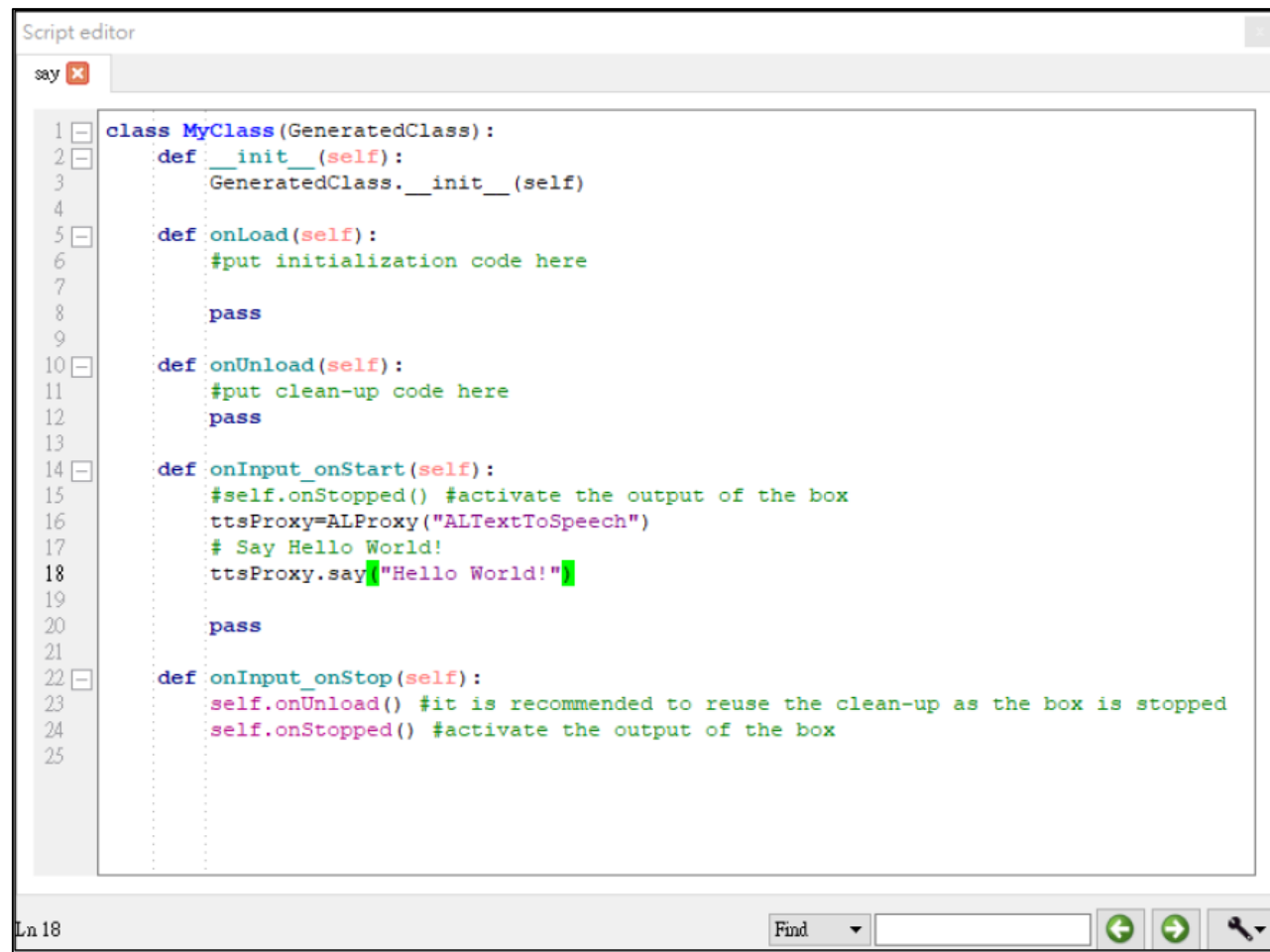
```
    #self.onStopped() #activate the output of the box
```

```
    ttsProxy=ALProxy("ALTextToSpeech") //16 建立物件
```

```
    # Say Hello World!
```

```
    ttsProxy.say("Hello World!") //18 說出 "Hello World!"
```

```
    pass
```



The screenshot shows a 'Script editor' window with a tab labeled 'say'. The code is a Python class named 'MyClass' that inherits from 'GeneratedClass'. It includes several methods: '__init__', 'onLoad', 'onUnload', 'onInput_onStart', and 'onInput_onStop'. The 'onInput_onStart' method is the one being edited, containing comments and code to initialize an 'ALTextToSpeech' proxy and say 'Hello World!'. The 'onInput_onStop' method calls 'onUnload' and 'onStopped'. The status bar at the bottom indicates 'Ln 18'.

```
Script editor
say
1 class MyClass(GeneratedClass):
2     def __init__(self):
3         GeneratedClass.__init__(self)
4
5     def onLoad(self):
6         #put initialization code here
7
8         pass
9
10    def onUnload(self):
11        #put clean-up code here
12        pass
13
14    def onInput_onStart(self):
15        #self.onStopped() #activate the output of the box
16        ttsProxy=ALProxy("ALTextToSpeech")
17        # Say Hello World!
18        ttsProxy.say("Hello World!")
19
20        pass
21
22    def onInput_onStop(self):
23        self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
24        self.onStopped() #activate the output of the box
25
Ln 18 Find
```

Let's TRY IT !

用Python設定語言使Nao說話

參數內容

ALTextToSpeechProxy::setLanguage(const std::string& language)

設定語言之函式，第一個參數值表示：所設定語言

任務

讓 NAO 用中文說“ 你好 世界”

Let's TRY IT !

任務

讓 NAO 用中文說“ 你好 世界”

```
def onInput_onStart(self):  
  
    #self.onStopped() #activate the output of the box  
  
    ttsProxy=ALProxy("ALTextToSpeech") //12  
  
    # Set Language <Chinese>  
  
    ttsProxy.setLanguage("Chinese") //14 設定為中文  
  
    # Say Hello World!  
  
    ttsProxy.say("你好，世界!") //16  
  
    pass
```

Script editor

Say x

```
1 class MyClass(GeneratedClass):  
2     def __init__(self):  
3         GeneratedClass.__init__(self)  
4     def onLoad(self):  
5         #put initialization code here  
6         pass  
7     def onUnload(self):  
8         #put clean-up code here  
9         pass  
10    def onInput_onStart(self):  
11        #self.onStopped() #activate the output of the box  
12        ttsProxy=ALProxy("ALTextToSpeech")  
13        # Set Language <Chinese>  
14        ttsProxy.setLanguage("Chinese") #設定為中文  
15        # Say Hello World!  
16        ttsProxy.say("你好，世界!")  
17        pass  
18    def onInput_onStop(self):  
19        self.onUnload() #it is recommended to reuse the clean-up as the box is stopped  
20        self.onStopped() #activate the output of the box
```

API 介紹



ALMotion API介紹

ALMotion 的 API → 表示與動作相關的函式庫

SoftBank Robotics | Store | Community | Jobs | Documentation | Plus

SOFTBANK ROBOTICS DOCUMENTATION | What's new in NAOqi 2.8?

NAOqi | Pepper | NAO | Romeo

NAOqi - Developer guide

- Getting Started
- Creating an application
- Programming for a living robot
- Other tutorials
- Choregraphe Suite
- SDKs
- NAOqi APIs
 - NAOqi Core
 - NAOqi Emotion
 - NAOqi Interaction engines
 - NAOqi Motion
 - ALAnimationPlayer
 - ALRobotPosture
 - ALNavigation
 - ALRecharge
 - ALMotion

ALMotion API

- Stiffness control
- Joint control
- Locomotion control
- Cartesian control
- Whole Body control
- Self-collision avoidance
- External-collision avoidance
- Fall manager
- Smart Stiffness

ALMotion API

On this page

Overview | API

Namespace : AL

```
#include <alproxies/alnotificationmanagerproxy.h>
```

ALMotion provides methods that help make the robot move. It contains commands allowing you to manipulate

Method Group list

As any module, this module inherits methods from [ALModule API](#). It also has the following specific methods:

```
class ALMotionProxy
```

Stiffness control API

- ALMotionProxy::wakeUp
- ALMotionProxy::rest
- ALMotionProxy::robotIsWakeUp
- ALMotionProxy::stiffnessInterpolation
- ALMotionProxy::setStiffnesses
- ALMotionProxy::getStiffnesses

左側選單
[NAOqi APIs]

[NAOqi Motion]

[ALMotion]

[ALMotion API]

ALMotion使用方法


匯入 ALMotion 使用



在 `def onInput_onStart(self):`內
加入 `mProxy = ALProxy("ALMotion")` 建立
`mProxy` 物件 → 以便後續呼叫使用

```
def onInput_onStart(self):  
    #self.onStopped() #activate the output of the box  
    mProxy=ALProxy("ALMotion") //15 將 ALMotion 之 API 建立為 mProxy 物  
    件  
    pass
```


Script editor

motion 

```
1 class MyClass(GeneratedClass):
2     def __init__(self):
3         GeneratedClass.__init__(self)
4
5     def onLoad(self):
6         #put initialization code here
7         pass
8
9     def onUnload(self):
10        #put clean-up code here
11        pass
12
13    def onInput_onStart(self):
14        #self.onStopped() #activate the output of the box
15        mProxy=ALProxy("ALMotion") #將ALMotion之API建立為mProxy物件
16        pass
17
18    def onInput_onStop(self):
19        self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
20        self.onStopped() #activate the output of the box
21
```

Let's TRY IT !

用Python喚醒Nao

參數內容 **ALMotionProxy::wakeUp()**
使 Nao 機器人呈現喚醒姿態之函式

任務

喚醒Nao機器人

Let's TRY IT !

任務

喚醒Nao機器人

```
def onInput_onStart(self):
```

```
    #self.onStopped() #activate the output of the box
```

```
    mProxy=ALProxy("ALMotion") //15 將 ALMotion 之 API 建立為 mProxy 物
```

```
    mProxy.wakeUp() //16 物件呼叫喚醒函式庫
```

```
    pass
```

件

Script editor

motion x

```
1 class MyClass(GeneratedClass):
2     def __init__(self):
3         GeneratedClass.__init__(self)
4
5     def onLoad(self):
6         #put initialization code here
7         pass
8
9     def onUnload(self):
10        #put clean-up code here
11        pass
12
13    def onInput_onStart(self):
14        #self.onStopped() #activate the output of the box
15        mProxy=ALProxy("ALMotion") #將ALMotion之API建立為mProxy物件
16        mProxy.wakeUp() #物件呼叫喚醒函式庫
17        pass
18
19    def onInput_onStop(self):
20        self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
21        self.onStopped() #activate the output of the box
```

Let's TRY IT !

用Python使Nao移動

參數內容

ALMotionProxy::moveTo(const float& x, const float& y, const float&theta)

移動之函式，第一個參數值表示：x 軸方向移動距離(單位公尺)

第二個參數值表示：y 軸方向移動距離(單位公尺)

第三個參數值表示：旋轉角度(其範圍值 $-3.14 \sim 3.14$ ，單位為徑度)

任務

使 Nao 機器人先向前走 10 公分
再向左走 10 公分
最後逆時針旋轉 1 徑度

Let's TRY IT !

任務

使 Nao 機器人先向前走 10 公分
再向左走 10 公分
最後逆時針旋轉 1 徑度

```
def onInput_onStart(self):  
  
    #self.onStopped() #activate the output of the box  
  
    mProxy=ALProxy("ALMotion") //15 將 ALMotion API 建立為 mProxy 物件  
  
    mProxy.wakeUp() //16 物件呼叫喚醒函式庫  
  
    mProxy.moveTo(0.1,0,0) //17 向前走 0.1 公尺  
  
    mProxy.moveTo(0,0.1,0) //18 向左走 0.1 公尺  
  
    mProxy.moveTo(0,0,1) //19 逆時針旋轉 1 徑度  
  
    pass
```

Let's TRY IT !

任務

使 Nao 機器人先向前走 10 公分
再向左走 10 公分
最後逆時針旋轉 1 徑度

```
Script editor
motion x
1 class MyClass(GeneratedClass):
2     def __init__(self):
3         GeneratedClass.__init__(self)
4
5     def onLoad(self):
6         #put initialization code here
7         pass
8
9     def onUnload(self):
10        #put clean-up code here
11        pass
12
13    def onInput_onStart(self):
14        #self.onStopped() #activate the output of the box
15        mProxy=ALProxy("ALMotion") #將ALMotion之API建立為mProxy物件
16        mProxy.wakeup() #物件呼叫喚醒函式庫
17        mProxy.moveTo(0.1,0,0) #向前走0.1公尺
18        mProxy.moveTo(0,0.1,0) #向左走0.1公尺
19        mProxy.moveTo(0,0,1) #逆時針旋轉1徑度
20        pass
21
22    def onInput_onStop(self):
23        self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
24        self.onStopped() #activate the output of the box
```

API 介紹



綜合應用

ARE YOU READY ?



沒錯~只有打勾的選項~



Let's TRY IT !

Nao 向前行走
0.2 公尺結束後
說
"I just walked"

Let's TRY IT !

任務

Nao 向前行走 0.2 公尺結束後
說 "I just walked"

```
def onInput_onStart(self):  
  
    #self.onStopped() #activate the output of the box  
  
    ttsProxy=ALProxy("ALTextToSpeech") //15 將 ALTextToSpeech 之 API 建立  
    為 ttsProxy 物件  
  
    mProxy=ALProxy("ALMotion") //16 將 ALMotion 之 API 建立為 mProxy 物  
    件  
  
    mProxy.wakeup() //17 Nao 呈現喚醒姿態  
  
    mProxy.moveTo(0.2,0,0) //18 Nao 向前行走 0.2 公尺  
  
    ttsProxy.say("I just walked") //19 Nao 說 "I just walked"  
  
    mProxy.rest() //20 Nao 呈現休息姿態  
  
    pass
```

Let's TRY IT !

任務

Nao 向前行走 0.2 公尺結束後
說 "I just walked"

```
Script editor
My_Python x
1 class MyClass(GeneratedClass):
2     def __init__(self):
3         GeneratedClass.__init__(self)
4
5     def onLoad(self):
6         #put initialization code here
7         pass
8
9     def onUnload(self):
10        #put clean-up code here
11        pass
12
13    def onInput_onStart(self):
14        #self.onStopped() #activate the output of the box
15        ttsProxy=ALProxy("ALTextToSpeech")
16        mProxy=ALProxy("ALMotion")
17        mProxy.wakeUp()
18        mProxy.moveTo(0.2,0,0)
19        ttsProxy.say("I just walked")
20        mProxy.rest()
21        pass
22
23    def onInput_onStop(self):
24        self.onUnload() #it is recommended to reuse the clean-up as the box is stopped
25        self.onStopped() #activate the output of the box
```

**Thank you for your
attention.**