**thread**

1. 定義訊號
   1. EX:

signal\_00 = pyqtSignal()

signal\_01 = pyqtSignal(int)

signal\_02 = pyqtSignal(int, str)

signal\_03 = pyqtSignal(list)

signal\_04 = pyqtSignal(dict)

定義無參數訊號

定義int參數訊號

定義int, str參數訊號

定義list參數訊號

定義dict參數訊號

* 1. continue

1. 連結訊號與槽函數(signal connect to slot function)
   1. EX:

self.signal\_00.connect(self.signal\_call\_00)

self.signal\_01.connect(self.signal\_call\_01)

self.signal\_02.connect(self.signal\_call\_02)

self.signal\_03.connect(self.signal\_call\_03)

self.signal\_04.connect(self.signal\_call\_04)

訊號連接到signal\_call\_00

訊號連接到signal\_call\_01

訊號連接到signal\_call\_02

訊號連接到signal\_call\_03

訊號連接到signal\_call\_04

* 1. 訊號與槽函數連結，可以發訊號後找到對應的槽函數去執行。

1. 發射訊號
   1. EX:

self.signal\_00.emit()

self.signal\_01.emit(1)

self.signal\_02.emit(1, “text”)

self.signal\_03.emit([1, 2, 3])

self.signal\_04.emit([“name”: ”gary”, “age”: ”40”]

訊號連接到signal\_call\_00

訊號連接到signal\_call\_01

訊號連接到signal\_call\_02

訊號連接到signal\_call\_03

訊號連接到signal\_call\_04

* 1. 發射訊號後才能夠真正的執行槽函數裡面的內容。

1. 槽函數的定義
   1. EX:

def signal\_call\_00(self):

print(“enter signal 00 func”)

def signal\_call\_01(self, val):

print(“enter signal 01 func”, val)

定義槽函數signal\_call\_00

定義槽函數signal\_call\_01

* 1. continue

1. 執行strar()函式，啟動執行緒,則run()函式才會開始跑．
2. run()函數結束則執行緒結束, 也就是start()結束。
3. Example\_00:
4. 建立一個新的執行緒
   1. EX:

class Thread(QThread):

def \_\_init\_\_(self):

super().\_\_init\_\_()

def run(self):

# 執行緒相關程式碼

pass

# 建立一個新的執行緒

thread = Thread()

thread.start()

定義類別名稱 Thread

定義建構

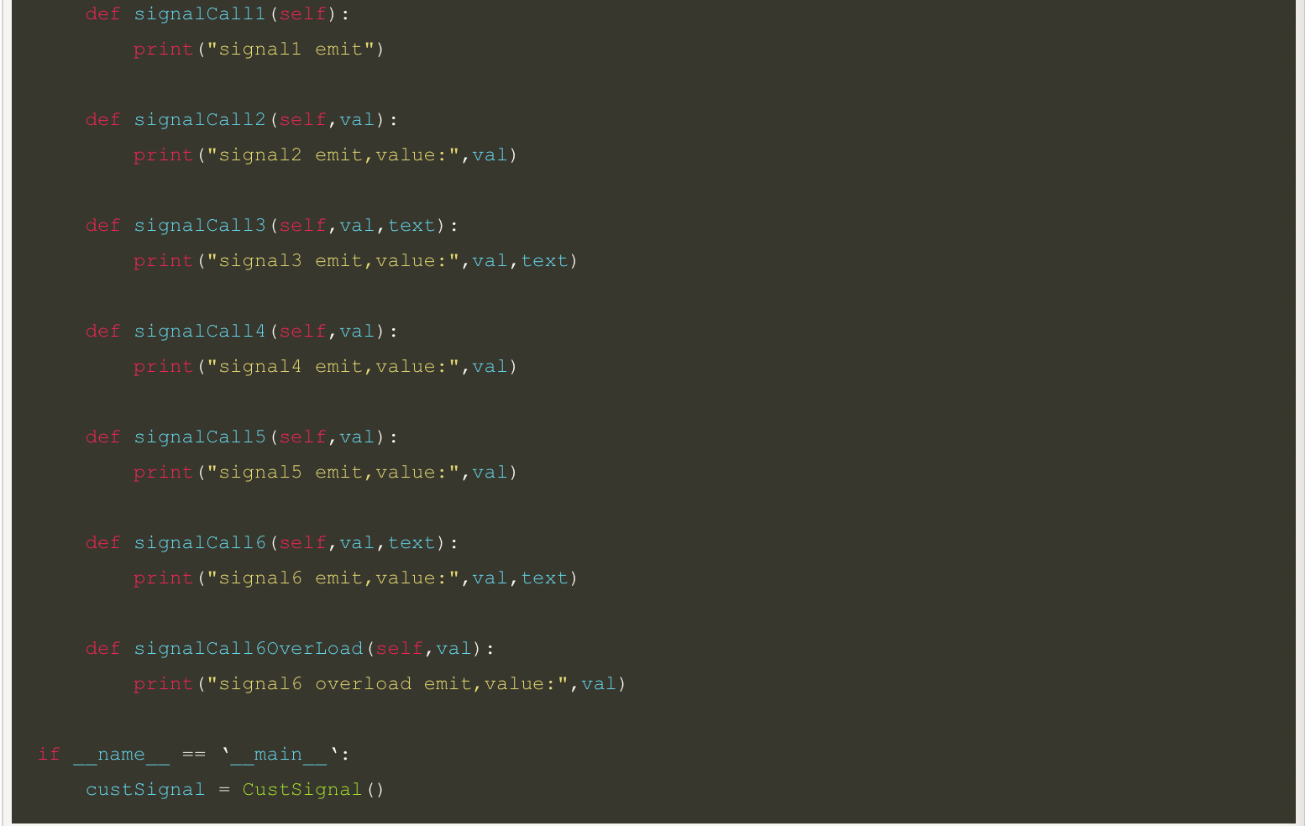
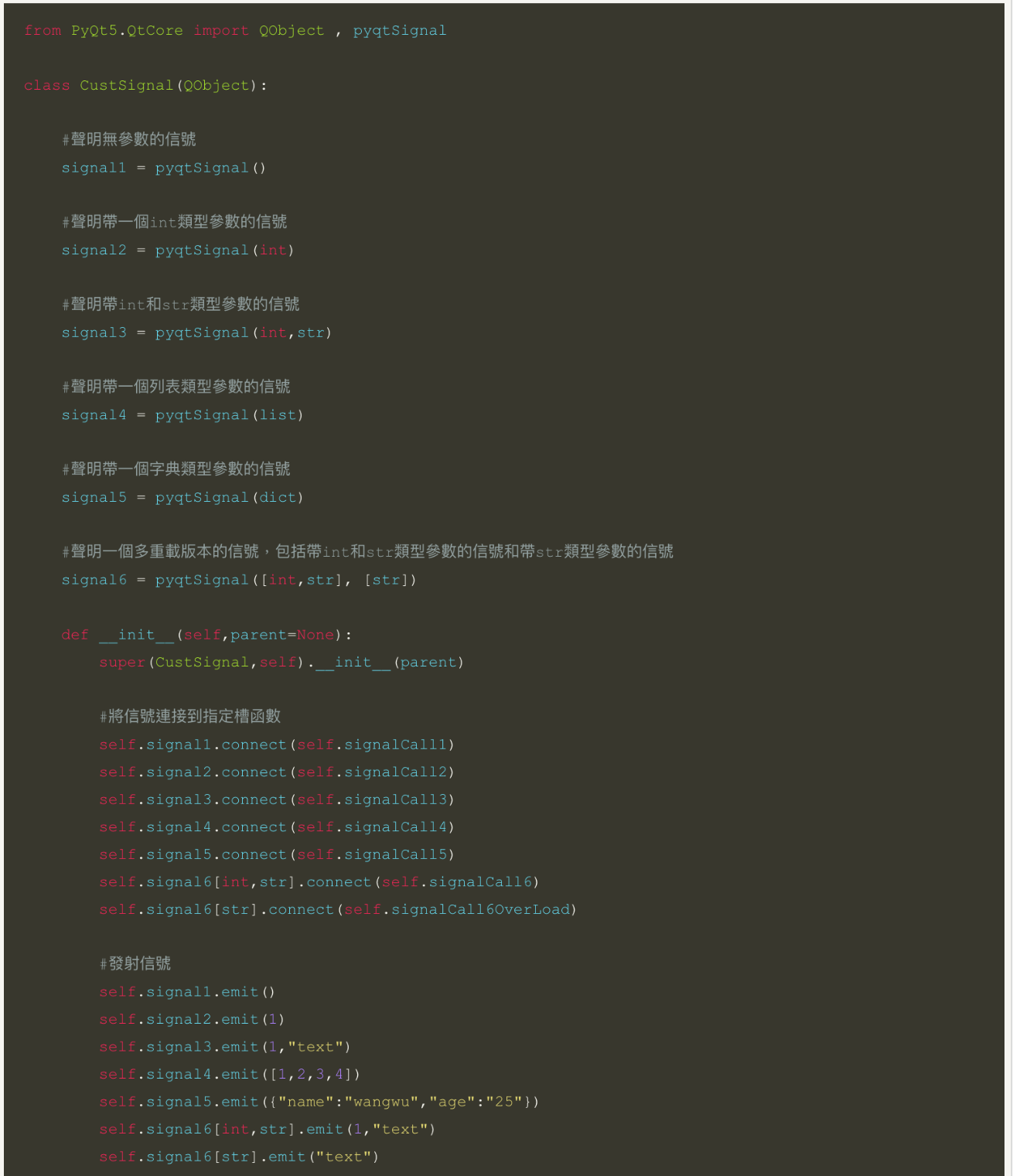
定義欲執行的程式碼

宣告類別變數

開始執行run()

* 1. 說明: 使用執行緒時可以直接得到Thread實例，呼叫其start()函式即可啟動執行緒，執行緒啟動後，會呼叫其實現的run方法，該方法就是執行緒的執行函式，當run()退出之後執行緒基本就結束了。

1. Example\_01:



1. continue