API Documentation

API Documentation

August 25, 2015

Contents

1 Module thymiolib

1.1 Variables

Name	Description
__package__	**Value:** `None`

1.2 Class ThymioController

This class allows the user to control all leds, motors and sensors of a thymio

1.2.1 Methods

__init__(*self*, *ip*=`'localhost'`, *port*=`3000`, *debug*=`False`)

Create a thymio controller with an ip address and a given port

Parameters

`ip`: asebahttp's ip
(type=string)

`port`: asebahttp's port
(type=int)

`debug`: true if debug mode is active (print detail about errors)
(type=bool)

get_motors(*self*)

Get real speed for each motor

This returns the speed of both thymio motors.

Return Value

A list of two values containing the left and right motor values
(type=list of int [left, right] (empty list in case of error))

Note:

```
>>> t.get_motors()
[-52, 137]
```

set_motors(*self*, *left*, *right*)

Change target speed for each motor

This changes the speed of both motors and return True on success.

Parameters

`left`: Left motor speed

(type=int (between -500 and 500))

`right`: Right motor speed

(type=int (between -500 and 500))

Return Value

True if the method terminated correctly

(type=bool)

Note:

```
>>> t.set_motors(500, -500)
True
>>> t.set_motors(-1000, 0)
False
>>> t.set_motors(0, 0)
True
```

get_prox_h(*self*)

Get values for horizontal ir sensor.

This returns the values of each horizontal ir sensor.

Return Value

A value between 0 and 5000 for each of 7 sensors. The higher the value, the closer an object

[0-4]: front sensors left to right [5-6]: back sensors left to right

(type=list of int (empty list in case of error))

Note:

```
>>> t.get_prox_h()
[0, 2701, 2579, 2489, 0, 3632, 4490]
```

get_prox_v(*self*)

Get values for vertical (ground) ir sensor.

This returns the values of each vertical ir sensor.

Return Value

A value between 0 and 1000 for each of 2 sensors. The higher the value, the lighter the ground.

[0-1]: ground sensors left to right

(type=list of int (empty list in case of error))

Note:

```
>>> t.get_prox_v()
[980,240]
```

get_acc(*self*)

Get values of accelerometer.

This returns the values of each axis.

Return Value

A value for each of 3 axis.

[roll,pitch,yaw]

(type=list of int (empty list in case of error))

Note:

```
>>> t.get_acc()
[1,-1,23]
```

get_button_backward(*self*)

Get value of backward button.

This returns the current state of the button

Return Value

1 if button is pressed, 0 otherwise; -1 in case of error

(type=int)

Note:

```
>>> t.get_button_backward()
0
```

get_button_center(*self*)

Get value of center button.

This returns the current state of the button

@@return: 1 if button is pressed, 0 otherwise; -1 in case of error

Return Value

int

Note:

```
>>> t.get_button_center()
1
```

get_button_forward(*self*)

Get value of forward button.

This returns the current state of the button

@@return: 1 if button is pressed, 0 otherwise; -1 in case of error

Return Value

int

Note:

```
>>> t.get_button_forward()
0
```

get_button_left(*self*)

Get value of left button.

This returns the current state of the button

@@return: 1 if button is pressed, 0 otherwise; -1 in case of error

Return Value

int

Note:

```
>>> t.get_button_left()
1
```

get_button_right(*self*)

Get value of right button.

This returns the current state of the button

@@return: 1 if button is pressed, 0 otherwise; -1 in case of error

Return Value

int

Note:

```
>>> t.get_button_right()
0
```

get_mic_intensity(*self*)

Get value of mic.

This returns the intensity of mic.

Return Value

Between 0 and 255: the intensit;. -1 in case of error

(type=int)

Note:

```
>>> t.get_mic_intensity()
129
```

get_temperature(*self*)

Get value of temperature.

This returns the temperature of sensor

Return Value

Temperature in tenths of a degree Celsius; -1 in case of error

(type=int)

Note:

```
>>> t.get_temperature()
312
```

get_rc_last_command(*self*)

Get last rc command received.

Return Value

Command number (between 0 and 127); -1 in case of error

(type=int)

Note:

```
>>> t.get_rc_last_command()
80
```

get_rc_last_address(*self*)

Get last rc address received.

Return Value

Address number (between 0 and 31). -1 in case of error

(type=int)

Note:

```
>>> t.get_rc_last_address()
0
```

get_rc_new(*self*)

Get if new rc code received.

Return Value

1 if new rc code received, 0 otherwise; -1 in case of error

(type=int)

Note:

```
>>> t.get_temperature()
129
```

set_sound_system(*self*, *sound*)

Set a sound system

Parameters

`sound`: sound to play: -1: stop playing sound 0: startup sound 1: shutdown sound 2: arrow button sound 3: central button sound 4: free-fall (scary) sound 5: collision sound 6: target ok for friendly behaviour 7: target detect for friendly behaviour

(type=int)

Return Value

True if the method terminated correctly

(type=bool)

Note:

```
>>> set_sound_system(-1)
True
>>> set_sound_system(20)
False
```

set_sound_freq(*self*, *freq*, *ds*)

Set a sound frequency

Parameters

`freq`: sound frequency (Hz)

(type=float (between 0 and 7812.5))

`ds`: sound duration in 1/60s. Specifying a 0 duration plays the sound continuously and specifying a -1 duration stops the sound.

(type=float)

Return Value

True if the method terminated correctly

(type=bool)

Note:

```
>>> set_sound_freq(200, 60)
True
```

set_led_top(*self, r, g, b*)

Set color for led on top

Parameters

`r`: value of red (0-32)

(type=int)

`g`: value of green (0-32)

(type=int)

`b`: value of blue (0-32)

(type=int)

Return Value

True if the method terminated correctly

(type=bool)

Note:

```
>>> set_led_top(10,0,32)
True
```

set_led_bottom_left(*self, r, g, b*)

Set color for led on bottom left

Parameters

`r`: value of red (0-32)

(type=int)

`g`: value of green (0-32)

(type=int)

`b`: value of blue (0-32)

(type=int)

Return Value

True if the method terminated correctly

(type=bool)

Note:

```
>>> set_led_bottom_left(10,0,32)
True
```

set_led_bottom_right(*self*, *r*, *g*, *b*)

Set color for led on bottom right

Parameters

`r`: value of red (0-32)

(type=int)

`g`: value of green (0-32)

(type=int)

`b`: value of blue (0-32)

(type=int)

Return Value

True if the method terminated correctly

(type=bool)

Note:

```
>>> set_led_bottom_right(10,0,32)
True
```

set_led_temp(*self*, *r*, *b*)

Set color for temperature led on right

Parameters

`r`: value of red (0-32)

(type=int)

`b`: value of blue (0-32)

(type=int)

Return Value

True if the method terminated correctly

(type=bool)

Note:

```
>>> set_led_temp(10,32)
True
```

set_led_sound(*self*, *r*)

Set color for sound led on left

Parameters

`r`: value of red (0-32)

(type=int)

Return Value

True if the method terminated correctly

(type=bool)

Note:

```
>>> set_led_sound(20)
True
```

set_led_button(*self*, *leds*)

Set color for led on top

Parameters

`leds`: value for leds for each arrow button (forward, right, backward, left)

(type=int[4])

Return Value

True if the method terminated correctly

(type=bool)

Note:

```
>>> set_led_button([0, 32, 0, 32])
True
```

set_led_circle(*self*, *leds*)

Set color for led circle on top

Parameters

`leds`: value for leds for each part of circle (clockwise from forward)

(type=int[8])

Return Value

True if the method terminated correctly

(type=bool)

Note:

```
>>> set_led_circle([0, 32, 0, 32, 0, 32, 0, 32])
True
```

set_led_prox_h(*self*, *leds*)

Set color for leds of horizontal proximity sensors

Parameters

`leds`: value for leds for each proximity sensor (0-5: front left to right, 6-7 back left to right)

(type=int[8])

Return Value

True if the method terminated correctly

(type=bool)

Note:

```
>>> set_led_prox_h([0, 0, 32, 32, 0, 0, 16, 16])
True
```

set_led_prox_v(*self*, *leds*)

Set color for leds of vertical proximity sensors

Parameters

`leds`: value for leds for each proximity sensor (0: left, 1: right)

(type=int[2])

Return Value

True if the method terminated correctly

(type=bool)

Note:

```
>>> set_led_prox_v([32, 32])
True
```

reset(*self*)

Set motors to 0 and turn off top and bottom leds

Return Value

True if the method terminated correctly

(type=bool)

Note:

```
>>> reset()
True
```

Index