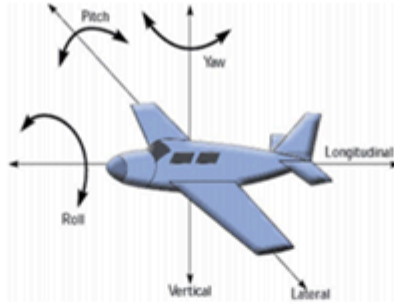


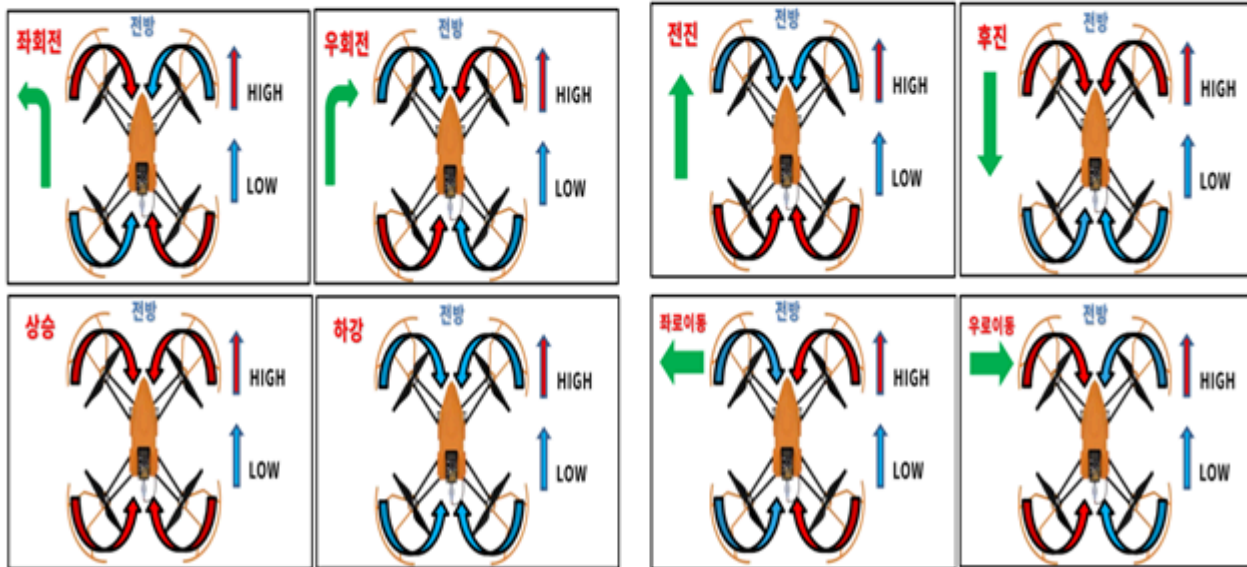
## Tello Drone Python 개발환경 구축

### ✓ 드론의 비행원리

- 요(yaw), 피치(pitch), 롤(roll)에 대해 배워보겠습니다. 요, 피치, 롤은 드론의 회전과 관계된 역학의 기본 구성 요소
  - 각각  $x, y, z$ 축 회전을 담당하고 있지요. 요, 피치, 롤을 통해서 드론은 앞, 뒤, 좌, 우 이동과 회전을 할 수 있음
  - 스로틀(throttle)은 모터의 출력을 결정하여 드론을 위, 아래로 움직일 수 있게 함



요, 피치, 롤



## 준비물

- PC용 와이파이 모듈
- DJI\_SDK - <https://github.com/dji-sdk/Tello-Python>

- 개발자에게 [다른 프로그램에 추가하거나 연결할 수 있는 커스텀 앱을 제작할 수 있는 기능을](#) 제공하는 도구 모음
- 

## ✓ [DJI 로보마스터 TT 텔로 텔런트 파이썬 제어](#)

---

## ✓ [실습 전 준비 - 파이썬 텔로 드론 제어 프로그램 설치](#)

- 1. [파이썬 과학툴 패키지 Anaconda 설치](#) - [아나콘다 설치](#) ([https://www.anaconda.com/products/individual-d\\_](https://www.anaconda.com/products/individual-d_))
- 2. [파이썬 통합개발 환경 Pycharm 설치](#) - [파이참 설치](#) (<https://www.jetbrains.com/ko-kr/pycharm/download/#section=windows>)
- 3. [파이썬 IDE](#)
- 4. 텔로/텔로 에듀/텔로 텔런트 로보마스터 준비
  - 텔로는 DJI의 기술이 들어간 RYZE ROBOTICS 사의 미니드론
  - 기본적으로 스마트폰으로 조종을 하며, 조종기로도 조종할 수 있음
  - 텔로드론은 [스크래치만 사용가능](#)
  - [텔로 에듀 드론은 스크래치 / 파이썬의 두가지 언어가 모두 사용가능](#)



 텔로	기체 구분	 텔로에듀
80g (프로펠러 및 배터리 포함)	무게	87g (프로펠러 및 배터리 포함)
Scratch	호환 코딩 프로그램	Scratch Python Swift
안됨	군집 비행	지원
Tello DroneBlocks	사용 앱	Tello Edu DroneBlocks
기체 x 1 배터리 x 1 프로펠러 2대 분량 프로펠러 가드 1대 분량	구성품	기체 x 1 배터리 x 1 프로펠러 2대 분량 프로펠러 가드 1대 분량 usb 충전 케이블 x 1 패드 x 4

---

## ✓ 드론 연결 순서

- 1. 드론의 전원을 작동시킨다
  - 2. Tello 드론을 와이파이에 연결한다
  - 3. 드론을 작동시킨다
- 

## ✓ Tello Python3 Control Demo

---

- C:/telloedu 폴더를 생성한 후 다음 사이트에서 Tello3.py를 다운로드한 후 복사
    - <https://dl-cdn.ryzerobotics.com/downloads/tello/20180222/Tello3.py>
- 

## ✓ Tello3.py 파일

```
1 # Tello Python3 Control Demo
2
3 import threading #스레딩을 사용하기 위해서는 'threading' 모듈을 import
```

```
4 import socket #소켓 통신을 하기 위해 socket이라는 모듈을 import
5 import sys #sys 라이브러리는 파이썬 인터프리터를 제어하는데 사용되는 기본 모듈
6 import time #시간과 관련된 time, sleep, localtime, strftime 등 다양한 함수를 제공하여 시간 측정, 딜레이, 시간 형식 변환 등에 사용
7
8
9 host = ''
10 port = 9000
11 locaddr = (host,port)
12
13 #Create a UDP socket
14 #AF_INET은 소켓이 통신할 수 있는 주소 유형(이 경우 Internet Protocol v4 주소)을 지정하는 데 사용되는 주소 체계
15 #SOCK_DGRAM을 전달하면 '비 연결 지향형 소켓'이 생성
16 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
17
18 tello_address = ('192.168.10.1', 8889)
19
20 sock.bind(locaddr)
21
22 def recv():
23     count = 0
24     while True:
25         try:
26             data, server = sock.recvfrom(1518) #UDP 소켓에서 데이터를 수신
27             print(data.decode(encoding="utf-8")) #바이트코드(가상머신이 이해할 수 있는 중간 코드로 컴파일한 것)를 유니코드로 변환
28         except Exception:
29             print('WnExit . . .Wn')
30             break
31
32
33 print ('WrWnWrWnTello Python3 Demo.WrWn')
34
35 print ('Tello: command takeoff land flip forward back left right WrWn          up down cw ccw speed speed?WrWn')
36
37 print ('end -- quit demo.WrWn')
38
39
40 #recvThread create - Recv 스레드 생성
```

```

41 #Thread() : 스레드 생성 함수
42 #Thread(target=함수명, args=(매개변수))
43 #소켓에 연결된 각각의 클라이언트의 메시지를 받을 스레드
44
45 recvThread = threading.Thread(target=recv)
46 recvThread.start() #소켓 프로그래밍에서 데이터 수신과 스레드를 생성하는 데 사용
47
48 while True:
49
50     try:
51         msg = input(""); #input("문자열"). input 함수는 사용자로부터 입력을 받는 함수
52
53         # 2.x버전으로 저장하기 위해서는
54         # msg=raw_input("");로 변경후 Tello32.py로 저장
55
56         if not msg:
57             break
58         if 'end' in msg:
59             print ('...')
60             sock.close()
61             break
62
63         # Send data
64         msg = msg.encode(encoding="utf-8") #encoding 파라미터를 지정해서 어떤 인코딩 방식을 사용하는지 명시적으로 지정
65         sent = sock.sendto(msg, tello_address) # 소켓(접속 되었건 접속되지 않았건)에서 데이터를 접속된 상대방으로 전송하는데 사용되는 함수
66     except KeyboardInterrupt:
67         print ('Wn . . .Wn')
68         sock.close()
69         break

```

- 텔로 에듀의 오른쪽 전원 버튼을 눌러 전원을 켜고 [Wifi 접속창에서 텔로 에듀의 SSID에 접속](#)
- [커맨드 창을 열고 파이썬을 실행](#) 또는 [탐색기에서 파일을 직접 더블클릭](#)
- 커맨드 창에서 [SDK 명령어를 키보드로 입력받아 UDP 통신을 통해 신호를 전송하여 기체를 컨트롤](#)



command → SDK 모드진입, Ok 회신

battery? → 배터리 잔량 회신

sn? → 시리얼넘버 회신

sdk? → sdk 버전 회신

wifi? → wifi 강도 회신

takeoff → 이륙, Ok 회신

cw 90 → 90도 우회전 (시계방향), Ok 회신

ccw 90 → 90도 좌회전 (반시계방향), Ok 회신

up 50 → 50cm 상승, Ok 회신

flip f → 전진 플립, Ok 회신

back 50 → 50cm 후진, Ok 회신

land → 착륙, Ok 회신

---

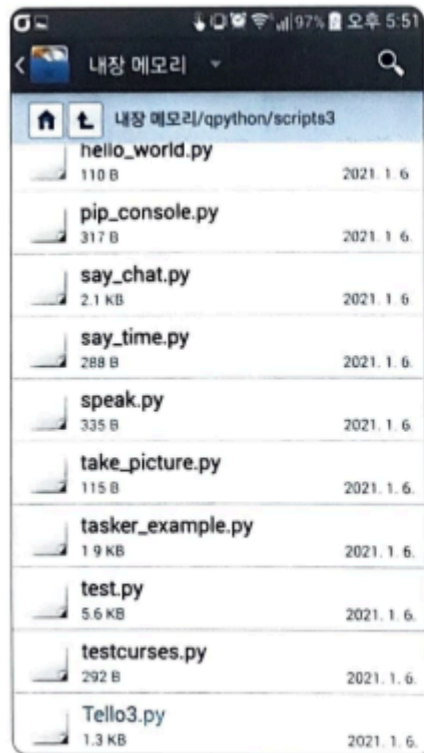
## ✓ Qpython3 앱을 활용한 텔로 에듀 컨트롤

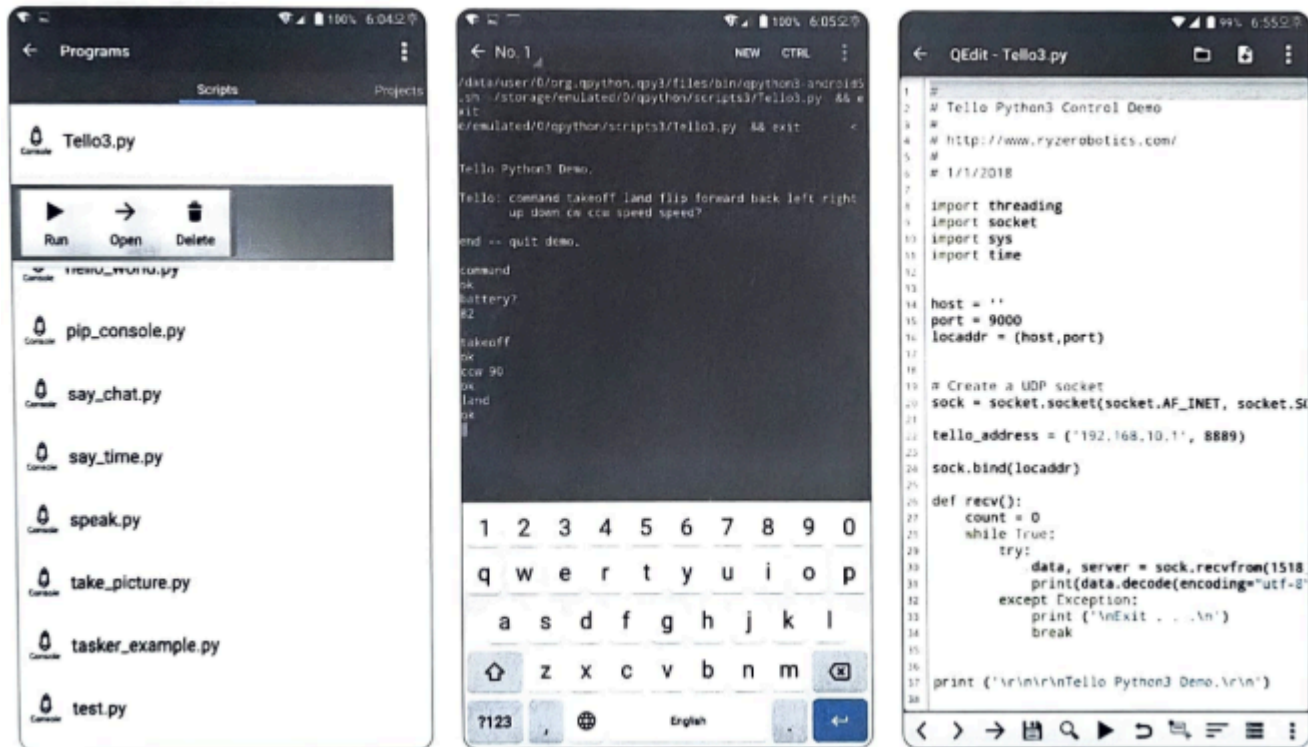


# QPython3 - Python for Android

QPythonLab

광고 포함





## ✓ 단일 텔로 자동비행 컨트롤 예제다운로드

- [Tello-Python-master.zip](#)를 다운로드한 후 [C:/telloedu](#) 폴더에 복사
  - <https://github.com/dji-sdk/Tello-Python>

## ✓ 텔로 에듀의 자동 비행을 테스트하기 전에 기체의 상태정보 표시방법 파악하기

- 파이썬 라이브러리 제공 사이트에서 접속하여 [curses 모듈](#)을 검색하여 [파이썬 2.7용 라이브러리중](#) 자신의 윈도우 버전에 맞는 파일을 선택하여 다운로드하여 [C:/telloedu](#) 폴더에 복사

- curses 모듈은 이식성 있는 고급 터미널 처리를 위한 사실상의 표준인 curses 라이브러리에 대한 인터페이스를 제공
- curses는 유닉스 환경에서 가장 널리 사용되지만, 윈도우, DOS 및 기타 시스템에서도 사용할 수 있는 버전이 있음
- 이 확장 모듈은 리눅스와 유닉스의 BSD 변형에서 동작하는 오픈 소스 curses 라이브러리인 ncurses의 API와 일치하도록 설계되었음

- 주소창에 [cmd](#)를 입력하여 커맨드 창을 열고 파이썬 명령어로 [모듈을 설치](#)

- <https://github.com/gtalarico/curses-win/blob/master/bin/curses-2.2%2Butf8-cp27-cp27m-win32.whl>
- <https://docs.python.org/ko/dev/howto/curses.html>

- [C:/telloedu>python -m pip install curses-2.2.1+utf8-cp27-cp27m-win\\_amd64.whl](#)
- Tello-Pytoh-Master 폴더에서 [tello\\_state.py](#)를 수정 후 실행

- INTERVAL=0.2 -> [INTERVAL=0.05](#)

## ✓ *tello\_state.py* 파일

```

1 #tello_state.py
2
3 import socket
4 from time import sleep
5 import curses # curses 확장 모듈을 사용하여 텍스트 모드 디스플레이를 제어
6 #https://python.flowdas.com/howto/curses.html
7

```

```

8 INTERVAL = 0.2
9 #INTERVAL= 0.05
10
11 def report(str):
12     stdscr.addstr(0, 0, str) #0번째 줄 0번째 열부터 str라는 문자열을 출력하라는 의미
13     stdscr.refresh() #화면을 갱신하기 위해 창 객체의 refresh() 메서드를 호출
14
15 if __name__ == "__main__":
16     stdscr = curses.initscr() #curses로 터미널을 제어하려면 먼저 initscr()을 호출하여 터미널 객체 stdscr을 생성해야함
17     curses.noecho() #키보드 입력값이 화면에 보이지 않도록 설정
18     #응용 프로그램은 또한 일반적으로 Enter 키를 누르지 않아도 즉시 키에 반응해야 함
19     #이것을 일반적인 버퍼 입력 모드와 대비하여 cbreak 모드라고 함
20     curses.cbreak()
21
22     local_ip = ''
23     local_port = 8890
24     socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # socket for sending cmd
25     socket.bind((local_ip, local_port))
26
27     tello_ip = '192.168.10.1'
28     tello_port = 8889
29     tello_adderss = (tello_ip, tello_port)
30
31     socket.sendto('command'.encode('utf-8'), tello_adderss)
32
33     try:
34         index = 0
35         while True:
36             index += 1
37             response, ip = socket.recvfrom(1024)
38             if response == 'ok':
39                 continue
40             out = response.replace(';', '\n')
41             out = 'Tello State:\n' + out
42             report(out)
43             sleep(INTERVAL)
44     except KeyboardInterrupt:

```

```

45 curses.echo() #키보드 입력값이 화면에 출력되도록 설정
46 curses.nocbreak() #cbreak 모드를 해제
47 curses.endwin() # endwin() 함수를 호출하여 터미널을 원래 작동 모드로 복원

```

---

## ✓ PC 설정 및 코딩

- PC에서 WiFi 접근이 가능해야 함
    - Tello의 전원을 넣고 PC의 WiFi를 Tello와 연결시킴(PC에서 와이파이로 Tello 드론을 접속).
  - 파이썬 IDE(Pycharm/vscode 등)을 이용하여 코딩후 실행
- 

## ✓ *tello\_control\_test\_1.py 파일*

```

1 #tello_control_test_1.py - 현재 텔로 드론과 연결 상태 정보 출력
2 #드론과 와이파이 연결
3
4 import socket #소켓 통신을 하기 위해 socket이라는 모듈을 import
5 from time import sleep #time 라이브러리의 sleep 함수를 사용하면 일정 시간동안 프로세스를 일시정지
6
7 #__name__ 변수는 현재 모듈의 이름을 담고 있는 내장 변수
8 #모듈이 직접 실행되었는지(import 되었는지 아닌지) 판단할 때 __name__ 변수의 값을 사용
9 #일반적으로, 모듈은 직접 실행되거나 다른 모듈에서 import 되어 사용됨
10 #만약 모듈이 직접 실행되면, __name__ 변수는 문자열 "__main__"이 할당됨
11 #반대로, 모듈이 import 되어 사용될 때는, __name__ 변수는 해당 모듈의 이름(파일명)이 할당됨
12 #따라서, __name__ 변수의 값을 "__main__"과 비교하면 현재 모듈이 직접 실행되는지(import 되는지)를 판단할 수 있음
13 #따라서 코드를 if __name__ == "__main__"로 감싸면, 해당 파일이 모듈로 사용될 때는 실행되지 않고, 직접 실행될 때만 실행됨

```

```
14
15 if __name__ == "__main__":
16     local_ip = ''
17     local_port = 8890 # 입력 포트 저장
18
19     #파이썬에서 비연결성 UDP 소켓을 생성
20     #socket.AF_INET - IPv4 주소를 의미
21     #socket.SOCK_DGRAM - 비연결성 소켓 유형
22     #socket 모듈 - BSD 소켓 인터페이스에 대한 액세스를 제공
23     #socket() 함수 - 소켓 객체를 반환하고, 이 소켓 객체의 메서드는 다양한 소켓 시스템 호출을 구현
24
25     socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # socket for sending cmd
26     socket.bind((local_ip, local_port)) #소켓 주소 정보 할당
27
28     tello_ip = '192.168.10.1' # Tello는 AP(Access Point)이며 주소는 ip 192.168.10.1을 사용
29
30     #Send Command & Receive Response
31     #명령어는 UDP Port 8889를 통하여 보내줄 수 있으며 첫번째는 'command' 명령을 보내어 SDK 모드가 되도록 함
32     tello_port = 8889 # port 출력(저장)
33     tello_adderss = (tello_ip, tello_port)
34     socket.sendto('command'.encode('utf-8'), tello_adderss) # SDK Mode 진입
35
36     try:
37         index = 0
38         while True:
39             outStr=""
40             response, ip = socket.recvfrom(1024) #클라이언트로부터 데이터 수신(데이터 및 주소 정보 반환)
41             if response == 'ok':
42                 continue
43             outStr = 'Tello State:' + str(response)
44             print(outStr)
45             sleep(0.2)
46     except KeyboardInterrupt:
47         pass # "아무것도 하지 않는" 명령어, 마치 #(코멘트) 처리되어 있는 라인과 같은 효과를 가짐
```

```

IDLE Shell 3.13.1
File Edit Shell Debug Options Window Help
6.00;agz:-1000.00;WrWn'
Tello State:b'mid:-1;x:-100;y:-100;z:-100;mpry:0,0,0;pitch:0;roll:0;yaw:0;vgx:0;
vgy:0;vgz:0;templ:50;temph:53;tof:10;h:0;bat:88;baro:26.87;time:0;agx:8.00;agy:-
6.00;agz:-1001.00;WrWn'
Tello State:b'mid:-1;x:-100;y:-100;z:-100;mpry:0,0,0;pitch:0;roll:0;yaw:0;vgx:0;
vgy:0;vgz:0;templ:50;temph:53;tof:10;h:0;bat:88;baro:26.94;time:0;agx:8.00;agy:-
4.00;agz:-998.00;WrWn'
Tello State:b'mid:-1;x:-100;y:-100;z:-100;mpry:0,0,0;pitch:0;roll:0;yaw:0;vgx:0;
vgy:0;vgz:0;templ:50;temph:53;tof:10;h:0;bat:88;baro:26.88;time:0;agx:8.00;agy:-
6.00;agz:-999.00;WrWn'
Tello State:b'mid:-1;x:-100;y:-100;z:-100;mpry:0,0,0;pitch:0;roll:0;yaw:0;vgx:0;
vgy:0;vgz:0;templ:50;temph:53;tof:10;h:0;bat:88;baro:26.91;time:0;agx:9.00;agy:-

```

- Send Command & Receive Response

- Tello는 AP(Access Point)이며 주소는 ip 192.168.10.1을 사용
- 명령어는 UDP Port 8889를 통하여 보내줄 수 있으며 첫번째는 'command' 명령을 보내어 SDK 모드가 되도록 함

- Receive Tello State

- PC에 UDP Server 0.0.0.0 UDP port: 8890를 설치하여 Tello로 부터 오는 메시지를 받을 수 있음

- Receive Tello Video Stream

- PC에 UDP Server 0.0.0.0 UDP port: 11111를 설치하여 Tello로 부터 오는 비디오를 받을 수 있음
- 비디오 스트림을 받기 위해서는 "streamon" 명령어를 UDP Port 8889로 보내주면 됨



## Control Commands

Command	Description	Possible Response
Command	Enter SDK mode.	ok / error
takeoff	Auto takeoff.	
land	Auto landing.	
streamon	Enable video stream.	
streamoff	Disable video stream.	
emergency	Stop motors immediately.	
up x	Ascend to "x" cm. x = 20-500	
down x	down "x" Descend to "x" cm. x = 20-500	
left x	Fly left for "x" cm. "x" = 20-500	
right x	Fly right for "x" cm. "x" = 20-500	
forward x	Fly forward for "x" cm. "x" = 20-500	
back x	Fly backward for "x" cm. "x" = 20-500	
cw x	Rotate "x" degrees clockwise. "x" = 1-360	

## ✎ SDK 라이브러리 사용

- 소켓 프로그램이 아닌 SDK에서 제공하는 Tello() 클래스를 사용하면 더욱 쉽게 프로그램 개발이 가능
    - 다운 받은 SDK에서 Tello() 클래스를 제공
    - 소켓프로그램을 클래스 안에 내장한 것으로 좀 더 빠르고 편하게 개발할 수 있음
- 

## ✓ Tello() 클래스

```
1 import socket
2 import threading
3 import time
4 from stats import Stats #stats 클래스
5
6 class Tello:
7     #초기화 initialize 메서드 - 객체가 생성될 때, 자동으로 호출되는 메서드
8     #객체의 초기값을 설정해야 할 때 사용
9     #self는 객체의 인스턴스 그 자체를 말하며 객체 자기 자신을 참조하는 매개변수
10    def __init__(self):
11        self.local_ip = ''
12        self.local_port = 8889
13        self.socket = socket.socket(socket.AF_INET, socket.SOCK_DGRAM) # socket for sending cmd
14        self.socket.bind((self.local_ip, self.local_port))
15
16        # thread for receiving cmd ack
17        self.receive_thread = threading.Thread(target=self._receive_thread)
18        self.receive_thread.daemon = True
19        self.receive_thread.start()
20
21        self.tello_ip = '192.168.10.1'
22        self.tello_port = 8889
23        self.tello_adderss = (self.tello_ip, self.tello_port)
24        self.log = []
25
```

```
26     self.MAX_TIME_OUT = 15.0
27
28     def send_command(self, command):
29         """
30         Send a command to the ip address. Will be blocked until
31         the last command receives an 'OK'.
32         If the command fails (either b/c time out or error),
33         will try to resend the command
34         :param command: (str) the command to send
35         :param ip: (str) the ip of Tello
36         :return: The latest command response
37         """
38         self.log.append(Stats(command, len(self.log)))
39
40         self.socket.sendto(command.encode('utf-8'), self.tello_adderss)
41         print('sending command: %s to %s' % (command, self.tello_ip))
42
43         start = time.time()
44         while not self.log[-1].got_response():
45             now = time.time()
46             diff = now - start
47             if diff > self.MAX_TIME_OUT:
48                 print('Max timeout exceeded... command %s') % command
49                 # TODO: is timeout considered failure or next command still get executed
50                 # now, next one got executed
51                 return
52         print('Done!!! sent command: %s to %s' % (command, self.tello_ip))
53
54     def _receive_thread(self):
55         """Listen to responses from the Tello.
56
57         Runs as a thread, sets self.response to whatever the Tello last returned.
58
59         """
60         while True:
61             try:
62                 self.response, ip = self.socket.recvfrom(1024)
```

```
63         print('from %s: %s' % (ip, self.response))
64
65         self.log[-1].add_response(self.response)
66     except (socket.error, exc):
67         print("Caught exception socket.error : %s" % exc)
68
69     def on_close(self):
70         pass
71         # for ip in self.tello_ip_list:
72         #     self.socket.sendto('land'.encode('utf-8'), (ip, 8889))
73         # self.socket.close()
74
75     def get_log(self):
76         return self.log
```

---

## ✓ Stats() 클래스

```
1 from datetime import datetime
2
3 class Stats:
4     def __init__(self, command, id):
5         self.command = command
6         self.response = None
7         self.id = id
8
9         self.start_time = datetime.now()
10        self.end_time = None
11        self.duration = None
12
13    def add_response(self, response):
14        self.response = response
15        self.end_time = datetime.now()
16        self.duration = self.get_duration()
```

```
17         # self.print_stats()
18
19     def get_duration(self):
20         diff = self.end_time - self.start_time
21         return diff.total_seconds()
22
23     def print_stats(self):
24         print('Wnid: %s' % self.id)
25         print('command: %s' % self.command)
26         print('response: %s' % self.response)
27         print('start time: %s' % self.start_time)
28         print('end_time: %s' % self.end_time)
29         print('duration: %sWn' % self.duration)
30
31     def got_response(self):
32         if self.response is None:
33             return False
34         else:
35             return True
36
37     def return_stats(self):
38         str = ''
39         str += 'Wnid: %sWn' % self.id
40         str += 'command: %sWn' % self.command
41         str += 'response: %sWn' % self.response
42         str += 'start time: %sWn' % self.start_time
43         str += 'end_time: %sWn' % self.end_time
44         str += 'duration: %sWn' % self.duration
45         return str
```


---

▼ *tello\_control\_test\_2.py 파일*

```
1 #tello_control_test_2.py - #자동 이착륙 코드
2
3 # 반드시 드론을 멀리 떨어져서 실행할 것
4 # 두 개의 파일 - 폴더 Single_Tello_Test에 있는 stats와 tello를 같은 폴더에 위치하고 실행할 것
5 # command 파일 작성할 것
6
7 # 드론과 와이파이 연결
8
9 from tello import Tello
10 from datetime import datetime
11 import time #time 모듈은 Python에서 시간 관련 작업을 수행하기 위한 표준 라이브러리
12
13 start_time = str(datetime.now())
14 file_name = "command.txt"
15 f = open(file_name, "r")
16 commands = f.readlines() # readlines() : 파일 내 텍스트에서 각 줄을 element로 하는 리스트로 반환
17 tello = Tello()
18
19 for command in commands:
20     if command != '' and command != '\n':
21         command = command.rstrip() #인자로 전달된 문자를 String의 오른쪽에서 제거
22         if command.find('delay') != -1:
23             sec = float(command.partition('delay')[2])
24             print('delay %s' % sec)
25             time.sleep(sec)
26             pass
27         else:
28             tello.send_command(command)
29
30 log = tello.get_log() #get_log method is used to get the log for a given log type
31 outFile = open('log.txt', 'w+')
32 for stat in log:
33     stat.print_stats()
34     str = stat.return_stats()
35     outFile.write(str)
36
37
```

```
38 #command.txt의 내용
39 #takeoff
40 #delay 5
41 #land
```

---

 command - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
command
takeoff
delay 5
land
```

---

## ✓ [DJI TelloPy API Reference](#)

---

## ✓ [기본동작제어](#)

- takeoff/land - takeoff(), land()
- move up/down - move\_up(), move\_down()
- move left/right - move\_left(), move\_right()
- rotate\_CW\_CCW - rotate\_clockwise(), rotate\_counter\_clockwise()
- send\_rc\_control\_async - send\_rc\_control(self, left\_right\_velocity, forward\_backward\_velocity, up\_down\_velocity, yaw\_velocity)

## ✓ 실습(1).

- takeoff -> 3회 up(40) -> down(40) 반복 -> landing
- 

## ✓ 실습(2).

- takeoff -> fwd(40) -> fwd(40) -> fwd(40) -> cw(180) -> fwd(40) -> fwd(40) -> fwd(40) -> land
- 

## 참고사항

### ✓ DJITelloPy 패키지 설치

- DJI Tello drone python interface using the official Tello SDK and Tello EDU SDK. This library has the following features:

- implementation of all tello commands
- easily retrieve a video stream
- receive and parse state packets
- control a swarm of drones
- [support for python >= 3.6](#)



- [pip install djitellopy](#)

- [pip install opencv-python](#)

- [from djitellopy import tello](#)

```
1 #기본적인 동작 제어
2 from djitellopy import Tello
3 tello = Tello()
4 tello.connect()
5 tello.takeoff()
6 tello.move_left(100)
7 tello.rotate_clockwise(90)
8 tello.move_forward(100)
9 tello.land()
```

```
1 #반복문 이용하기 : for in range()
2 from djitellopy import Tello
3 myTello = Tello()
4 myTello.connect()
5 myTello.takeoff()
6 myTello.move_up(30)
7 myTello.move_down(30)
8 myTello.move_up(30)
9 myTello.move_down(30)
10 myTello.move_up(30)
11 myTello.move_down(30)
12 myTello.land()
13
14 '''from djitellopy import Tello
15 myTello = Tello()
16 myTello.connect()
17 myTello.takeoff()
18 for i in range(0, 3):
19     myTello.move_up(30)
```

```
20 myTello.rotate_counter_clockwise(90)
21 myTello.move_down(30)
22 myTello.land()'''
```

```
1 #파이썬 함수 이용하기 : def
2 from djitellopy import Tello
3 myTello = Tello()
4 myTello.connect()
5 myTello.takeoff()
6 myTello.move_up(30)
7 myTello.move_down(30)
8 myTello.move_up(30)
9 myTello.move_down(30)
10 myTello.move_up(30)
11 myTello.move_down(30)
12 myTello.land()
13
14 '''def move_up_down(t):
15 myTello.move_up(t)
16 myTello.move_down(t)
17 for i in range(3):
18     t = 30
19     move_down(t)'''
```

---

## ✓ 실습(3)- for문 이용

- takeoff -> fwd(40) -> fwd(40) -> fwd(40) -> cw(180) -> fwd(40) -> fwd(40) -> fwd(40) -> cw(180) -> land

```
1 #input() 함수를 사용한 제어
2 from djitellopy import Tello
3 myTello = Tello()
```

```
4 myTello.connect()
5 battery_level = tello.get_battery()
6 print(battery_level)
7
8 while True:
9     command = int(input("Enter Command!"))
10    print(command, end="\n")
11    if (command == 1):
12        myTello.takeoff()
13    elif (command == 2):
14        myTello.move_up(30)
15    elif (command == 3):
16        myTello.move_down(30)
17    elif (command == 4):
18        myTello.land()
19    else :
20        break
21 print("Drone mission completed!")
```

---

## ✓ 실습(4).

- input() 함수를 이용한 드론 조종기 만들기

- takeoff()
- move\_up(20)
- move\_down(20)
- move\_left(20)
- move\_right(20)
- move\_forward(20)
- move\_backward(20)

- rotate\_clockwise(90)
- rotate\_counter\_clockwise(90)
- flip\_back()
- flip\_forward()
- flip\_left()
- flip\_right()
- land()

### ✓ 실습(5) - cross flight mission

```
go_xyz_speed(x, y, z, speed)
```

Fly to x y z relative to the current position. Speed defines the traveling speed in cm/s.

#### Parameters:

Name	Type	Description	Default
------	------	-------------	---------