AutoClean/Sensors

1

Generated by Doxygen 1.8.17

1 Namespace Index **1**

1.1 Namespace List 1

2 Class Index **3**

2.1 Class List 3

3 File Index **5**

3.1 File List 5

4 Namespace Documentation **7**

4.1 Sensors Namespace Reference 7

4.2 Sensors.sensors Namespace Reference 7

4.2.1 Detailed Description 8

4.2.2 Function Documentation 8

4.2.2.1 export_sensors_to_excel() 8

4.2.3 Variable Documentation 8

4.2.3.1 action 8

4.2.3.2 args 8

4.2.3.3 conn 8

4.2.3.4 cursor 9

4.2.3.5 help 9

4.2.3.6 HISTORY_WRITE 9

4.2.3.7 module_path 9

4.2.3.8 p 9

4.2.3.9 parser 9

4.2.3.10 sensors_info 10

4.2.3.11 TX_Pixhawk_telem 10

4.2.3.12 vehicleConnected 10

4.3 Sensors.telemetry Namespace Reference 10

4.3.1 Detailed Description 10

4.3.2 Function Documentation 10

4.3.2.1 check_none() 10

5 Class Documentation **11**

5.1 Sensors.telemetry.SensorsInfo Class Reference 11

5.1.1 Detailed Description 12

5.1.2 Constructor & Destructor Documentation 12

5.1.2.1 __init__() 12

5.1.3 Member Data Documentation 12

5.1.3.1 alt_ 12

5.1.3.2 groundspeed_ 13

5.1.3.3 heading_ 13

5.1.3.4 last_heartbeat_ 13

5.1.3.5 relative_alt_ 13

5.2 Sensors.telemetry.Telemetry Class Reference 14
5.2.1 Detailed Description 15
5.2.2 Constructor & Destructor Documentation 15
5.2.2.1 __init__() 15
5.2.3 Member Function Documentation 15
5.2.3.1 close() 15
5.2.3.2 initialize() 17
5.2.3.3 is_arm() 18
5.2.3.4 read_channel8() 18
5.2.3.5 read_information() 18
5.2.4 Member Data Documentation 19
5.2.4.1 sitl 19
5.2.4.2 vehicle 19
5.2.4.3 vehicleConnected 19

6 File Documentation **21**
6.1 __init__.py File Reference 21
6.2 sensors.py File Reference 21
6.3 telemetry.py File Reference 22

Index **23**

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Sensors 7
Sensors.sensors
Sensors: reading data out of sensors of the vehicle to sql server using telemetry class 7
Sensors.telemetry
Telemtry: class implementation for reading data out of sensors of the vehicle. optional to connect either to sitl simulation or the real vehicle with uart 10

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Sensors.telemetry.SensorsInfo
Class for sensors info which saves the needed telem info 11
Sensors.telemetry.Telemetry
Class to hold dronekit API vehicle handler and connect to it 14

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

__init__.py 21
sensors.py 21
telemetry.py 22

Chapter 4

Namespace Documentation

4.1 Sensors Namespace Reference

Namespaces

- sensors

  sensors: reading data out of sensors of the vehicle to sql server using telemetry class
- telemetry

  telemtry: class implementation for reading data out of sensors of the vehicle. optional to connect either to sitl simulation or the real vehicle with uart

4.2 Sensors.sensors Namespace Reference

sensors: reading data out of sensors of the vehicle to sql server using telemetry class

Functions

- def export_sensors_to_excel ()

  this function runs as a parallel process in sensors module.

Variables

- module_path = os.path.abspath(os.getcwd())
- parser = argparse.ArgumentParser(description="Drone detection project")

  action chooses true or false hardcoded for the arguments of this module
- action
- help
- args = parser.parse_args()
- int HISTORY_WRITE = 10
- conn
- cursor
- bool vehicleConnected = True
- TX_Pixhawk_telem = telemetry.Telemetry(vehicleConnected)
- p = Process(target=export_sensors_to_excel)
- sensors_info = TX_Pixhawk_telem.read_information()

4.2.1 Detailed Description

sensors: reading data out of sensors of the vehicle to sql server using telemetry class

4.2.2 Function Documentation

4.2.2.1 export_sensors_to_excel()

```
def Sensors.sensors.export_sensors_to_excel ( )
```

this function runs as a parallel process in sensors module.

Exports the sensors sql table to xlsx format using sql_to_excel function in sql_config (possible to change to csv)

Definition at line 34 of file sensors.py.

```
def export_sensors_to_excel():
    conn, cursor = connect_to_db()
    while True:
        sql_to_excel(conn, "SensorsInfo", "sql/SensorsInfo.xlsx")
        time.sleep(1)


```

4.2.3 Variable Documentation

4.2.3.1 action

```
Sensors.sensors.action
```

Definition at line 22 of file sensors.py.

4.2.3.2 args

```
Sensors.sensors.args = parser.parse_args()
```

Definition at line 28 of file sensors.py.

4.2.3.3 conn

```
Sensors.sensors.conn
```

Definition at line 43 of file sensors.py.

4.2.3.4 cursor

```
Sensors.sensors.cursor
```

Definition at line 43 of file sensors.py.

4.2.3.5 help

```
Sensors.sensors.help
```

Definition at line 23 of file sensors.py.

4.2.3.6 HISTORY_WRITE

```
int Sensors.sensors.HISTORY_WRITE = 10
```

Definition at line 30 of file sensors.py.

4.2.3.7 module_path

```
Sensors.sensors.module_path = os.path.abspath(os.getcwd())
```

Definition at line 13 of file sensors.py.

4.2.3.8 p

```
Sensors.sensors.p = Process(target=export_sensors_to_excel)
```

Definition at line 54 of file sensors.py.

4.2.3.9 parser

```
Sensors.sensors.parser = argparse.ArgumentParser(description="Drone detection project")
```

action chooses true or false hardcoded for the arguments of this module

Definition at line 21 of file sensors.py.

4.2.3.10 sensors_info

```
Sensors.sensors.sensors_info = TX_Pixhawk_telem.read_information()
```

Definition at line 59 of file sensors.py.

4.2.3.11 TX_Pixhawk_telem

```
Sensors.sensors.TX_Pixhawk_telem = telemetry.Telemetry(vehicleConnected)
```

Definition at line 51 of file sensors.py.

4.2.3.12 vehicleConnected

```
bool Sensors.sensors.vehicleConnected = True
```

Definition at line 48 of file sensors.py.

4.3 Sensors.telemetry Namespace Reference

telemtry: class implementation for reading data out of sensors of the vehicle. optional to connect either to sitl simulation or the real vehicle with uart

Classes

- class SensorsInfo

    class for sensors info which saves the needed telem info.
- class Telemetry

    class to hold dronekit API vehicle handler and connect to it.

Functions

- def check_none (val)

4.3.1 Detailed Description

telemtry: class implementation for reading data out of sensors of the vehicle. optional to connect either to sitl simulation or the real vehicle with uart

4.3.2 Function Documentation

4.3.2.1 check_none()

```
def Sensors.telemetry.check_none (
              val )
```

Definition at line 12 of file telemetry.py.

```
def check_none(val):
    if val is None:
        return -1
    else:
        return val

```

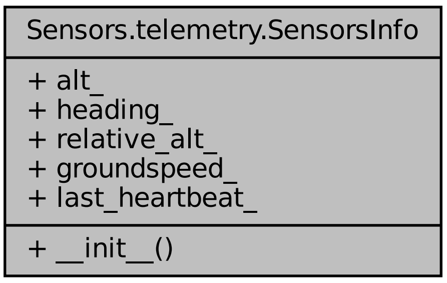
Chapter 5

Class Documentation

5.1 Sensors.telemetry.SensorsInfo Class Reference

class for sensors info which saves the needed telem info.

Collaboration diagram for Sensors.telemetry.SensorsInfo:



Public Member Functions

- def __init__ (self, alt, heading, relative_alt, groundspeed, last_heartbeat)

Public Attributes

- alt_

  altitue of vehicle in meters
- heading_

  current heading in degrees - 0..360, where North = 0 (int).
- relative_alt_
- groundspeed_

  groundspeed of vehicle in meters/second
- last_heartbeat_

  Time since last MAVLink heartbeat was received (in seconds).

5.1.1 Detailed Description

class for sensors info which saves the needed telem info.

might alter information to suit to our needs

Definition at line 20 of file telemetry.py.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 __init__()

```
def Sensors.telemetry.SensorsInfo.__init__ (
                self,
                alt,
                heading,
                relative_alt,
                groundspeed,
                last_heartbeat )
```

Definition at line 21 of file telemetry.py.

```
    def __init__(self, alt, heading, relative_alt, groundspeed, last_heartbeat):

        self.alt_ = alt

        self.heading_ = heading
        self.relative_alt_ = relative_alt

        self.groundspeed_ = groundspeed
        # self.home_location_ = home_location

        self.last_heartbeat_ = last_heartbeat


```

5.1.3 Member Data Documentation

5.1.3.1 alt_

```
Sensors.telemetry.SensorsInfo.alt_
```

altitue of vehicle in meters

Definition at line 23 of file telemetry.py.

5.1.3.2 groundspeed_

```
Sensors.telemetry.SensorsInfo.groundspeed_
```

groundspeed of vehicle in meters/second

Definition at line 28 of file telemetry.py.

5.1.3.3 heading_

```
Sensors.telemetry.SensorsInfo.heading_
```

current heading in degrees - 0..360, where North = 0 (int).

Definition at line 25 of file telemetry.py.

5.1.3.4 last_heartbeat_

```
Sensors.telemetry.SensorsInfo.last_heartbeat_
```

Time since last MAVLink heartbeat was received (in seconds).

The attribute can be used to monitor link activity and implement script-specific timeout handling.

Definition at line 32 of file telemetry.py.

5.1.3.5 relative_alt_

```
Sensors.telemetry.SensorsInfo.relative_alt_
```

Definition at line 26 of file telemetry.py.

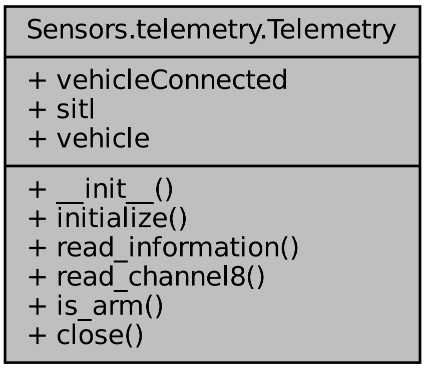
The documentation for this class was generated from the following file:

- telemetry.py

5.2 Sensors.telemetry.Telemetry Class Reference

class to hold dronekit API vehicle handler and connect to it.

Collaboration diagram for Sensors.telemetry.Telemetry:



Public Member Functions

- def __init__ (self, vehicle_connected)

    constructor of telemetry which connects to vehicle or simulation
- def initialize (self, indoor)

    Download the vehicle waypoints (commands) to acquire take-off parameters.
- def read_information (self)

    One time read information from the Pixhawk flight computer.
- def read_channel8 (self)

    Reads CH8IN (the drone operator can signal the TX2 through this channel).
- def is_arm (self)

    Reads armed status of the system (arm is ready to flight - motors are running) Disarm is while on ground.
- def close (self)

    Closing interface & simulation (if there is)

Public Attributes

- vehicleConnected

    holds True if we're running this on a TX2 connected to the Pixhawk by Uart (Telem2) and False otherwise
- sitl

    simulation handle if we don't use the pixhawk itself
- vehicle

    vehicle is a dronekitAPI class for the connection to the vehicle

5.2.1 Detailed Description

class to hold dronekit API vehicle handler and connect to it.

this class connect either to vehicle or simulation and implement methods to read info from pixhawk

Definition at line 38 of file telemetry.py.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 __init__()

```
def Sensors.telemetry.Telemetry.__init__ (
              self,
              vehicle_connected )
```

constructor of telemetry which connects to vehicle or simulation

Parameters

vehicle_connected	True/False

Definition at line 42 of file telemetry.py.

```
    def __init__(self, vehicle_connected):

        self.vehicleConnected = vehicle_connected

        if not self.vehicleConnected:

            self.sitl = dronekit_sitl.start_default()  # (sitl.start)
            connection_string = self.sitl.connection_string()  # now we have the connection string (the
     ip and udp port)

        print("Connecting with the drone")
        if not self.vehicleConnected:

            self.vehicle = connect(connection_string,
                                   wait_ready=True)  # wait_ready flag hold the program until all the
     parameters have been read
        else:
            print("Trying to connect")
            self.vehicle = connect('/dev/ttyTHS2', wait_ready=True, # in j121 it was ttyTHS1
                                   baud=57600)  # this is the name of the Pixhawk/Telem2 as the TX2 sees
     it;
            # the same baud rate as configured in the Pixhawk using Mission Planner
        print("Connection success")


```

5.2.3 Member Function Documentation

5.2.3.1 close()

```
def Sensors.telemetry.Telemetry.close (
              self )
```

Closing interface & simulation (if there is)

Parameters

-	

Returns

-

Definition at line 135 of file telemetry.py.

```
    def close(self):
        self.vehicle.close()
        if not self.vehicleConnected:
            self.sitl.stop()

        print("Telemetry is closed")


```

5.2.3.2 initialize()

```
def Sensors.telemetry.Telemetry.initialize (
                self,
                indoor )
```

Download the vehicle waypoints (commands) to acquire take-off parameters.

Parameters

indoor	True/False

Returns

-

Definition at line 68 of file telemetry.py.

```
    def initialize(self, indoor):
        print("Initialization...")
        if indoor:
            cmds = self.vehicle.commands
            cmds.download()
            cmds.wait_ready()
            print("Indoor initialization completed")
        else:  # if there won't be a GPS signal, we will stuck here (self.vehicle.home_location will be
    None forever)
            num_satellites = 0
            GPSInfo(None, None, None, satellites_visible=num_satellites)
            while not self.vehicle.home_location:
                cmds = self.vehicle.commands
                cmds.download()
                cmds.wait_ready()
                if not self.vehicle.home_location:  # will be `None` until first set by autopilot and
    download completes
                    print("Waiting for home location...")
                    time.sleep(1)
            print("\n Home location: {}".format(self.vehicle.home_location))
            print("GPS info: {}".format(self.vehicle.gps_0))
            print("Visible satellites: {}".format(num_satellites))
            print("Outdoor initialization completed")

```

5.2.3.3 is_arm()

```
def Sensors.telemetry.Telemetry.is_arm (
              self )
```

Reads armed status of the system (arm is ready to flight - motors are running) Disarm is while on ground.

Parameters

-	

Returns

vehicle.armed True/False

Definition at line 114 of file telemetry.py.

```
    def is_arm(self):
        self.vehicle.wait_ready(True)
        return self.vehicle.armed

```

5.2.3.4 read_channel8()

```
def Sensors.telemetry.Telemetry.read_channel8 (
              self )
```

Reads CH8IN (the drone operator can signal the TX2 through this channel).

Used for reset the system

Definition at line 106 of file telemetry.py.

```
    def read_channel8(self):
        self.vehicle.wait_ready(True)
        return self.vehicle.channels['8']

```

5.2.3.5 read_information()

```
def Sensors.telemetry.Telemetry.read_information (
              self )
```

One time read information from the Pixhawk flight computer.

Returns

sensors_info class

Definition at line 93 of file telemetry.py.

```
    def read_information(self):
        self.vehicle.wait_ready(True)  # waits for specified attributes to be populated from the vehicle
    (values are initially None).
        takeoff_alt_barom = self.vehicle.location.global_frame.alt
        sensors_info = SensorsInfo(check_none(self.vehicle.location.global_relative_frame.lat),
                    check_none(self.vehicle.heading),
                    check_none(self.vehicle.location.global_frame.alt - takeoff_alt_barom),
                    check_none(self.vehicle.groundspeed),
                     # check_none(self.vehicle.home_location),
                     check_none(self.vehicle.last_heartbeat)
                     )
        return sensors_info

```

5.2.4 Member Data Documentation

5.2.4.1 sitl

`Sensors.telemetry.Telemetry.sitl`

simulation handle if we don't use the pixhawk itself

Definition at line 48 of file telemetry.py.

5.2.4.2 vehicle

`Sensors.telemetry.Telemetry.vehicle`

vehicle is a dronekitAPI class for the connection to the vehicle

Definition at line 54 of file telemetry.py.

5.2.4.3 vehicleConnected

`Sensors.telemetry.Telemetry.vehicleConnected`

holds True if we're running this on a TX2 connected to the Pixhawk by Uart (Telem2) and False otherwise

Definition at line 44 of file telemetry.py.

The documentation for this class was generated from the following file:

- telemetry.py

Chapter 6

File Documentation

6.1 __init__.py File Reference

Namespaces

- Sensors

6.2 sensors.py File Reference

Namespaces

- Sensors.sensors

  sensors: reading data out of sensors of the vehicle to sql server using telemetry class

Functions

- def Sensors.sensors.export_sensors_to_excel ()

  this function runs as a parallel process in sensors module.

Variables

- Sensors.sensors.module_path = os.path.abspath(os.getcwd())
- Sensors.sensors.parser = argparse.ArgumentParser(description="Drone detection project")

  action chooses true or false hardcoded for the arguments of this module
- Sensors.sensors.action
- Sensors.sensors.help
- Sensors.sensors.args = parser.parse_args()
- int Sensors.sensors.HISTORY_WRITE = 10
- Sensors.sensors.conn
- Sensors.sensors.cursor
- bool Sensors.sensors.vehicleConnected = True
- Sensors.sensors.TX_Pixhawk_telem = telemetry.Telemetry(vehicleConnected)
- Sensors.sensors.p = Process(target=export_sensors_to_excel)
- Sensors.sensors.sensors_info = TX_Pixhawk_telem.read_information()

6.3 telemetry.py File Reference

Classes

- class Sensors.telemetry.SensorsInfo

  class for sensors info which saves the needed telem info.
- class Sensors.telemetry.Telemetry

  class to hold dronekit API vehicle handler and connect to it.

Namespaces

- Sensors.telemetry

  telemtry: class implementation for reading data out of sensors of the vehicle. optional to connect either to sitl simulation or the real vehicle with uart

Functions

- def Sensors.telemetry.check_none (val)

Index

__init__
    Sensors.telemetry.SensorsInfo, 12
    Sensors.telemetry.Telemetry, 15
__init__.py, 21

action
    Sensors.sensors, 8
alt_
    Sensors.telemetry.SensorsInfo, 12
args
    Sensors.sensors, 8

check_none
    Sensors.telemetry, 10
close
    Sensors.telemetry.Telemetry, 15
conn
    Sensors.sensors, 8
cursor
    Sensors.sensors, 8

export_sensors_to_excel
    Sensors.sensors, 8

groundspeed_
    Sensors.telemetry.SensorsInfo, 12

heading_
    Sensors.telemetry.SensorsInfo, 13
help
    Sensors.sensors, 9
HISTORY_WRITE
    Sensors.sensors, 9

initialize
    Sensors.telemetry.Telemetry, 17
is_arm
    Sensors.telemetry.Telemetry, 17

last_heartbeat_
    Sensors.telemetry.SensorsInfo, 13

module_path
    Sensors.sensors, 9

p
    Sensors.sensors, 9
parser
    Sensors.sensors, 9

read_channel8
    Sensors.telemetry.Telemetry, 18
read_information
    Sensors.telemetry.Telemetry, 18
relative_alt_
    Sensors.telemetry.SensorsInfo, 13

Sensors, 7
sensors.py, 21
Sensors.sensors, 7
    action, 8
    args, 8
    conn, 8
    cursor, 8
    export_sensors_to_excel, 8
    help, 9
    HISTORY_WRITE, 9
    module_path, 9
    p, 9
    parser, 9
    sensors_info, 9
    TX_Pixhawk_telem, 10
    vehicleConnected, 10
Sensors.telemetry, 10
    check_none, 10
Sensors.telemetry.SensorsInfo, 11
    __init__, 12
    alt_, 12
    groundspeed_, 12
    heading_, 13
    last_heartbeat_, 13
    relative_alt_, 13
Sensors.telemetry.Telemetry, 14
    __init__, 15
    close, 15
    initialize, 17
    is_arm, 17
    read_channel8, 18
    read_information, 18
    sitl, 19
    vehicle, 19
    vehicleConnected, 19
sensors_info
    Sensors.sensors, 9
sitl
    Sensors.telemetry.Telemetry, 19

telemetry.py, 22
TX_Pixhawk_telem
    Sensors.sensors, 10

vehicle
  Sensors.telemetry.Telemetry, 19
vehicleConnected
  Sensors.sensors, 10
  Sensors.telemetry.Telemetry, 19