

AirMapSDK

version 1.0A

Airmap

September 26, 2016

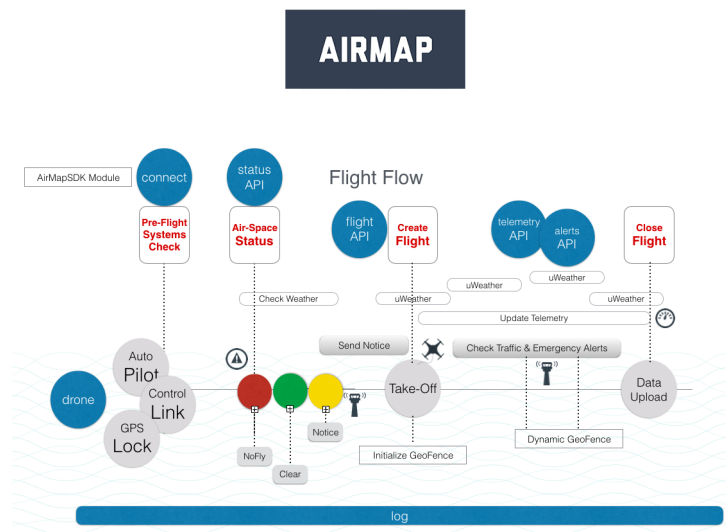
Contents

Welcome to AirMap's documentation!	1
AirMap Package	2
airdefs Module	2
connect Module	4
statusAPI Module	5
flightAPI Module	6
drone Module	7
log Module	7
telemetryAPI Module	7
alertsAPI Module	7
Indices and tables	7
Index	9
Python Module Index	11

Welcome to AirMap's documentation!

Airmap -- Airspace Management For Drones.

AirMapSDK is a Python library for interfacing with the [Airmap](#) API.



Pre-Requisites:

Airmap developer account
X-API-Key from your account
GPS data (fake gps included)
OAuth token
getAirmap.py (copy to /home/root/installs directory or similar)
python requests package

Install requests* | Insert USB drive with requests-master.zip | `sudo mount /dev/sda1 /mnt | cp /mnt/requests-master.zip /home/root/new_dir | cd /home/root/new_dir | unzip requests-master.zip | cd requests-master | python setup.py install`

Wi-Fi Connectivity

ifconfig (check for wlan0 or LTE ethX)
cd /etc
sudo wpa_passphrase yourSSID yourPassphrase > wpa-Aero.conf
sudo wpa_supplicant -i wlan0 -c /etc/wpa-Aero.conf &
sudo dhclient wlan0

Install simplejson

```
cd /home/root
mkdir installs
cd installs
python getAirmap.py simplejson simplejson
unzip simplejson-master.zip
cd simplejson-master
python setup.py install
```

Install AirMapSDK

```
cd /home/root
mkdir Installs
cd installs
python getAirmap.py ricardoairmap EmbeddedSDK
unzip EmbeddedSDK-master.zip
cd EmbeddedSDK-master
```

User Example

AirMap Package

```
cd /home/root/Installs/EmbeddedSDK-master
python userapp.py
```

API Example

```
(connect) airmap.connect.set_XAPIKey(xapikey) - Set your X-API-Key from your Airmap account
(connect) airmap.connect.get_CIDID() - Retrieve CID information
(connect) airmap.connect.connect() - Check internet connection and endpoint status Returns: True if ready
(statusAPI) airmap.statusAPI.get_status(lat,lon,Weather.on) - Given position retrieve airspace data
(statusAPI) Parse status information including weather, advisories, and maximum flight bounds (see statusAPI
documentation)
(user) Process required notifications if needed
(connect) airmap.connect.get_SecureToken() - get security token
(flightAPI) airmap.flightAPI.create_FlightPoint (flight time,lat,lon,publicflight?,sendnotifications?) - Setup the flight
and flight time Returns flightID
(flightAPI) airmap.flightAPI.get_PilotID() - Returns pilotID for this account
(flightAPI) airmap.flightAPI.end_Flight(flightID) - End the flight specified by flight ID
(flightAPI) airmap.flightAPI.delete_Flight(flightID) - Delete flight described by flightID
(flightAPI) airmap.flightAPI.get_FlightList(pilotID) - Get all flights for pilotID
(flightAPI) airmap.flightAPI.cmd_KillFlights(pilotID) - Delete all flights under specified pilotID
```

Documentation can be found at <https://developer.airmap.io>

Source code can be found at <https://github.com/...>

Airmap is a trademark of Airmap, Inc.

Contents:

AirMap Package

airdefs Module

airdefs AirMapSDK

Created by AirMap Team on 6/28/16. Copyright (c) 2016 AirMap, Inc. All rights reserved.

```
class airmap.airdefs.Advisories (distance, last_updated, name, city, color, country, longitude,
properties, state, latitude, type, id)
```

Advisory group list

Notes: Distance, date of last update, name, city, color(status), country, longitude, properties, state, latitude, type, id

```
class airmap.airdefs.Advisory
```

Advisory information (Pre-Flight, In-Flight)

Notes: Advisory status color list

```
class Color
```

Flight status color code

Colors

Color code

Parameters:

- **gray** -- Disabled
- **green** -- Go
- **yellow** -- Advise
- **red** -- NoFly

alias of **Enum**

```
enum (*sequential, **named)
```

AirMap Package

`class airmap.airdefs.Globals`

Global settings

Notes: Session parameters address, port, timeout, api key, token, pilot id, flight id

`AirConnected = False`

`dbgPrint (data)`

Debug send to console

Parameters: `data` -- Debug or information to send to console

Returns: None

`httpPort = 80`

`httpsAddr = 'api.airmap.io'`

`httpsPort = 443`

`keyAddr = 'sso.airmap.io'`

`myFlightID = None`

`myToken = None`

`pilotIDValid = False`

`pilot_id = None`

`strPrint (data)`

Information send to console

Parameters: `data` -- information to send to console

Returns: None

`telemetryAddr = 'api-aero-telemetry.airmap.com'`

`thisCID = None`

`timeOut = 18`

`xapikey = None`

`class airmap.airdefs.Notify`

Enable notifications

Parameters:

- **on** -- Enable notifications
- **off** -- Disable notifications

`off = False`

`on = True`

`class airmap.airdefs.Properties (prop_name, prop_value)`

Name to Value Keypairs

`class airmap.airdefs.Public`

Make flight public

Parameters:

- **on** -- Public flight
- **off** -- Private flight

off = *False*

on = *True*

class airmap.airdefs.**Requirement**

Notification requirments contact key pair list

State

alias of **Enum**

enum (**sequential*, ***named*)

class airmap.airdefs.**Startup**

Startup configuration

Notes:

class **Drone**

Drone information (ID, Location, Status)

State

alias of **Enum**

enum (**sequential*, ***named*)

class airmap.airdefs.**Weather**

Weather control parameters

Parameters:

- **on** -- Enable weather data
- **off** -- Disable weather data

off = *'false'*

on = *'true'*

connect *Module*

connect AirMapSDK

Created by AirMap Team on 6/28/16. Copyright (c) 2016 AirMap, Inc. All rights reserved.

class airmap.connect.**Connect**

connect ()

Connect to service

Param: None

Returns: True - if connected otherwise False

connection = *None*

Connection instance

Notes: HTTPS access variable

get_CIDID ()

Retrieve CID from mmcbk0

Param: None
Returns: CID otherwise False

get_SecureToken ()

Retrieve security token and refresh

Param: None
Returns: Token if successful otherwise False
Todo: Remove hardcoded token and add token from https endpoint based on CID

headers = None

Connection headers security and format

Notes: Security and format headers

os = <module 'os' from '/usr/lib/python2.7/os.pyc'>

OS access

Notes: Run sys commands

set_Timeout (time_out)

Set https request timeout time

Parameters: **time_out** -- Timeout time in seconds
Returns: True - Success, False - Fail

set_XAPIKey (xapikey)

Set https request timeout time

Parameters: **xapikey** -- X-API-Key from developers account
Returns: True - Success, False - Fail

thisGlobals = <airmap.airdefs.Globals instance at 0x2afd1f5346c8>

Global parameter access

Notes: Endpoint address, ports, token, id(s)

statusAPI Module

statusAPI AirMapSDK

Created by AirMap Team on 6/28/16. Copyright (c) 2016 AirMap, Inc. All rights reserved.

class airmap.statusAPI.Status

cmd_ProcessAdvisories ()

connection = None

get_Advisories ()

get_Advisory (id)

get_Condition ()

get_Humidity ()

get_MaxDistance ()

get_Precipitation ()

```
get_StatusCode ()
get_StatusColor ()
get_Temperature ()
get_Visibility ()
get_WindGusting ()
get_WindHeading ()
get_WindSpeed ()
get_status (gps_lat, gps_lon, weather)
headers = None
levelDown (data)
localAdvisories = []
localLevelDown = []
localProperties = []
os = <module 'os' from '/usr/lib/python2.7/os.pyc'>
status_json = None
thisGlobals = <airmap.airdefs.Globals instance at 0x2afd1f549830>
```

flightAPI Module

flightAPI AirMapSDK

Created by AirMap Team on 6/28/16. Copyright (c) 2016 AirMap, Inc. All rights reserved.

```
class airmap.flightAPI.Flight
```

```
    cmd_KillFlights (pilotID)
    connection = None
    create_FlightPoint (time, lat, lon, public, notify)
    delete_Flight (flightID)
    end_Flight (flightID)
    get_FlightList (pilotID)
    get_PilotID ()
    headers = None
    os = <module 'os' from '/usr/lib/python2.7/os.pyc'>
    thisGlobals = <airmap.airdefs.Globals instance at 0x2afd1f544f38>
```

drone Module

drone AirMapSDK

Created by AirMap Team on 6/28/16. Copyright (c) 2016 AirMap, Inc. All rights reserved.

```
class airmap.drone.drone
```

log Module

log AirMapSDK

Created by AirMap Team on 6/28/16. Copyright (c) 2016 AirMap, Inc. All rights reserved.

```
class airmap.log.log
```

telemetryAPI Module

telemetryAPI AirMapSDK

Created by AirMap Team on 6/28/16. Copyright (c) 2016 AirMap, Inc. All rights reserved.

```
class airmap.telemetryAPI.Telemetry

    connection = None

    headers = None

    os = <module 'os' from '/usr/lib/python2.7/os.pyc'>

    put_Telemetry (flightID, lat, lon)

    thisGlobals = <airmap.airdefs.Globals instance at 0x2afd1f544f80>
```

alertsAPI Module

alertsAPI AirMapSDK

Created by AirMap Team on 6/28/16. Copyright (c) 2016 AirMap, Inc. All rights reserved.

```
class airmap.alertsAPI.alertsAPI
```

Indices and tables

- *genindex*
- *modindex*
- *search*

Index

A

Advisories (class in [airmap.airdefs](#))
Advisory (class in [airmap.airdefs](#))
Advisory.Color (class in [airmap.airdefs](#))
AirConnected ([airmap.airdefs.Globals](#) attribute)
[airmap.airdefs](#) (module)
[airmap.alertsAPI](#) (module)
[airmap.connect](#) (module)
[airmap.drone](#) (module)
[airmap.flightAPI](#) (module)
[airmap.log](#) (module)
[airmap.statusAPI](#) (module)
[airmap.telemetryAPI](#) (module)
[alertsAPI](#) (class in [airmap.alertsAPI](#))

C

[cmd_KillFlights\(\)](#) ([airmap.flightAPI.Flight](#) method)
[cmd_ProcessAdvisories\(\)](#) ([airmap.statusAPI.Status](#) method)
Colors ([airmap.airdefs.Advisory.Color](#) attribute)
Connect (class in [airmap.connect](#))
[connect\(\)](#) ([airmap.connect.Connect](#) method)
[connection](#) ([airmap.connect.Connect](#) attribute)
 ([airmap.flightAPI.Flight](#) attribute)
 ([airmap.statusAPI.Status](#) attribute)
 ([airmap.telemetryAPI.Telemetry](#) attribute)
[create_FlightPoint\(\)](#) ([airmap.flightAPI.Flight](#) method)

D

[dbgPrint\(\)](#) ([airmap.airdefs.Globals](#) method)
[delete_Flight\(\)](#) ([airmap.flightAPI.Flight](#) method)
[drone](#) (class in [airmap.drone](#))

E

[end_Flight\(\)](#) ([airmap.flightAPI.Flight](#) method)
[enum\(\)](#) ([airmap.airdefs.Advisory.Color](#) method)
 ([airmap.airdefs.Requirement](#) method)
 ([airmap.airdefs.Startup.Drone](#) method)

F

[Flight](#) (class in [airmap.flightAPI](#))

G

[get_Advisories\(\)](#) ([airmap.statusAPI.Status](#) method)
[get_Advisory\(\)](#) ([airmap.statusAPI.Status](#) method)
[get_CIDID\(\)](#) ([airmap.connect.Connect](#) method)
[get_Condition\(\)](#) ([airmap.statusAPI.Status](#) method)
[get_FlightList\(\)](#) ([airmap.flightAPI.Flight](#) method)
[get_Humidity\(\)](#) ([airmap.statusAPI.Status](#) method)
[get_MaxDistance\(\)](#) ([airmap.statusAPI.Status](#) method)
[get_PilotID\(\)](#) ([airmap.flightAPI.Flight](#) method)
[get_Precipitation\(\)](#) ([airmap.statusAPI.Status](#) method)
[get_SecureToken\(\)](#) ([airmap.connect.Connect](#) method)
[get_status\(\)](#) ([airmap.statusAPI.Status](#) method)
[get_StatusCode\(\)](#) ([airmap.statusAPI.Status](#) method)
[get_StatusColor\(\)](#) ([airmap.statusAPI.Status](#) method)
[get_Temperature\(\)](#) ([airmap.statusAPI.Status](#) method)
[get_Visibility\(\)](#) ([airmap.statusAPI.Status](#) method)
[get_WindGusting\(\)](#) ([airmap.statusAPI.Status](#) method)
[get_WindHeading\(\)](#) ([airmap.statusAPI.Status](#) method)
[get_WindSpeed\(\)](#) ([airmap.statusAPI.Status](#) method)
[Globals](#) (class in [airmap.airdefs](#))

H

[headers](#) ([airmap.connect.Connect](#) attribute)
 ([airmap.flightAPI.Flight](#) attribute)
 ([airmap.statusAPI.Status](#) attribute)
 ([airmap.telemetryAPI.Telemetry](#) attribute)
[httpPort](#) ([airmap.airdefs.Globals](#) attribute)
[httpsAddr](#) ([airmap.airdefs.Globals](#) attribute)
[httpsPort](#) ([airmap.airdefs.Globals](#) attribute)

K

[keyAddr](#) ([airmap.airdefs.Globals](#) attribute)

L

[levelDown\(\)](#) ([airmap.statusAPI.Status](#) method)
[localAdvisories](#) ([airmap.statusAPI.Status](#) attribute)
[localLevelDown](#) ([airmap.statusAPI.Status](#) attribute)
[localProperties](#) ([airmap.statusAPI.Status](#) attribute)
[log](#) (class in [airmap.log](#))

M

[myFlightID](#) ([airmap.airdefs.Globals](#) attribute)
[myToken](#) ([airmap.airdefs.Globals](#) attribute)

N

Notify (class in airmap.airdefs)

O

off (airmap.airdefs.Notify attribute)

(airmap.airdefs.Public attribute)

(airmap.airdefs.Weather attribute)

on (airmap.airdefs.Notify attribute)

(airmap.airdefs.Public attribute)

(airmap.airdefs.Weather attribute)

os (airmap.connect.Connect attribute)

(airmap.flightAPI.Flight attribute)

(airmap.statusAPI.Status attribute)

(airmap.telemetryAPI.Telemetry attribute)

P

pilot_id (airmap.airdefs.Globals attribute)

pilotIDValid (airmap.airdefs.Globals attribute)

Properties (class in airmap.airdefs)

Public (class in airmap.airdefs)

put_Telemetry() (airmap.telemetryAPI.Telemetry method)

R

Requirement (class in airmap.airdefs)

S

set_Timeout() (airmap.connect.Connect method)

set_XAPIKey() (airmap.connect.Connect method)

Startup (class in airmap.airdefs)

Startup.Drone (class in airmap.airdefs)

State (airmap.airdefs.Requirement attribute)

(airmap.airdefs.Startup.Drone attribute)

Status (class in airmap.statusAPI)

status_json (airmap.statusAPI.Status attribute)

strPrint() (airmap.airdefs.Globals method)

T

Telemetry (class in airmap.telemetryAPI)

telemetryAddr (airmap.airdefs.Globals attribute)

thisCID (airmap.airdefs.Globals attribute)

thisGlobals (airmap.connect.Connect attribute)

(airmap.flightAPI.Flight attribute)

(airmap.statusAPI.Status attribute)

(airmap.telemetryAPI.Telemetry attribute)

timeOut (airmap.airdefs.Globals attribute)

W

Weather (class in airmap.airdefs)

X

xapikey (airmap.airdefs.Globals attribute)

Python Module Index

a

[airmap](#)

[airmap.airdefs](#)

[airmap.alertsAPI](#)

[airmap.connect](#)

[airmap.drone](#)

[airmap.flightAPI](#)

[airmap.log](#)

[airmap.statusAPI](#)

[airmap.telemetryAPI](#)