# Raspberry Pi

Generated by Doxygen 1.9.6

# Chapter 1

# Namespace Index

## 1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

# Chapter 2

# Hierarchical Index

## 2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

# Chapter 3

# Class Index

## 3.1 Class List

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

# Chapter 4

# File Index

## 4.1 File List

Here is a list of all files with brief descriptions:

# Chapter 5

# Namespace Documentation

## 5.1 core_node Namespace Reference

### Classes

- class FollowAlgorithm

### Functions

- def main (args=None)

### Variables

- Users C :\Users\sin_p\OneDrive - USN\Dokumenter\Bachelor\Doxygen\Dokumentasjon\Doxygen\↩ Pi>doxypypy -a -c "C:/Users/sin_p/OneDrive - USN/Dokumenter/Bachelor/Doxygen/Dokumentasjon/Doxygen/Pi/core↩ _node.py"

### 5.1.1 Function Documentation

#### 5.1.1.1 main()

```
def core_node.main (
              args = None )
```

### 5.1.2 Variable Documentation

#### 5.1.2.1 C

```
Users core_node.C :\Users\sin_p\OneDrive - USN\Dokumenter\Bachelor\Doxygen\Dokumentasjon\↩
Doxygen\Pi>doxypypy -a -c "C:/Users/sin_p/OneDrive - USN/Dokumenter/Bachelor/Doxygen/Dokumentasjon/Doxygen/Pi
_node.py"
```

## 5.2 qtm_logger Namespace Reference

### Classes

- class Listener

  *Listener node that subscribes to drone_pose topic and logs data into a file.*

### Functions

- def main (args=None)

  *The main function to initialize and run the listener node.*
- def exit_handler (Listener node)

  *Function to handle the exit of the program.*

### Variables

- Users C :\Users\sin_p\OneDrive - USN\Dokumenter\Bachelor\Doxygen\Dokumentasjon\Doxygen\↩ Pi>doxypypy -a -c "C:/Users/sin_p/OneDrive - USN/Dokumenter/Bachelor/Doxygen/Dokumentasjon/Doxygen/Pi/qtm↩ _logger.py"
- time time_prev = time.time()

### 5.2.1 Function Documentation

#### 5.2.1.1 exit_handler()

```
def qtm_logger.exit_handler (
            Listener node )
```

Function to handle the exit of the program.

This function closes the log file and logs that the file is closed.

**Parameters**

| | |
|---|---|
| *node* | The Listener node object. |

#### 5.2.1.2 main()

```
def qtm_logger.main (
              args = None )
```

The main function to initialize and run the listener node.

**Parameters**

| | |
|---|---|
| *args* | Command-line arguments passed to the script. |

### 5.2.2 Variable Documentation

#### 5.2.2.1 C

```
Users qtm_logger.C :\Users\sin_p\OneDrive - USN\Dokumenter\Bachelor\Doxygen\Dokumentasjon\↩
Doxygen\Pi>doxypypy -a -c "C:/Users/sin_p/OneDrive - USN/Dokumenter/Bachelor/Doxygen/Dokumentasjon/Doxygen/Pi
_logger.py"
```

#### 5.2.2.2 time_prev

```
time qtm_logger.time_prev = time.time()
```

## 5.3 qtm_node Namespace Reference

### Classes

- class Qualisys_node
- class UdpProtocol

  *A callable asyncio Datagram Protocol implementation.*

### Functions

- def main (args=None)

## Variables

- Users C :\Users\sin_p\OneDrive - USN\Dokumenter\Bachelor\Doxygen\Dokumentasjon\Doxygen\↩ Pi>doxypypy -a -c "C:/Users/sin_p/OneDrive - USN/Dokumenter/Bachelor/Doxygen/Dokumentasjon/Doxygen/Pi/qtm↩ _node.py"
- int q_freq = 0

  *GLOBAL VARIABLES###.*
- None prev_vel = None
- bool newPose = False

  *GAZEBO GLOBAL VARIABLES###.*
- int non_published_packages = 0

  *QUALISYS GLOBAL VARIABLES###.*
- None prev_msg = None
- str qualisys_ip = '192.168.1.103'
- x
- y
- z
- data

### 5.3.1 Function Documentation

#### 5.3.1.1 main()

```
def qtm_node.main (
              args = None )
```

### 5.3.2 Variable Documentation

#### 5.3.2.1 C

```
Users qtm_node.C :\Users\sin_p\OneDrive - USN\Dokumenter\Bachelor\Doxygen\Dokumentasjon\↩
Doxygen\Pi>doxypypy -a -c "C:/Users/sin_p/OneDrive - USN/Dokumenter/Bachelor/Doxygen/Dokumentasjon/Doxygen/Pi
_node.py"
```

#### 5.3.2.2 data

```
qtm_node.data
```

#### 5.3.2.3 newPose

```
bool qtm_node.newPose = False
```

GAZEBO GLOBAL VARIABLES###.

#### 5.3.2.4 non_published_packages

```
int qtm_node.non_published_packages = 0
```

QUALISYS GLOBAL VARIABLES###.

#### 5.3.2.5 prev_msg

```
DronePose qtm_node.prev_msg = None
```

#### 5.3.2.6 prev_vel

```
list qtm_node.prev_vel = None
```

#### 5.3.2.7 q_freq

```
int qtm_node.q_freq = 0
```

GLOBAL VARIABLES###.

#### 5.3.2.8 qualisys_ip

```
str qtm_node.qualisys_ip = '192.168.1.103'
```

#### 5.3.2.9 x

```
qtm_node.x
```

#### 5.3.2.10 y

```
qtm_node.y
```

#### 5.3.2.11 z

```
qtm_node.z
```

## 5.4 util Namespace Reference

### Functions

- def calculatePosition2 (positions, current_yaw)

  *Mock function for position calculation, currently returns a constant value.*
- def calculatePosition (positions, current_yaw)

  *Function to calculate the drone's position and velocity.*
- def angle_diff (angle_1, angle_2)

  *Compute the difference between two angles, handling wraparound.*
- def calculate_yaw (rot_matrix)

  *Function to calculate yaw angle from a rotation matrix.*
- def rotation_matrix (yaw, pitch, roll)

  *Function to calculate a rotation matrix given yaw, pitch, and roll.*

### Variables

- Users C :\Users\sin_p\OneDrive - USN\Dokumenter\Bachelor\Doxygen\Dokumentasjon\Doxygen\↩ Pi>doxypypy -a -c "C:/Users/sin_p/OneDrive - USN/Dokumenter/Bachelor/Doxygen/Dokumentasjon/Doxygen/Pi/util.↩ py"
- time start_time_ns = time.time_ns()
- time start_time = time.time()
- int time_prev = 0
- np prev_vel = np.zeros(3)
- int yaw = 0
- bool data_failure_prev = True

### 5.4.1 Function Documentation

#### 5.4.1.1 angle_diff()

```
def util.angle_diff (
              angle_1,
              angle_2 )
```

Compute the difference between two angles, handling wraparound.

**Parameters**

| | |
|---|---|
| *angle_1* | The first angle, in radians. |
| *angle_2* | The second angle, in radians. |

**Returns**

The difference between the two angles, in radians, wrapped to [-pi, pi].

#### 5.4.1.2 calculate_yaw()

```
def util.calculate_yaw (
              rot_matrix )
```

Function to calculate yaw angle from a rotation matrix.

Returns None if we have a failure at any point.

**Parameters**

| | |
|---|---|
| *rot_matrix* | A 3x3 rotation matrix. |

**Returns**

The calculated yaw angle, or None if a calculation error occurred.

#### 5.4.1.3 calculatePosition()

```
def util.calculatePosition (
              positions,
              current_yaw )
```

Function to calculate the drone's position and velocity.

**Parameters**

| | |
|---|---|
| *positions* | The positions of the drone. |
| *current_yaw* | The current yaw angle of the drone. |

**Returns**

Returns a DronePose object containing position, velocity, acceleration, yaw, and yaw velocity.

#### 5.4.1.4 calculatePosition2()

```
def util.calculatePosition2 (
              positions,
              current_yaw )
```

Mock function for position calculation, currently returns a constant value.

This code is borrowed from Hilde Marie Moholt.

**Parameters**

| | |
|---|---|
| *positions* | The positions of the drone. |
| *current_yaw* | The current yaw angle of the drone. |

**Returns**

Returns a constant value of 2.

#### 5.4.1.5 rotation_matrix()

```
def util.rotation_matrix (
              yaw,
              pitch,
              roll )
```

Function to calculate a rotation matrix given yaw, pitch, and roll.

This function is not used in qualisys, but is used to test qualisys.

**Parameters**

| | |
|---|---|
| *yaw* | The yaw angle. |
| *pitch* | The pitch angle. |
| *roll* | The roll angle. |

**Returns**

A 3x3 rotation matrix.

### 5.4.2 Variable Documentation

#### 5.4.2.1 C

```
Users util.C :\Users\sin_p\OneDrive - USN\Dokumenter\Bachelor\Doxygen\Dokumentasjon\Doxygen\↩
Pi>doxypypy -a -c "C:/Users/sin_p/OneDrive - USN/Dokumenter/Bachelor/Doxygen/Dokumentasjon/Doxygen/Pi/util.↩
py"
```

#### 5.4.2.2 data_failure_prev

```
bool util.data_failure_prev = True
```

#### 5.4.2.3 prev_vel

```
np util.prev_vel = np.zeros(3)
```

#### 5.4.2.4 start_time

```
time util.start_time = time.time()
```

#### 5.4.2.5 start_time_ns

```
time util.start_time_ns = time.time_ns()
```

#### 5.4.2.6 time_prev

```
int util.time_prev = 0
```

#### 5.4.2.7 yaw

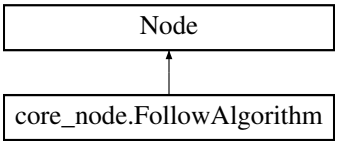
```
int util.yaw = 0
```

# Chapter 6

# Class Documentation

## 6.1 core_node.FollowAlgorithm Class Reference

Inheritance diagram for core_node.FollowAlgorithm:




### Public Member Functions

- def __init__ (self)

  *Initializes the FollowAlgorithm class.*
- def qualisys_callback (self, DronePose msg)

  *Callback function for 'drone_pose' topic.*
- def find_drone_position (self, yaw, distance_to_object, object_x, object_y, object_z, camera_angle)

  *Calculates the drone's position based on received data.*
- def plotter (self, drone_pos, qualisys_pos)

  *Updates the plot with the drone's and qualisys's positions.*
- def save_plot (self)

  *Saves the 3D plot as an image.*
- def position_and_distance_callback (self, Int32MultiArray msg)

  *Callback function for 'object_pos_and_distance' topic.*

### Public Attributes

- distance
- fig
- drone_x_arr
- drone_y_arr
- drone_z_arr
- qualisys_x_arr
- qualisys_y_arr
- qualisys_z_arr
- counter

### 6.1.1 Constructor & Destructor Documentation

#### 6.1.1.1 __init__()

```
def core_node.FollowAlgorithm.__init__ (
              self )
```

Initializes the FollowAlgorithm class.

```
   This class is a ROS2 node that performs a follow algorithm for a drone.
   It subscribes to 'object_pos_and_distance' and 'drone_pose' topics.
   It also initializes arrays and counters for plotting.
```

### 6.1.2 Member Function Documentation

#### 6.1.2.1 find_drone_position()

```
def core_node.FollowAlgorithm.find_drone_position (
              self,
              yaw,
              distance_to_object,
              object_x,
              object_y,
              object_z,
              camera_angle )
```

Calculates the drone's position based on received data.

**Parameters**

| | |
|---|---|
| *yaw* | Yaw angle of the drone in radians. |
| *distance_to_object* | Distance from the drone to the object in meters. |
| *object_x* | X position of the object in meters. |
| *object_y* | Y position of the object in meters. |
| *object_z* | Z position of the object in meters. |
| *camera_angle* | Angle of the camera in degrees. |

**Returns**

Tuple containing the drone's calculated

#### 6.1.2.2 plotter()

```
def core_node.FollowAlgorithm.plotter (
              self,
              drone_pos,
              qualisys_pos )
```

Updates the plot with the drone's and qualisys's positions.

**Parameters**

| | |
|---|---|
| *drone_pos* | Tuple containing the drone's position (x, y, z). |
| *qualisys_pos* | Tuple containing the qualisys's position (x, y, z). |

#### 6.1.2.3 position_and_distance_callback()

```
def core_node.FollowAlgorithm.position_and_distance_callback (
              self,
             Int32MultiArray msg )
```

Callback function for 'object_pos_and_distance' topic.

```
This function is triggered when a message is received on the 'object_pos_and_distance' topic.
It extracts the distance to the object from the received message and performs logging.
```

**Parameters**

| | |
|---|---|
| *msg* | Int32MultiArray message received from the 'object_pos_and_distance' topic. |

#### 6.1.2.4 qualisys_callback()

```
def core_node.FollowAlgorithm.qualisys_callback (
              self,
             DronePose msg )
```

Callback function for 'drone_pose' topic.

```
This function is triggered when a message is received on the 'drone_pose' topic.
It calculates the drone's estimated position based on received data and performs logging and plotting.
```

**Parameters**

| | |
|---|---|
| *msg* | DronePose message received from the 'drone_pose' topic. |

#### 6.1.2.5 save_plot()

```
def core_node.FollowAlgorithm.save_plot (
              self )
```

Saves the 3D plot as an image.

### 6.1.3 Member Data Documentation

#### 6.1.3.1 counter

```
core_node.FollowAlgorithm.counter
```

#### 6.1.3.2 distance

```
core_node.FollowAlgorithm.distance
```

#### 6.1.3.3 drone_x_arr

```
core_node.FollowAlgorithm.drone_x_arr
```

#### 6.1.3.4 drone_y_arr

```
core_node.FollowAlgorithm.drone_y_arr
```

#### 6.1.3.5 drone_z_arr

```
core_node.FollowAlgorithm.drone_z_arr
```

#### 6.1.3.6 fig

```
core_node.FollowAlgorithm.fig
```

#### 6.1.3.7 qualisys_x_arr

```
core_node.FollowAlgorithm.qualisys_x_arr
```

#### 6.1.3.8 qualisys_y_arr

```
core_node.FollowAlgorithm.qualisys_y_arr
```

#### 6.1.3.9 qualisys_z_arr

```
core_node.FollowAlgorithm.qualisys_z_arr
```

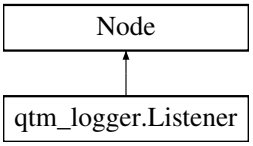
The documentation for this class was generated from the following file:

- core_node.py

## 6.2 qtm_logger.Listener Class Reference

Listener node that subscribes to drone_pose topic and logs data into a file.

Inheritance diagram for qtm_logger.Listener:




### Public Member Functions

- def __init__ (self)

  *Initialize the listener node, subscription, and log file.*
- def callback (self, data)

  *Callback function to process DronePose messages.*

## Public Attributes

- subscription
- file

### 6.2.1 Detailed Description

Listener node that subscribes to drone_pose topic and logs data into a file.

The logged data includes time, position, velocity, acceleration, and yaw of the drone.

### 6.2.2 Constructor & Destructor Documentation

#### 6.2.2.1 __init__()

```
def qtm_logger.Listener.__init__ (
              self )
```

Initialize the listener node, subscription, and log file.

### 6.2.3 Member Function Documentation

#### 6.2.3.1 callback()

```
def qtm_logger.Listener.callback (
              self,
              data )
```

Callback function to process DronePose messages.

```
   This function logs the data into the log file.
```

**Parameters**

| | |
|---|---|
| *data* | The received DronePose message. |

### 6.2.4 Member Data Documentation

#### 6.2.4.1 file

```
qtm_logger.Listener.file
```

#### 6.2.4.2 subscription

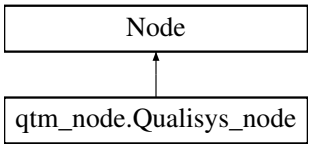
```
qtm_logger.Listener.subscription
```

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

- qtm_logger.py

## 6.3 qtm_node.Qualisys_node Class Reference

Inheritance diagram for qtm_node.Qualisys_node:




### Public Member Functions

- def __init__ (self)

  *Initializer for the Qualisys_node class.*
- def talker (self, data)

  *Function for publishing data to the 'drone_pose' topic.*
- def create_msg (self, x, y, z, v_x, v_y, v_z, a_x, a_y, a_z, yaw, v_yaw, freq, full_msg)

  *Function to create a new DronePose message and store the previous one for calculations.*
- def calculate_vel_a (self, position, freq, yaw)

  *Function to calculate the velocity, acceleration, and yaw velocity from position data.*
- def calc (self, position, rot, freq)

  *Function to calculate the real frequency and call calculate_vel_a() with the position, frequency, and yaw.*
- def create_body_index (self, xml_string)

  *Function to extract a name to index dictionary from 6-DOF settings xml.*
- def get_freq (self, xml_string)

  *Function to get the frequency of data packets from an xml string.*
- def qtmMain (self)

  *Main asynchronous function for the Qualisys node.*

### Public Attributes

- pub

### 6.3.1 Constructor & Destructor Documentation

#### 6.3.1.1 __init__()

```
def qtm_node.Qualisys_node.__init__ (
              self )
```

Initializer for the Qualisys_node class.

Creates a publisher for DronePose messages on the 'drone_pose' topic.

### 6.3.2 Member Function Documentation

#### 6.3.2.1 calc()

```
def qtm_node.Qualisys_node.calc (
              self,
              position,
              rot,
              freq )
```

Function to calculate the real frequency and call calculate_vel_a() with the position, frequency, and yaw.

**Parameters**

| | |
|---|---|
| *position* | The current position. |
| *rot* | The current rotation. |
| *freq* | The frequency of the data. |

#### 6.3.2.2 calculate_vel_a()

```
def qtm_node.Qualisys_node.calculate_vel_a (
              self,
              position,
              freq,
              yaw )
```

Function to calculate the velocity, acceleration, and yaw velocity from position data.

It uses the previous message to calculate these values.

**Parameters**

| | |
|---|---|
| *position* | The current position. |
| *freq* | The frequency of the data. |
| *yaw* | The current yaw. |

**Returns**

A tuple containing whether the message was published and whether the position was saved.

#### 6.3.2.3 create_body_index()

```
def qtm_node.Qualisys_node.create_body_index (
              self,
              xml_string )
```

Function to extract a name to index dictionary from 6-DOF settings xml.

**Parameters**

| | |
|---|---|
| *xml_string* | The xml string containing the 6-DOF settings. |

**Returns**

A dictionary mapping body names to indices.

#### 6.3.2.4 create_msg()

```
def qtm_node.Qualisys_node.create_msg (
              self,
              x,
              y,
              z,
              v_x,
              v_y,
              v_z,
              a_x,
              a_y,
              a_z,
              yaw,
              v_yaw,
              freq,
              full_msg )
```

Function to create a new DronePose message and store the previous one for calculations.

It will not publish the message if full_msg is False.

**Parameters**

| | |
|---|---|
| *x* | The x position. |
| *y* | The y position. |
| *z* | The z position. |
| *v_x* | The x velocity. |
| *v_y* | The y velocity. |
| *v_z* | The z velocity. |
| *a_x* | The x acceleration. |
| *a_y* | The y acceleration. |
| *a_z* | The z acceleration. |
| *yaw* | The yaw. |
| *v_yaw* | The yaw velocity. |
| *freq* | The frequency of the data. |
| *full_msg* | Whether the message is complete and should be published. |

**Returns**

A tuple containing whether the message was published and whether the position was saved.

#### 6.3.2.5 get_freq()

```
def qtm_node.Qualisys_node.get_freq (
              self,
              xml_string )
```

Function to get the frequency of data packets from an xml string.

**Parameters**

| | |
|---|---|
| *xml_string* | The xml string containing the frequency. |

**Returns**

The frequency.

#### 6.3.2.6 qtmMain()

```
def qtm_node.Qualisys_node.qtmMain (
              self )
```

Main asynchronous function for the Qualisys node.

This function handles the connection to the Qualisys server and the streaming of data.

#### 6.3.2.7 talker()

```
def qtm_node.Qualisys_node.talker (
              self,
              data )
```

Function for publishing data to the 'drone_pose' topic.

**Parameters**

| | |
|---|---|
| *data* | The data to be published. |

### 6.3.3 Member Data Documentation

#### 6.3.3.1 pub

```
qtm_node.Qualisys_node.pub
```

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

- qtm_node.py

## 6.4 qtm_node.UdpProtocol Class Reference

A callable asyncio Datagram Protocol implementation.

### Public Member Functions

- def __init__ (self)
  *Initializer function for the UdpProtocol class.*
- def connection_made (self, transport)
  *Callback for when a connection is made.*
- def datagram_received (self, data, addr)
  *Callback for when a datagram is received.*
- def connection_lost (self, transport)
  *Callback for when a connection is lost.*
- def error_received (self, exc)
  *Callback for when an error is received.*
- def recvfrom (self)
  *Asynchronous function to get data from the queue.*

### Public Attributes

- packets

### 6.4.1 Detailed Description

A callable asyncio Datagram Protocol implementation.

For robotics programming purpose, I need this protocol does last-come-first-serve. This code is borrowed from Hilde Marie Moholt, and modified to fit ros2 by Aerial Edge.

**Reference**

https://docs.python.org/3/library/asyncio-protocol.html#datagram-protocols https://docs.python.org/3/library/asyncio-eventloop.html#asyncio.↩ loop.create_datagram_endpoint https://stackoverflow.com/questions/46140556/proper-wa

### 6.4.2 Constructor & Destructor Documentation

#### 6.4.2.1 __init__()

```
def qtm_node.UdpProtocol.__init__ (
              self )
```

Initializer function for the UdpProtocol class.

Initializes an asyncio queue.

### 6.4.3 Member Function Documentation

#### 6.4.3.1 connection_lost()

```
def qtm_node.UdpProtocol.connection_lost (
              self,
              transport )
```

Callback for when a connection is lost.

Currently, it only prints a message to the console.

**Parameters**

| | |
|---|---|
| *transport* | The transport layer that lost the connection. |

#### 6.4.3.2 connection_made()

```
def qtm_node.UdpProtocol.connection_made (
              self,
              transport )
```

Callback for when a connection is made.

Currently, it only prints a message to the console.

**Parameters**

| | |
|---|---|
| *transport* | The transport layer responsible for the connection. |

#### 6.4.3.3 datagram_received()

```
def qtm_node.UdpProtocol.datagram_received (
              self,
              data,
              addr )
```

Callback for when a datagram is received.

Clears the current queue and puts the new data in it.

**Parameters**

| | |
|---|---|
| *data* | The received data. |
| *addr* | The address of the sender. |

#### 6.4.3.4 error_received()

```
def qtm_node.UdpProtocol.error_received (
              self,
              exc )
```

Callback for when an error is received.

Currently, it does nothing.

**Parameters**

| | |
|---|---|
| *exc* | The exception that was raised. |

#### 6.4.3.5 recvfrom()

```
def qtm_node.UdpProtocol.recvfrom (
              self )
```

Asynchronous function to get data from the queue.

**Returns**

The data from the queue.

### 6.4.4 Member Data Documentation

#### 6.4.4.1 packets

```
qtm_node.UdpProtocol.packets
```

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

- qtm_node.py

# Chapter 7

# File Documentation

## 7.1 core_node.py File Reference

### Classes

- class core_node.FollowAlgorithm

### Namespaces

- namespace core_node

### Functions

- def core_node.main (args=None)

### Variables

- Users core_node.C :\Users\sin_p\OneDrive - USN\Dokumenter\Bachelor\Doxygen\Dokumentasjon\↩ Doxygen\Pi>doxypypy -a -c "C:/Users/sin_p/OneDrive - USN/Dokumenter/Bachelor/Doxygen/Dokumentasjon/Doxygen/Pi/core _node.py"

## 7.2 qtm_logger.py File Reference

### Classes

- class qtm_logger.Listener

  *Listener node that subscribes to drone_pose topic and logs data into a file.*

### Namespaces

- namespace qtm_logger

### Functions

- def qtm_logger.main (args=None)

  *The main function to initialize and run the listener node.*
- def qtm_logger.exit_handler (Listener node)

  *Function to handle the exit of the program.*

### Variables

- Users qtm_logger.C :\Users\sin_p\OneDrive - USN\Dokumenter\Bachelor\Doxygen\Dokumentasjon\↩ Doxygen\Pi>doxypypy -a -c "C:/Users/sin_p/OneDrive - USN/Dokumenter/Bachelor/Doxygen/Dokumentasjon/Doxygen/Pi/qtm↩ _logger.py"
- time qtm_logger.time_prev = time.time()

## 7.3 qtm_node.py File Reference

### Classes

- class qtm_node.UdpProtocol

  *A callable asyncio Datagram Protocol implementation.*
- class qtm_node.Qualisys_node

### Namespaces

- namespace qtm_node

### Functions

- def qtm_node.main (args=None)

### Variables

- Users qtm_node.C :\Users\sin_p\OneDrive - USN\Dokumenter\Bachelor\Doxygen\Dokumentasjon\↩ Doxygen\Pi>doxypypy -a -c "C:/Users/sin_p/OneDrive - USN/Dokumenter/Bachelor/Doxygen/Dokumentasjon/Doxygen/Pi/qtm↩ _node.py"
- int qtm_node.q_freq = 0

  *GLOBAL VARIABLES###.*
- None qtm_node.prev_vel = None
- bool qtm_node.newPose = False

  *GAZEBO GLOBAL VARIABLES###.*
- int qtm_node.non_published_packages = 0

  *QUALISYS GLOBAL VARIABLES###.*
- None qtm_node.prev_msg = None
- str qtm_node.qualisys_ip = '192.168.1.103'
- qtm_node.x
- qtm_node.y
- qtm_node.z
- qtm_node.data

# 7.4 util.py File Reference

## Namespaces

- namespace util

## Functions

- def util.calculatePosition2 (positions, current_yaw)

  *Mock function for position calculation, currently returns a constant value.*
- def util.calculatePosition (positions, current_yaw)

  *Function to calculate the drone's position and velocity.*
- def util.angle_diff (angle_1, angle_2)

  *Compute the difference between two angles, handling wraparound.*
- def util.calculate_yaw (rot_matrix)

  *Function to calculate yaw angle from a rotation matrix.*
- def util.rotation_matrix (yaw, pitch, roll)

  *Function to calculate a rotation matrix given yaw, pitch, and roll.*

## Variables

- Users util.C :\Users\sin_p\OneDrive - USN\Dokumenter\Bachelor\Doxygen\Dokumentasjon\Doxygen\↩ Pi>doxypypy -a -c "C:/Users/sin_p/OneDrive - USN/Dokumenter/Bachelor/Doxygen/Dokumentasjon/Doxygen/Pi/util.↩ py"
- time util.start_time_ns = time.time_ns()
- time util.start_time = time.time()
- int util.time_prev = 0
- np util.prev_vel = np.zeros(3)
- int util.yaw = 0
- bool util.data_failure_prev = True

# Index