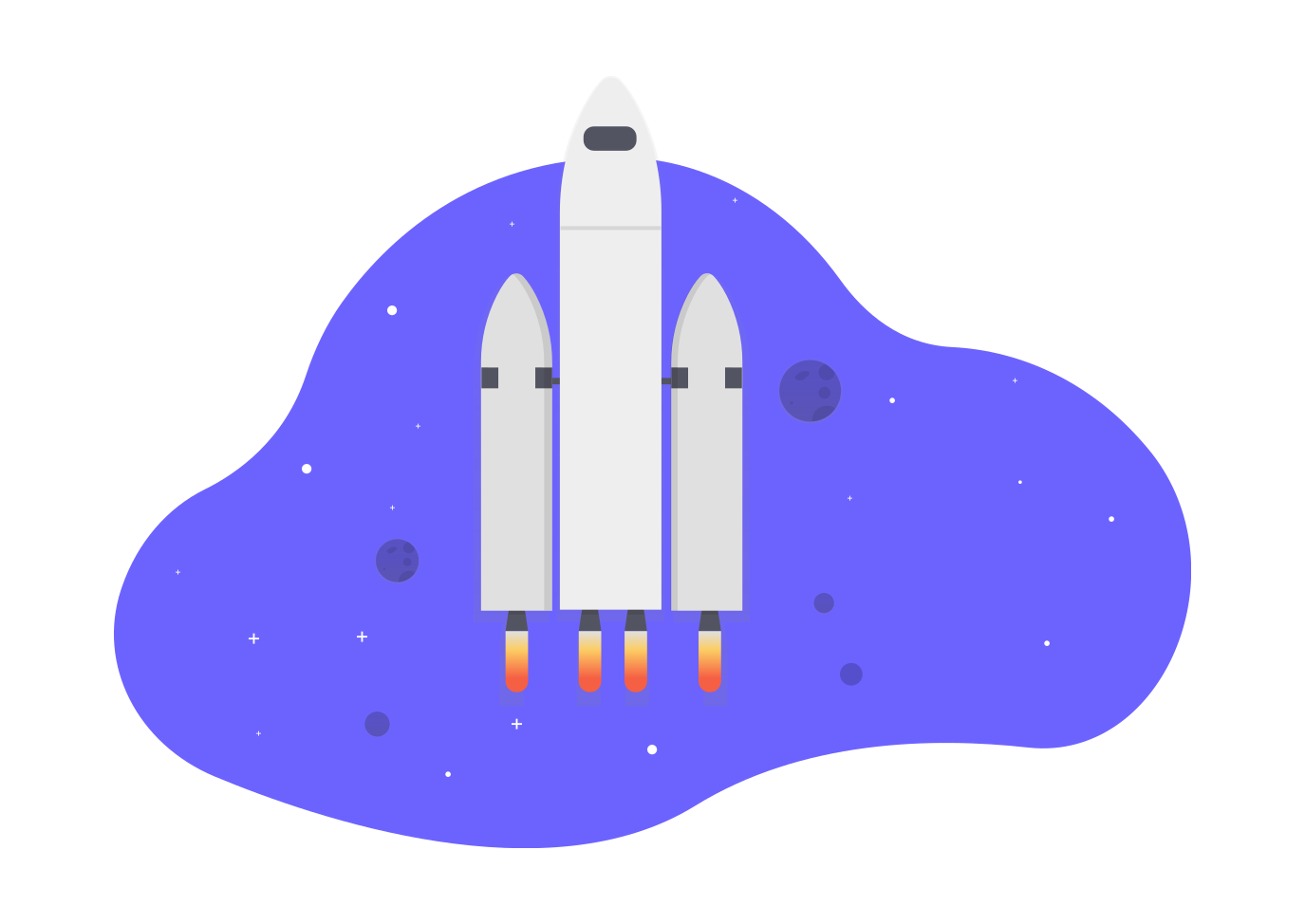
How to Use

ADB Controller

API Doc Version 1.0



**M2 Team Solution Development Part** 이한림

How to Use ADB Controller

[1. ADB Controller 소개 4](#_Toc77685661)

[1) 프로그램 소개 4](#_Toc77685662)

[2) 프로그램 특징 4](#_Toc77685663)

[3) 실행 전 요구 사항 4](#_Toc77685664)

[2. Support API 5](#_Toc77685665)

[1) 프로그램 지원 함수 5](#_Toc77685666)

[init 5](#_Toc77685667)

[setup\_test 6](#_Toc77685668)

[set\_print 7](#_Toc77685669)

[execute\_cmd 7](#_Toc77685670)

[save\_dump\_xml 8](#_Toc77685671)

[get\_pos\_elements 9](#_Toc77685672)

[2) Value Return 함수 10](#_Toc77685673)

[cmd\_return\_getPid 10](#_Toc77685674)

[cmd\_return\_getCallState 11](#_Toc77685675)

[3) Status Return 함수 12](#_Toc77685676)

[cmd\_status\_currentActivity 12](#_Toc77685677)

[cmd\_status\_killApp 13](#_Toc77685678)

[cmd\_status\_click 14](#_Toc77685679)

[cmd\_status\_swipe 15](#_Toc77685680)

[cmd\_status\_sendKey 17](#_Toc77685681)

[cmd\_status\_install 18](#_Toc77685682)

[cmd\_status\_uninstall 19](#_Toc77685683)

[cmd\_status\_backButton 20](#_Toc77685684)

[cmd\_status\_menuButton 21](#_Toc77685685)

[cmd\_status\_powerButton 22](#_Toc77685686)

[cmd\_status\_sendCall 22](#_Toc77685687)

[cmd\_status\_sendVideoCall 23](#_Toc77685688)

[cmd\_status\_endCall 24](#_Toc77685689)

[cmd\_status\_receiveCall 25](#_Toc77685690)

[cmd\_status\_sendSMS 26](#_Toc77685691)

[cmd\_status\_screenShot 27](#_Toc77685692)

[cmd\_status\_wifiOnOff 28](#_Toc77685693)

[cmd\_status\_cellOnOff 29](#_Toc77685694)

[cmd\_status\_gpsOnOff 30](#_Toc77685695)

[cmd\_status\_autoRotateOnOff 31](#_Toc77685696)

[cmd\_status\_blueToothOnOff 32](#_Toc77685697)

[cmd\_status\_airplaneOnOff 33](#_Toc77685698)

[3. How to Executes 35](#_Toc77685699)

[1) 단말기 PC와 연결 35](#_Toc77685700)

[2) Pycharm IDE 설치하기 36](#_Toc77685701)

[3) 프로그램 패키지 살펴보기 37](#_Toc77685702)

[4) 실행스크립트 작성 및 실행 방법 37](#_Toc77685703)

# ADB Controller 소개

1. 프로그램 소개

ADB Controller는 Python 기반으로 개발된 Android Phone 전용 자동화 실행 지원 프로그램 입니다. 해당 프로그램에서는 각종 실행 method(문자발신, 전화발수신, 비행기모드, App 실행 등등) 및 정보 확인 method API를(App PID, mCallStatus 등등) 지원하여 자동화 컨트롤이 가능하도록 하였으며 순수 ADB를 통해 Device 자동 Control 될 수 있도록 하여 별도의 라이브러리가 필요하지 않도록 하였습니다.

1. 프로그램 특징

* 복잡하지 않은 구조로 되어 있으며 다양한 자동 실행 함수 지원과 실행 자유도가 높음
* ADB 특성 상 원래 한글 입력이 어렵지만 ADB Keyboard 자동 설치 지원을 통해 가능하도록 변경(별도 설정 과정은 필요)
* Screen Capture 이미지 별도의 특정된 Folder로 저장 지원
* ADB가 Click 또는 Swipe등의 Action에서 절대 좌표 값으로 지정해야 하는 문제가 있었으나 프로그램에서 xml dump 파일을 통한 element 좌표 찾기 기능의 method를 제공하여 디바이스 변경에 따른 해상도 문제를 해결함
* 각각의 실행 Script를 독립적으로 따로 실행됨으로 동시에 여러 디바이스가 병렬적으로 동작함

1. 실행 전 요구 사항

* Python이(3.x 버전 이상) 설치되어 있어야 합니다. 설치는 Anaconda3 패키지로 설치하는 것을 권장합니다. 설치 관련 글은 [여기](https://somjang.tistory.com/entry/Windows-Windows%EC%97%90%EC%84%9C-Anaconda-%EC%84%A4%EC%B9%98%ED%95%98%EA%B8%B0?category=348812)를 Click 하여 참고하시기 바랍니다.
* ADB도 로컬 PC에 설치되어 있어야 합니다. 현재 Android Studio를 설치하면 자동으로 ADB가 설치되지만 용량이 무거운 관계로 Platform-Tool만 설치하고 싶다면 [여기](https://tnmsoft.tistory.com/214)를 Click하여 참고하시기 바랍니다.
* ADB가 Local PC에 설치된 상태에서 테스트 대상 단말기와 디버깅 연결을 미리 하셔야 원활한 테스트가 가능합니다.
* 전화, 문자 등등 기본 App 설정을 미리 정하시면 ADB 실행 연결 시 팝업 발생으로 인한 테스트 중단 현상을 예방할 수 있습니다.

# Support API

1. 프로그램 지원 함수

### init

|  |
| --- |
| **Function Desc** |
| 자동화 Script 실행 전 CMDS Class Initialized method이며 Class instance 시 자동으로 실행되며 아래와 같이 작업됨   * 파라메터 uuid에 의해 serial\_num 값 설정 * 결정된 serial\_num에 의해 ADB cmds 값 설정 * 파라메터 divide\_window에 의해 wm\_divide\_count 값 설정 |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| uuid | String | 테스트 대상 단말의 uuid(ADB상의 serial num) |
| divide\_window | Int | 테스트 대상 단말의 화면 X/Y 가상 구역 분리 개수  (Click 혹은 Swipe 시 선택될 가상의 구역) |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
|  |  |  |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20) |

### setup\_test

|  |
| --- |
| **Function Desc** |
| CMDS Class instance 후 가장 먼저 호출해야 하는 함수로 여기서 디바이스 연결상태, 화면 Display 상태 등등을 Check하고 한글 지원 ADBKeyboard 가 설치됨   * ADBKeyboard 설치 절차 진행 * check\_device\_serial method 실행(단말 연결 상태 확인) * check\_device\_manuInfo method 실행(제조사 확인) * check\_device\_modelName method 실행(디바이스 모델명 확인) * check\_device\_displayOn method 실행(디바이스 화면 상태 확인) * 이미지 저장 Folder 생성 * Xml dump file 저장 Folder 생성 |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
|  |  |  |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  cmd.setup\_test() |

### set\_print

|  |
| --- |
| **Function Desc** |
| Console에 print 하고 싶은 log 내용을 보여주는 method 시간 정보와 현재 디바이스 모델과 serial\_num가 log 내용과 함께 표시됨 |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| text | String | Console에 보여주고 싶은 Text 내용 표시 |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
|  |  |  |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  cmd.setup\_test()  cmd.set\_print(‘테스트입니다.’)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  테스트입니다. |

### execute\_cmd

|  |
| --- |
| **Function Desc** |
| ADB cmd를 parameter로 전달받아 해당 명령을 subprocess pipe로 Android device에 명령어 전달 지원 method이며 실행 상태에 따라 status 값과 cmd 실행 후 text를 return함 |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| cmd | String | ADB cmd 명령어 |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| return\_data | Array | **실행결과가 정상 Case**  [True, stdout\_text]  **실행결과가 비정상 Case**  [False, stderr\_text] |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  manu\_cmd = cmd.getManuInfo  returns = cmd.execute\_cmd(manu\_cmd)  cmd.set\_print(returns)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  [True, '[ro.product.manufacturer]: [samsung]'] |

### save\_dump\_xml

|  |
| --- |
| **Function Desc** |
| Android activity xml dump 실행 method로 실행 시점의 화면 xml tree node 파일을 저장한다. 주로 get\_pos\_elements 실행 전에 호출하는 것이 대부분인 지원 method 이며 실행 후 ‘xmlDump’ Folder에 xml 파일이 저장함 |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
|  |  |  |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # 현재 Activity xml tree node dump 파일 저장  cmd.save\_dump\_xml() |

### get\_pos\_elements

|  |
| --- |
| **Function Desc** |
| 보통 save\_dump\_xml 함수 호출 후에 주로 쓰이며 앞서 저장된 xml dump file에서 parameter로 입력된 attribute와 찾고자 하는 속성 값을 입력하면 해당하는 좌표 Array 값을 return함  주요 attribute는 아래와 같으며 거의 모든 attr 유형을 지원함   * **class:** Element class => android.widget.TextView * **resource-id**: Element ID => com.android.settings:id/switch\_widget * **content-desc**: accessibility element desc => login * **text**: Element text => 사용 |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| attr | String | Target Element의 attr 유형 입력(class ,resource-id..) |
| name | String | Target Element의 attr 유형 값 입력  \* 하나 이상 Case ‘|’을 구분자로 입력(확인|OK…) |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| List\_pos | Array | Target Element에 해당하는 모든 elements의 좌표 값이 tuple 형태의 Array로 return 됨 |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # 현재 Activity xml tree node dump 파일 저장  cmd.save\_dump\_xml()  # Target Element get pos   list\_pos = cmd.get\_pos\_elements(attr=’content-desc’, name=’공유하기’)  # list\_pos 값 확인  cmd.set\_print(list\_pos)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  [(235.25, 1220.73)] |

1. Value Return 함수

### cmd\_return\_getPid

|  |
| --- |
| **Function Desc** |
| App package name을 parameter로 입력하면 현재 android device process 목록에서 해당 App의  Pid를 return함 |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| name | String | Target App의 package 이름 입력(com.kakao.talk) |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| pid | String | **실행결과가 정상 Case**  해당 App pid 값(27096)  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # 현재 Activity xml tree node dump 파일 저장  pid = cmd.cmd\_return\_getPid(name=’com.kakao.talk’)  cmd.set\_print(pid)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  27096 |

### cmd\_return\_getCallState

|  |
| --- |
| **Function Desc** |
| Adnroid Device에서 Call State 값을 확인하여 return 함 |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| mCallState | String | **실행결과가 정상 Case**  “**idle**”(대기), “**ringing**”(수신 중), “**activity call**”(통화 중) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # 현재 Activity xml tree node dump 파일 저장  mCallState = cmd.cmd\_return\_getCallState()  cmd.set\_print(mCallState)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  idle |

1. Status Return 함수

### cmd\_status\_currentActivity

|  |
| --- |
| **Function Desc** |
| Foreground에서 현재 노출되고 있는 App의 package name을 parameter로 입력한 package name과 비교하여 일치 불일치 상태를 return 함 |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| name | String | 확인해야 할 App package 이름 입력(com.kakao.talk) |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (일치), **2**(불일치), **0**(비정상 동작) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # 현재 Foreground Activity package name 확인  status = cmd.cmd\_status\_currentActivity(name=’com.kakao.talk’)  cmd.set\_print(status)  # 현재 Foreground Activity package name이 com.kakao.talk로 일치  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_killApp

|  |
| --- |
| **Function Desc** |
| Parameter로 전달 받은 App의 package name을 기준으로 강제 종료 실행 후 상태를 return 함 |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| name | String | 종료해야 할 App package 이름 입력(com.kakao.talk) |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 종료), **0**(비정상 동작)**, 2**(이미 종료됨) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # parameter package name App 종료  status = cmd.cmd\_status\_killApp(name=’com.kakao.talk’)  cmd.set\_print(status)  # com.kakao.talk App 정상 종료  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_launchApp

|  |
| --- |
| **Function Desc** |
| Parameter로 전달 받은 App의 package name을 기준으로 app을 실행 후 상태를 return 함 |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| name | String | 실행할 App package 이름 입력(com.kakao.talk) |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 종료), **2**(App 실행 실패), **0**(비정상 동작) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # parameter package name App 실행  status = cmd.cmd\_status\_launchApp(name=’com.kakao.talk’)  cmd.set\_print(status)  # com.kakao.talk App 정상 실행  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_click

|  |
| --- |
| **Function Desc** |
| Parameter로 전달받은 X/Y absolute Position 또는 divided\_window로 나누어진 구역을 기준으로 click event를 실행 후 상태를 return 함 |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| width | Float/Int | X 좌표값으로 입력 값은 아래와 같음  **pos\_type이 abs Case**  X의 절대 좌표값(230.54)  **Pos\_type이 rate Case**  X의 선택한 구역 중간 값 |
| height | Float/Int | Y 좌표값으로 입력 값은 아래와 같음  **pos\_type이 abs Case**  Y의 절대 좌표값(230.54)  **Pos\_type이 rate Case**  Y의 선택한 구역 중간 값 |
| pos\_type | String | width / height의 입력 타입(‘abs’ 또는 ‘rate’) |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 동작), **0**(비정상 동작) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # 현재 abs type으로 절대좌표에 click event 실행  status = cmd.cmd\_status\_click(width=273.14, height=1150.12,  pos\_type=’abs’)  cmd.set\_print(status)  # 정상 실행  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1  # 현재 rate type으로 좌표에 click event 실행(divided count 가 10인 경우)  status = cmd.cmd\_status\_click(width=0, height=2,  pos\_type=’rate’)  cmd.set\_print(status)  # 정상 실행  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1  # Element으로 절대좌표에 click event 실행  cmd.save\_dump\_xml()  # Target Element get pos   list\_pos = cmd.get\_pos\_elements(attr=’content-desc’, name=’공유하기’)  status = cmd.cmd\_status\_click(width=list\_pos[0][0],  height=list\_pos[0][1], pos\_type=’abs’)  cmd.set\_print(status)  # 정상 실행  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_swipe

|  |
| --- |
| **Function Desc** |
| Parameter로 전달받은 X/Y absolute Position 또는 divided\_window로 나누어진 구역을 기준으로 Touch swipe event를 실행 후 상태를 return 함 |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| s\_width | Float/Int | Touch start X 좌표값으로 입력 값은 아래와 같음  **pos\_type이 abs Case**  X의 절대 좌표값(230.54)  **Pos\_type이 rate Case**  X의 선택한 구역 중간 값 |
| s\_height | Float/Int | Touch Start Y 좌표값으로 입력 값은 아래와 같음  **pos\_type이 abs Case**  Y의 절대 좌표값(230.54)  **Pos\_type이 rate Case**  Y의 선택한 구역 중간 값 |
| e\_width | Float/Int | Touch End X 좌표값으로 입력 값은 아래와 같음  **pos\_type이 abs Case**  X의 절대 좌표값(230.54)  **Pos\_type이 rate Case**  X의 선택한 구역 중간 값 |
| e\_height | Float/Int | Touch End Y 좌표값으로 입력 값은 아래와 같음  **pos\_type이 abs Case**  Y의 절대 좌표값(230.54)  **Pos\_type이 rate Case**  Y의 선택한 구역 중간 값 |
| duration | Int | Swipe event 동작 시간 (ms) |
| pos\_type | String | width / height의 입력 타입(‘abs’ 또는 ‘rate’) |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 동작), **0**(비정상 동작) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # 현재 abs type으로 절대좌표에 swipe event 실행  status = cmd.cmd\_status\_swipe(s\_width=273.14, S\_height=1150.12,  e\_width=273.14, e\_height=1150.12, duration=300, pos\_type=’abs’)  cmd.set\_print(status)  # 정상 실행  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1  # 현재 rate type으로 좌표에 swipe event 실행(divided count 가 10인 경우)  status = cmd.cmd\_status\_swipe s\_width=2, S\_height=6,  e\_width=2, e\_height=2, duration=300, pos\_type=’rate)  cmd.set\_print(status)  # 정상 실행  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1  # Element으로 절대좌표에 swipe event 실행  cmd.save\_dump\_xml()  # Target Element get pos   s\_list\_pos = cmd.get\_pos\_elements(attr=’content-desc’, name=’공유하기’)  e\_list\_pos = cmd.get\_pos\_elements(attr=’text’, name=’끝’)  status = cmd.cmd\_status\_swipe(s\_width=s\_list\_pos[0][0],  s\_height=s\_list\_pos[0][1], e\_width=e\_list\_pos[0][0],  e\_height=e\_list\_pos[0][1], duration=300, pos\_type=’abs’)  cmd.set\_print(status)  # 정상 실행  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_sendKey

|  |
| --- |
| **Function Desc** |
| Parameter로 전달한 message를 ADB cmd를 통해 입력되도록 실행 후 상태 return |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| message | String | 입력해야 할 Text 입력 |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (일치), **0**(비정상 동작) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # 현재 abs type으로 android Input type절대좌표에 click event 실행 cmd.cmd\_status\_click(width=273.14, height=1150.12,  pos\_type=’abs’)  status = cmd.cmd\_status\_sendKey(message=’테스트입니다.’)  # sendKey event status print  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_install

|  |
| --- |
| **Function Desc** |
| Parameter로 전달한 apk path로 Android device에 App을 install 실행 후 상태를 return |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| path | String | Install 대상 apk 파일 위치(Local PC) |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 설치), **0**(비정상 동작), **2**(설치 실패) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # test apk 설치 실행  status = cmd.cmd\_status\_install(path=’c://program Files/test.apk’)  # install event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_uninstall

|  |
| --- |
| **Function Desc** |
| Parameter로 전달한 package name으로 Android device에 App을 uninstall 실행 후 상태를 return |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| name | String | unInstall 대상 app package name(com.kakao.talk) |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 삭제), **0**(비정상 동작), **2**(삭제 실패) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # test app 삭제 실행  status = cmd.cmd\_status\_uninstall(name=’com.kakao.talk’)  # uninstall event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_backButton

|  |
| --- |
| **Function Desc** |
| Parameter로 전달받은 iter\_count 만큼 반복해서 delay 시간의(s) 만큼 유예시간을 주고 Android Device back button event를 실행 후 상태를 return |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| iter\_count | Int | 입력한 숫자 횟수에 따라 Back button event가 반복됨 |
| delay | Int | 입력한 시간(s)에 따라 유예를 두고 Back button event 실행 |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 실행), **0**(비정상 동작) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # back button event실행  status = cmd.cmd\_status\_backButton(iter\_count=2, delay=2)  # back button event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_menuButton

|  |
| --- |
| **Function Desc** |
| Android Device menu button event를 실행 후 상태를 return |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 실행), **0**(비정상 동작) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # menu button event실행  status = cmd.cmd\_status\_menuButton()  # menu button event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_powerButton

|  |
| --- |
| **Function Desc** |
| Android Device power button event 실행 후 상태를 return |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 실행), **0**(비정상 동작) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # power button event실행  status = cmd.cmd\_status\_powerButton()  # power button event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_sendCall

|  |
| --- |
| **Function Desc** |
| Parameter로 전달한 phone\_num로 Android 기본 전화 App을 통해 발신 실행 후 상태를 return(**반드시 전화 기본 App 설정이 되어 있어야 함**) |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| phone\_num | String | 수신자 전화번호(01055551234) |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 발신), **0**(비정상 동작), **2**(발신 실패) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # 01055551234 전화 발신 실행  status = cmd.cmd\_status\_sendCall(phone\_num=’01055551234’)  # send Call event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_sendVideoCall

|  |
| --- |
| **Function Desc** |
| Parameter로 전달한 phone\_num로 Android 기본 전화 App을 통해 영상통화 발신 실행 후 상태를 return(**반드시 전화 기본 App 설정이 되어 있어야 함**) |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| phone\_num | String | 수신자 전화번호(01055551234) |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 발신), **0**(비정상 동작), **2**(발신 실패) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # 01055551234 전화 영상통화 발신 실행  status = cmd.cmd\_status\_sendVideoCall(phone\_num=’01055551234’)  # send Call event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_endCall

|  |
| --- |
| **Function Desc** |
| Android 기본 전화 App의 통화 종료 event 실행 후 상태를 return |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 실행), **0**(비정상 동작) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # power button event실행  status = cmd.cmd\_status\_powerButton()  # power button event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_receiveCall

|  |
| --- |
| **Function Desc** |
| Android 기본 전화 App의 전화 수신 시 receive event 실행 후 상태를 return |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 실행), **0**(비정상 동작) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # 전화 수신 대기  call\_state = ‘idle’  while True:  call\_state = cmd.cmd\_return\_getCallState()  if call\_state == ‘ringing’:  break  sleep(1)    # receive call event실행  status = cmd.cmd\_status\_receiveCall()  # power button event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_sendSMS

|  |
| --- |
| **Function Desc** |
| Parameter로 전달한 phone\_num로 message 내용을 Android 기본 문자 App을 통해 SMS 발신 실행 후 상태를 return(**반드시 문자 기본 App 설정이 되어 있어야 함**) |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| phone\_num | String | 수신자 전화번호(01055551234) |
| message | String | SMS로 보낼 내용 |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 발신), **0**(비정상 동작) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # 01055551234 SMS 발신 실행  status = cmd.cmd\_status\_sendSMS(phone\_num=’01055551234’,  message=’안녕하세요 테스터 입니다.’)  # send SMS event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_screenShot

|  |
| --- |
| **Function Desc** |
| Parameter로 전달받은 name을 파일 이름으로 하여 delay 시간의(s) 만큼 유예시간을 주고 스크린 샷 event를 실행 파일을 ‘capture’ Folder에 png 형태로 저장 후 상태를 return |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| delay | Int | 입력한 시간(s)에 따라 유예를 두고 스크린 샷 실행 |
| name | String | 파일이름 지정(확장자명은 제외) |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 동작), **0**(비정상 동작) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # 현재 화면 scrren shot 실행  status = cmd.cmd\_status\_screenShot(delay=2,  name=’바탕화면\_1’)  # screen shot event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_wifiOnOff

|  |
| --- |
| **Function Desc** |
| Parameter로 전달받은 exe\_type을 기준으로 On/Off 실행하며 delay 시간의(s) 만큼 유예시간을 주고 wifi enable/disable 상태를 check 후 return |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| exe\_type | Int | On/Off 실행 타입  **On : 1입력 Off: 0입력** |
| delay | Int | 입력한 시간(s)에 따라 유예를 두고 wifi state check |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 동작), **0**(비정상 동작). **2**(On/Off control 실패) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # wifi On실행  status = cmd.cmd\_status\_wifiOnOff(exe\_type=1, delay=1)  # wifi on event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1  # wifi Off실행  status = cmd.cmd\_status\_wifiOnOff(exe\_type=0, delay=1)  # wifi off event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_cellOnOff

|  |
| --- |
| **Function Desc** |
| Parameter로 전달받은 exe\_type을 기준으로 On/Off 실행하며 delay 시간의(s) 만큼 유예시간을 주고 cellular(data) enable/disable 상태를 check 후 return |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| exe\_type | Int | On/Off 실행 타입  **On : 1입력 Off: 0입력** |
| delay | Int | 입력한 시간(s)에 따라 유예를 두고 cellular state check |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 동작), **0**(비정상 동작). **2**(On/Off control 실패) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # cellular(data) On실행  status = cmd.cmd\_status\_cellOnOff(exe\_type=1, delay=1)  # cellular on event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1  # cellular(data) Off실행  status = cmd.cmd\_status\_cellOnOff(exe\_type=0, delay=1)  # cellular off event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_gpsOnOff

|  |
| --- |
| **Function Desc** |
| Parameter로 전달받은 exe\_type을 기준으로 On/Off 실행하며 delay 시간의(s) 만큼 유예시간을 주고 gps service enable/disable 상태를 check 후 return |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| exe\_type | Int | On/Off 실행 타입  **On : 1입력 Off: 0입력** |
| delay | Int | 입력한 시간(s)에 따라 유예를 두고 gps state check |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 동작), **0**(비정상 동작). **2**(On/Off control 실패) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # gps service On실행  status = cmd.cmd\_status\_cellOnOff(exe\_type=1, delay=1)  # gps on event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1  # gps service Off실행  status = cmd.cmd\_status\_cellOnOff(exe\_type=0, delay=1)  # gps off event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_autoRotateOnOff

|  |
| --- |
| **Function Desc** |
| Parameter로 전달받은 exe\_type을 기준으로 On/Off 실행하며 delay 시간의(s) 만큼 유예시간을 주고 auto rotate enable/disable 상태를 check 후 return |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| exe\_type | Int | On/Off 실행 타입  **On : 1입력 Off: 0입력** |
| delay | Int | 입력한 시간(s)에 따라 유예를 두고 auto rotate state check |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 동작), **0**(비정상 동작). **2**(On/Off control 실패) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # auto rotate On실행  status = cmd.cmd\_status\_autoRotateOnOff(exe\_type=1, delay=1)  # auto rotate on event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1  # auto rotate Off실행  status = cmd.cmd\_status\_autoRotateOnOff(exe\_type=0, delay=1)  # auto rotate off event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_blueToothOnOff

|  |
| --- |
| **Function Desc** |
| Parameter로 전달받은 exe\_type을 기준으로 On/Off 실행하며 delay 시간의(s) 만큼 유예시간을 주고 blueTooth service enable/disable 상태를 check 후 return |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| exe\_type | Int | On/Off 실행 타입  **On : 1입력 Off: 0입력** |
| delay | Int | 입력한 시간(s)에 따라 유예 후 blueTooth state check |

|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 동작), **0**(비정상 동작). **2**(On/Off control 실패) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # blue Tooth service On실행  status = cmd.cmd\_status\_blueToothOnOff(exe\_type=1, delay=1)  # blueTooth on event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1  # blueTooth service Off실행  status = cmd.cmd\_status\_blueToothOnOff(exe\_type=0, delay=1)  # blueTooth off event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

### cmd\_status\_airplaneOnOff

|  |
| --- |
| **Function Desc** |
| Parameter로 전달받은 exe\_type을 기준으로 On/Off 실행하며 delay 시간의(s) 만큼 유예시간을 주고 airplane mode enable/disable 상태를 check 후 return  **\* LG/Samsung Device만 지원 가능** |

|  |  |  |
| --- | --- | --- |
| **Parameter** | | |
| **Name** | **Type** | **Description** |
| exe\_type | Int | On/Off 실행 타입  **On : 1입력 Off: 0입력** |
| delay | Int | 입력한 시간(s)에 따라 유예 후 airplane mode state check |

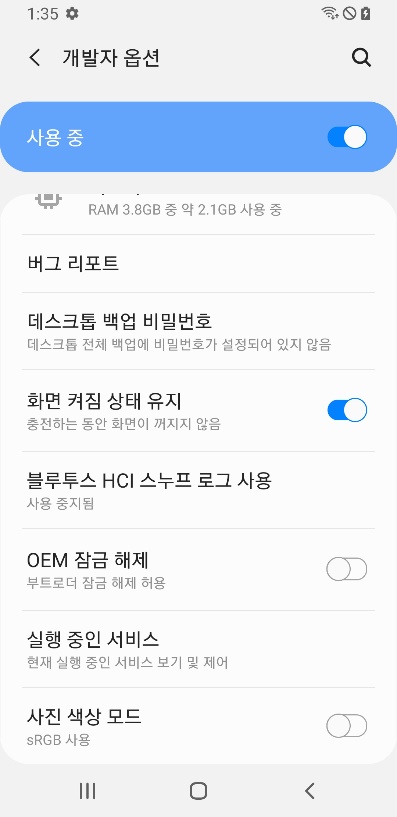
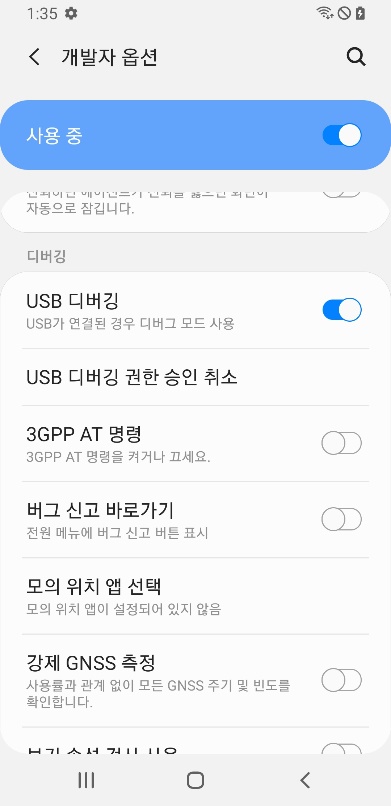
|  |  |  |
| --- | --- | --- |
| **Return** | | |
| **Name** | **Type** | **Description** |
| status | Int | **실행결과가 정상 Case**  **1** (정상 동작), **0**(비정상 동작). **2**(On/Off control 실패) 이들 중 하나가 리턴  **실행결과가 비정상 Case**  None |

|  |
| --- |
| **Exam** |
| if \_\_name\_\_ == "\_\_main\_\_":   serial\_num = '1c25c664460c7ece'  window\_count = 20  cmd = CMDS(serial\_num, 20)  # 필수 setup method 반드시 호출  cmd.setup\_test()  # airplane mode On실행  status = cmd.cmd\_status\_cellOnOff(exe\_type=1, delay=1)  # airplane mode on event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1  # airplane mode Off실행  status = cmd.cmd\_status\_cellOnOff(exe\_type=0, delay=1)  # airplane mode off event print status  cmd.set\_print(status)  [SM-G960N(1c25c664460c7ece)]2021-07-19 11:28:15:  1 |

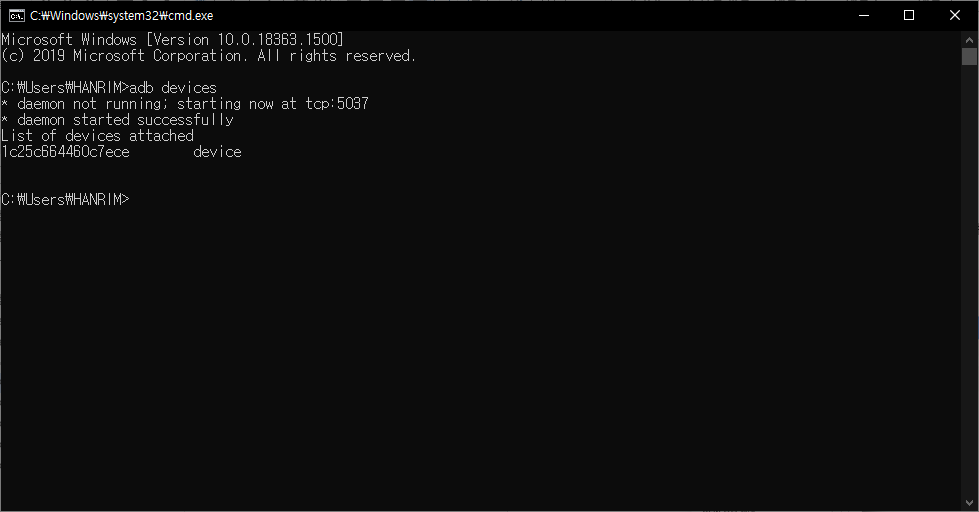
# How to Executes

1. 단말기 PC와 연결

* 단말기를 PC와 USB로 연결한 후 개발자모드(옵션)를 활성화 합니다. 개발자 모드 활성화에 대한 방법은 [여기](https://appstips.tistory.com/entry/%EC%95%88%EB%93%9C%EB%A1%9C%EC%9D%B4%EB%93%9C-%EA%B0%9C%EB%B0%9C%EC%9E%90%EB%AA%A8%EB%93%9C-%ED%99%9C%EC%84%B1%ED%99%94-%EB%B9%84%ED%99%9C%EC%84%B1%ED%99%94-%EB%B0%A9%EB%B2%95%EA%B3%BC-%EC%B5%9C%EC%A0%81%ED%99%94-%ED%8C%81)를 참조합니다.
* 개발자 모드(옵션) 화면에서 ‘USB 디버깅’ 및 '화면 켜짐 상태 유지’를 활성화 합니다.

* window cmd 창에서 ‘**adb devices’**를 입력하면(adb가 설치되어 있으며 환경변수 설정도 된 상태여야 함) 아래 그림과 같이 ‘serial number’와 ‘deivce’ text가 노출되면 정상적으로 연결된 상태이며 출력된 serial number’도 메모장에 저장하여 둡니다.



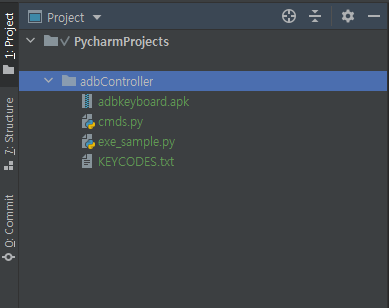
1. Pycharm IDE 설치하기

* 물론 cmd 창으로 Python 프로그램을 실행시킬 수 있지만 자동화 Script 작성 및 실행을 좀더 편하고 빠르게 할 수 있도록 IDE 도구를 이용합니다.
* Python에 특화된 IDE로 Pycharm 커뮤니티 무료 버전을 추천드리며 설명에서도 Pycharm을 기준으로 설명합니다. 설치 관련해서는 [여기](https://blockdmask.tistory.com/346)를 참조해 주세요.



1. 프로그램 패키지 살펴보기

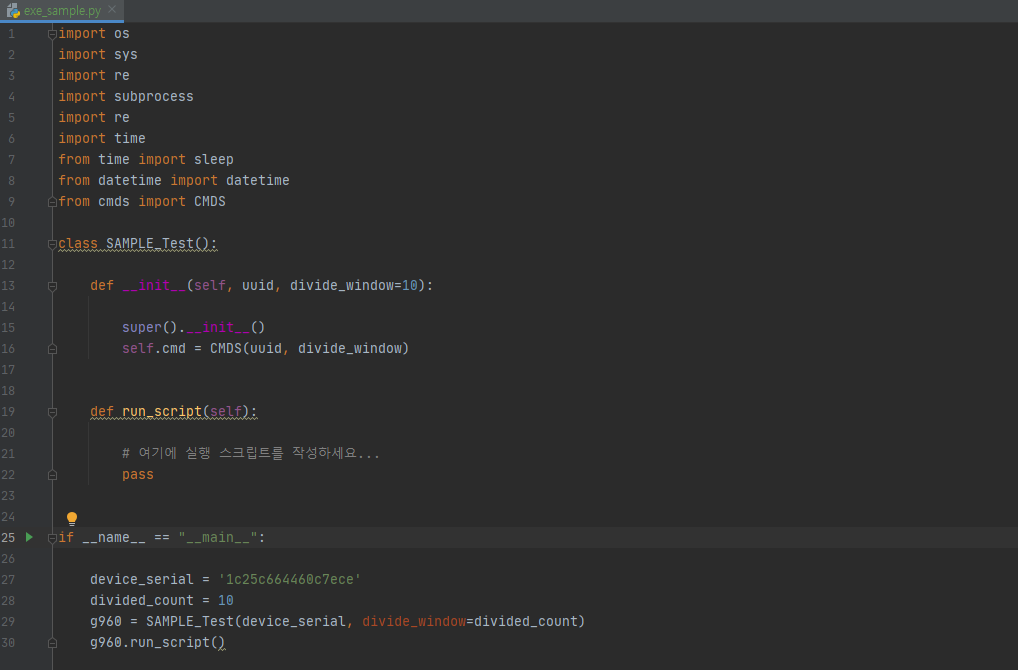
* Pycharm으로 프로그램 패키지를 열면 아래와 같이 구성되어 있습니다.



* ‘adbKeyboard.apk’ 는 setup\_test 함수 실행 시 테스트 대상 단말기에 adb로 한글 입력이 가능하도록 지원하는 app이며 설치가 안되어 있다면 설치 및 기본 키보드로 설정을 유도하게 됩니다. 기본 키보드 변경 방법은 [여기](https://anywhereifyoucan.com/152)를 참조해 주세요.
* cmds.py는 core 모듈로서 자동화 Script 실행 모듈에서 필요로 하는 method, adb 명령어 집합, 각종 필요 flag 변수 등이 있으며 해당 모듈을 실행 모듈에서 instance해서 사용합니다.
* exe\_sample.py는 실제 자동화 실행 모듈로서 테스트할 연결 단말 수만큼 새로이 생성할 수도 있으며 모듈명도 작성자 마음대로 변경할 수 있습니다. 해당 모듈에서 실행 Script를 작성하면 됩니다.
* KEYCODES.txt 파일에는 많이 쓰이는 adb 명령어와 동영상 또는 사진 실행 순서 등을 정리한 텍스트 파일이며 실행 Script 참고용으로 보면 됩니다.

1. 실행스크립트 작성 및 실행 방법

* 실행스크립트 모듈을(exe\_sample.py) 보면 아래 그림과 같이 되어 있습니다.

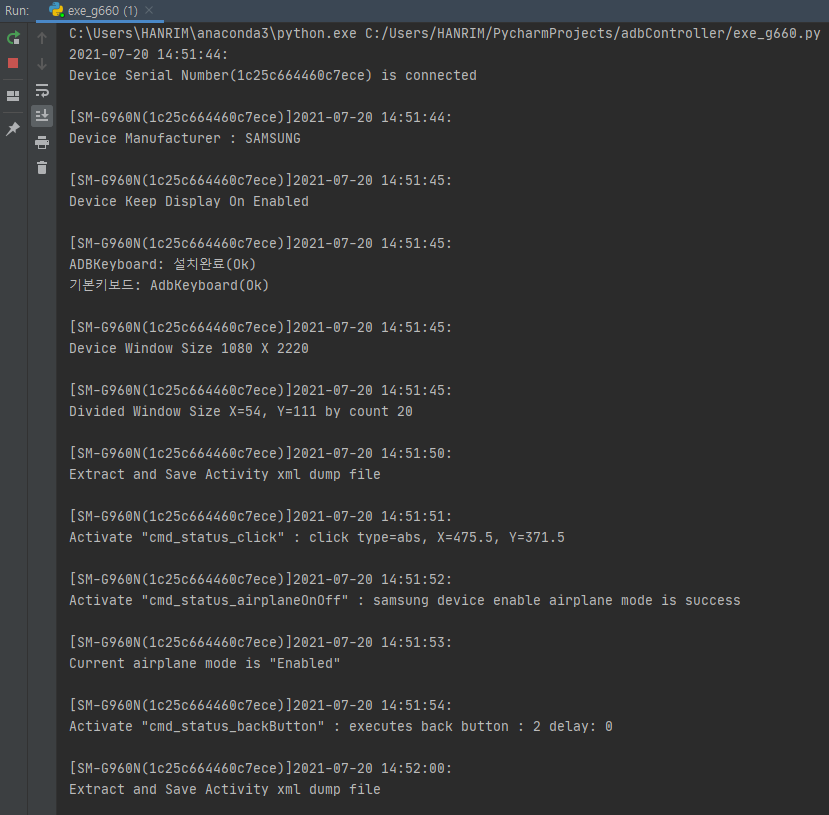


* 상단에 import 부분은 필요 패키지를 불러오는 부분으로 만약 실행 모듈에 필요한 패키지가 있다면 해당 영역에 import 하도록 합니다.
* run\_script 함수 부분에 사용자가 자동으로 실행해야 할 스크립트를 작성합니다. 실행 함수에서 필요한 adb 지원함수는 ‘\_ \_init\_ \_’ 생성자에서 ‘**CMDS’** class를 instance 해 놓은 상태임으로 **self.cmd**로 불러오도록 하며 또는 사용자가 생성자의 ‘self.cmd’ 이름을 바꾸셔도 무방합니다.
* 하단의 ‘if \_ \_name\_ \_ == “\_ \_main\_ \_:”’ 부분이 실행함수 start 부분이며 ‘run\_script’가 속해있는 함수의 class를 instance 하여주고 {class name}.run\_script()를 넣어주도록 합니다. Class instance 시 필수 생성자인 device\_serial 과 divided\_count를 입력하도록 합니다.
* Run\_script 함수 작성 예시는 아래와 같이 할 수 있습니다.

import os  
import sys  
import re  
import subprocess  
import re  
import time  
from time import sleep  
from datetime import datetime  
from cmds import CMDS  
  
class SAMPLE\_Test():  
  
 def \_\_init\_\_(self, uuid, divide\_window=10):  
  
 super().\_\_init\_\_()  
 self.cmd = CMDS(uuid, divide\_window

def run\_script(self):  
  
 # 필수 setup method 반드시 호출  
 self.cmd.setup\_test()  
 # ###############\_\_비행기 모드 테스트\_\_#################  
 self.cmd.cmd\_status\_airplaneOnOff(exe\_type=1, delay=1)  
 self.cmd.cmd\_status\_backButton(iter\_count=2)  
 sleep(2)  
 self.cmd.cmd\_status\_airplaneOnOff(exe\_type=0, delay=1)  
 self.cmd.cmd\_status\_backButton(iter\_count=2)  
 sleep(2)  
 self.cmd.cmd\_status\_airplaneOnOff(exe\_type=0,delay=1)  
 self.cmd.cmd\_status\_backButton(iter\_count=2)  
  
 self.cmd.cmd\_status\_autoRotateOnOff(exe\_type=1, delay=2)  
 sleep(3)  
 self.cmd.cmd\_status\_autoRotateOnOff(exe\_type=0, delay=2)  
 sleep(3)  
 self.cmd.cmd\_status\_backButton(iter\_count=2)  
  
 # ##############\_\_BlueTooth 모드 테스트\_\_###############  
 self.cmd.cmd\_status\_blueToothOnOff(exe\_type=1, delay=1)  
 self.cmd.cmd\_status\_backButton(iter\_count=2)  
 sleep(2)  
 self.cmd.cmd\_status\_blueToothOnOff(exe\_type=0, delay=1)  
 self.cmd.cmd\_status\_backButton(iter\_count=2)  
 sleep(2)  
 self.cmd.cmd\_status\_blueToothOnOff(exe\_type=0, delay=1)  
 self.cmd.cmd\_status\_backButton(iter\_count=2)  
  
 # # #############\_\_화면 회전 모드 테스트\_\_###############  
 self.cmd.cmd\_status\_autoRotateOnOff(exe\_type=1, delay=2)  
 sleep(3)  
 self.cmd.cmd\_status\_autoRotateOnOff(exe\_type=0, delay=2)  
 sleep(3)  
 self.cmd.cmd\_status\_backButton(iter\_count=2)  
 self.cmd.cmd\_status\_screenShot(delay=2, name='sample\_test1')  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
  
 device\_serial = '1c25c664460c7ece'  
 divided\_count = 10  
 sample = SAMPLE\_Test(device\_serial, divide\_window=divided\_count)  
 sample.run\_script()

* run\_script에서 실제 동작 Script를 작성하는 방식이며 **if, while, try catch** 등 논리나 반복문과 같은 logic을 넣어서 작성하면 더욱 정교하게 실행될 수 있습니다.
* 실행 방법은 exe\_sample 모듈 화면에서 아무 영역에서 마우스 우클릭 ‘Run exe\_sample’을 클릭하면 아래 그림과 같이 실행돕니다.



* 만약 cmd\_status\_screenShot을 실행한 부분이 있는 경우엔 현재 프로그램이 위치한 폴더에 ‘capture({단말기 serial num})’이름으로 screen shot 결과가 저장되어 있습니다. 해당 폴더에서 확인하시면 됩니다.

