

mruby-on-ev3rt+tecs_package APIリファレンスマニュアル

安積卓也（大阪大学）
長谷川涼（大阪大学）
山本拓朗（大阪大学）
小南靖雄（TOPPERS個人会員）

最終更新日：2018/ 5/ 7

クラス一覧

- Balancer
- Battery
- Button
- LCD
- LED
- Speaker, • RTOS, • Parameter

クラスメソッドとして
呼び出す



- Motor
- ColorSensor
- GyroSensor
- UltrasonicSensor
- TouchSensor

インスタンス化 (new) が必要
及びポート番号の指定が必要

引数について

- 小数点以下切り捨て

Balancerクラス

- Balancer.control

Balancer.control

- ETロボコン向けバランス制御（4ミリ秒周期）
- 引数
 - forward…前進/後進命令。100(前進最大値)~-100(後進最大値)
 - turn …旋回命令。100(右旋回最大値)~-100(左旋回最大値)
 - gyro …ジャイロセンサ値
 - gyro_offset …ジャイロセンサオフセット値
 - angle_l …左モータエンコーダ値
 - angle_r … 右モータエンコーダ値
 - battery … バッテリ電圧値(mV)
 - backlashhalf … バックラッシュの半分の値(rd)
- 戻り値:[pwm_l, pwm_r]
 - pwm_l: 左モータPWM出力値
 - pwm_r: 右モータPWM出力値

Batteryクラス

- Battery.mA
- Battery.mV

Battery.mA -> Fixnum

- バッテリーの電流を取得する.
- 引数
 - なし
- 戻り値
 - バッテリーの電流 (mA)

Battery.mV -> Fixnum

- バッテリーの電圧を取得する.
- 引数
 - なし
- 戻り値
 - バッテリーの電圧 (mV)

Buttonクラス

- Button[button].pressed?
 - buttonは下記のどれか
 - :left
 - :right
 - :up
 - :down
 - :enter
 - :back ※本体電源OFFで利用

Button[button]pressed? -> bool

- ボタンの押下状態を取得する.
- 引数
 - なし
- 戻り値
 - true…押されている状態
 - false…押されていない状態

LCDクラス

- LCD.font=(fnt)
 - LCD.draw(str, x, y)
 - LCD.fill_rect(x, y, w, h, color)
 - LCD.draw_line(x0, y0, x1, y1)
 - LCD.show_message_box(title, msg)
 - LCD.error_puts (msg)
 - LCD.print (msg)
 - LCD.puts (msg)
-
- コンソールの幅は0~178, 高さは0~128 (範囲外の数値も指定可能だが、コンソールには表示されない) .

LCD.font=(fnt) -> nil

- フォントを設定する.
- 引数
 - fnt …フォント (シンボル)
 - :small
 - :medium
- 戻り値
 - nil

LCD.draw(str, x, y) -> nil

- 指定位置で文字列を描く.
- 引数
 - str ...文字列
 - x ...左上隅の水平方向の位置
(横方向にフォントサイズ * x文字分ずらした位置)
 - y ...左上隅の垂直方向の位置
(縦方向にフォントサイズ * x文字分ずらした位置)
- 戻り値
 - nil

LCD.fill_rect(x, y, w, h, color) -> nil

- 矩形を描いて色を塗る.
- 引数
 - x …左上隅の水平方向の位置
 - y …左上隅の垂直方向の位置
 - w … 矩形の幅
 - h …矩形の高さ
 - color カラー
 - :white
 - :black
- 戻り値
 - nil

LCD.draw_line(x0, y0, x1, y1) -> nil

- 指定座標で線を引く.
- 引数
 - x0 …始点の水平方向の位置
 - y0 …始点の垂直方向の位置
 - x1 …終点の水平方向の位置
 - y1 …終点の垂直方向の位置
- 戻り値
 - nil

LCD.show_message_box(title, msg) -> nil

- メッセージボックスにメッセージを表示する。
※メッセージボックスを表示中なmrubyのプログラムを一時停止し、中央（Enter）ボタンを押して再開する
- 引数
 - title …メッセージボックスのタイトル
 - msg …メッセージ
- 戻り値
 - nil

LCD.error_puts (msg) -> nil

- メッセージボックスにエラーを出力する.
- 引数
 - msg …エラーメッセージ
- 戻り値
 - nil

LCD.print (str) -> nil

- LCDコンソールに文字列を表示する（改行なし）。
- 引数
 - str …文字列
- 戻り値
 - nil

LCD.puts (str) -> nil

- LCDコンソールに文字列を表示する（改行あり）
- 引数
 - str …文字列
- 戻り値
 - nil

LEDクラス

- LED.color=(clr)
- LED.off

LED.color=(clr) -> nil

- LEDライトのカラーを設定する
- 不正の設定値を指定した場合, LEDライトのカラーを変えない.
- 引数
 - clr…LEDカラーの設定値
 - :red
 - :green
 - :orange
 - :off
- 戻り値
 - nil

LED.off -> nil

- LEDをオフにする
- 引数
 - なし
- 戻り値
 - nil

Speakerクラス

- Speaker.volume=(vol)
- Speaker.tone(frequency, duration)

Speaker.volume=(vol) -> nil

- 音量を調整する.
- 引数
 - vol…ボリュームの値. 範囲：0から+100. 0はミュート. +100を超えた値を指定すると, 実際の値は+100になる.
- 戻り値
 - nil

Speaker.tone(frequency, duration) -> nil

- 指定した周波数でトーン出力する. 今再生しているサウンドは停止される.
- 引数
 - frequency…トーンの周波数
 - duration…出力持続時間. 単位: ミリ秒.
- 戻り値
 - nil

frequencyシンボレー覧

:c4	ノートC4の周波数(261.63)
:cs4	ノート#C4の周波数(277.18)
:d4	ノートD4の周波数(293.66)
:ds4	ノートD#4の周波数(311.13)
:e4	ノートE4の周波数(329.63)
:f4	ノートF4の周波数(349.23)
:fs4	ノートF#4の周波数(369.99)
:g4	ノートG4の周波数(392.00)
:gs4	ノートG#4の周波数(415.30)
:a4	ノートA4の周波数(440.00)
:as4	ノートA#4の周波数(466.16)
:b4	ノートB4の周波数(493.88)
:c5	ノートC5の周波数(523.25)
:cs5	ノートC#5の周波数(554.37)
:d5	ノートD5の周波数(587.33)
:ds5	ノートD#5の周波数(622.25)
:e5	ノートE5の周波数(659.25)
:f5	ノートF5の周波数(698.46)
:fs5	ノートF#5の周波数(739.9)
:g5	ノートG5の周波数(783.99)
:gs5	ノートG#5の周波数(830.61)
:a5	ノートA5の周波数(880.00)
:as5	ノートA#5の周波数(932.33)
:b5	ノートB5の周波数(987.77)

:c6	ノートC6の周波数(1046.50)
:cs6	ノートC#6の周波数(1108.73)
:d6	ノートD6の周波数(1174.66)
:ds6	ノートD#6の周波数(1244.51)
:e6	ノートE6の周波数(1318.51)
:f6	ノートF6の周波数(1396.91)
:fs6	ノートF#6の周波数(1479.98)
:g6	ノートG6の周波数(1567.98)
:gs6	ノートG#6の周波数(1661.22)
:a6	ノートA6の周波数(1760.00)
:as6	ノートA#6の周波数(1864.66)
:b6	ノートB6の周波数(1975.53)

RTOSクラス

- RTOS.delay(msec)
- RTOS.usec
- RTOS.msec
- RTOS.sleep

RTOS.delay(msec) -> nil

- 指定された時間遅延する
(指定された時間後に実行が再開される)
- 引数
 - msec…遅延時間(ミリ秒)
- 戻り値
 - nil

RTOS.usec -> Fixnum

- 性能評価用システム時刻の参照
- 引数
 - なし
- 戻り値
 - 性能評価用システム時刻の現在値（マイクロ秒）

RTOS.msec -> Fixnum

- システム時刻の参照
- 引数
 - なし
- 戻り値
 - システムの時刻の現在値（ミリ秒）

RTOS.sleep -> nil

- タスクを起床待ち状態にさせる
- 引数
 - なし
- 戻り値
 - nil

Motorクラス

- initialize(port, type=:large)
- type
- power=(pwr)
- power
- stop(flt=false)
- rotate(deg, spd, blk=false)
- count
- reset_count

Motor

`initialize(port, type=:large) -> nil`

- モータポートを設定する.
- モータポートに接続しているモータのタイプを設定する.
- 引数
 - port…モータポート番号
 - :port_a
 - :port_b
 - :port_c
 - :port_d
 - type…モータタイプ
 - :large
 - :medium
- 戻り値
 - nil

Motor

type -> Symbol

- モータポートのモータタイプを取得する
- 引数
 - なし
- 返り値
 - 指定したモータポートのモータタイプ
 - :large
 - :medium

Motor

`power=(pwm) -> nil`

- モータのスピードを設定する
- 引数
 - pwr…モータのフルパワーのパーセント値.
範囲：-100から+100 (PWM値)
マイナスの値でモータを逆方向に回転させることができる.
範囲外の場合, ± 100 として扱われる.
- 戻り値
 - nil

Motor

power -> Fixnum

- モータのパワーを取得する
- 引数
 - なし
- 戻り値
 - モータのパワー

Motor

`stop(brake=true) -> nil`

- モータを停止する
- 引数
 - flt...ブレーキモードの指定
 - true* (ブレーキモード)
 - false* (フロートモード)
- 戻り値
 - *nil*

Motor

`rotate(deg, spd, blk=false) -> nil`

- モータを指定した角度だけ回転させる
- 引数
 - `deg`…回転角度,
マイナスの値でモータを逆方向に回転させることができる.
 - `spd`…回転速度, モータポートのフルスピードのパーセント値.
範囲: -100から+100. マイナス値で逆回転
範囲外の場合±100として扱われる
 - `blk`…*true* (関数は回転が完了してからリターン),
false (関数は回転操作を待たずにリターン)
- 戻り値
 - `nil`

Motor

count -> Fixnum

- モータの角位置を取得する
- 不正のモータポート番号を指定した場合, 常に0を返す (エラーログが出力される) .
- 引数
 - なし
- 戻り値
 - モータの角位置 (単位は度) , マイナスの値は逆方向に回転されたことを指す

Motor

reset_count -> nil

- モータの角位置をゼロにリセットする.
- モータの角位置センサの値を設定するだけ, モータの実際のパワーと位置に影響を与えない.
- 引数
 - なし
- 戻り値
 - nil

ColorSensorクラス

- initialize(port)
- reflect
- ambient
- rgb
- color
- black?
- blue?
- green?
- yellow?
- red?
- white?
- brown?

ColorSensor

`initialize(port) -> nil`

- センサポートを設定する
- センサポートに接続しているセンサのタイプを設定する。既に設定した場合も新しいセンサタイプを指定できる。
- 引数
 - port センサポート番号
 - :port_1
 - :port_2
 - :port_3
 - :port_4
- 戻り値
 - nil

ColorSensor

reflect -> Fixnum

- カラーセンサで反射光の強さを測定する.
- 引数
 - なし
- 戻り値
 - 反射光の強さ (0~100)

ColorSensor

ambient -> Fixnum

- カラーセンサで環境光の強さを測定する.
- 引数
 - なし
- 戻り値
 - 環境光の強さ (0~100)

ColorSensor

rgb -> Fixnum

- カラーセンサでRGB値を測定する.
- 引数
 - なし
- 戻り値
 - レッド(R)値 , グリーン(G)値, ブルー(B)値

ColorSensor

color -> Symbol

- カラーセンサでカラーを識別する.
- 引数
 - なし
- 戻り値
 - 識別したカラー (:black, :blue, :green, :yellow, :red, :white, :brown) または、識別できなかった (:none)

ColorSensor

black? -> bool

- 黒色か判別する.
- 引数
 - なし
- 戻り値
 - true: 黒色
 - false: その他の色

ColorSensor

blue? -> bool

- 青色か判別する.
- 引数
 - なし
- 戻り値
 - true: 青色
 - false: その他の色

ColorSensor

green? -> bool

- 緑色か判別する.
- 引数
 - なし
- 戻り値
 - true: 緑色
 - false: その他の色

ColorSensor

yellow? -> bool

- 黄色か判別する.
- 引数
 - なし
- 戻り値
 - true: 黄色
 - false: その他の色

ColorSensor

red? -> bool

- 赤色か判別する.
- 引数
 - なし
- 戻り値
 - true: 赤色
 - false: その他の色

ColorSensor

white? -> bool

- 白色か判別する.
- 引数
 - なし
- 戻り値
 - true: 白色
 - false: その他の色

ColorSensor

brown? -> bool

- 茶色か判別する.
- 引数
 - なし
- 戻り値
 - true: 茶色
 - false: その他の色

GyroSensorクラス

- initialize(port)
- rate
- angle
- reset
- calibrate

GyroSensor

initialize(port) -> nil

- ジャイロセンサポートを設定する
- 引数
 - port…センサポート番号
 - :port_1
 - :port_2
 - :port_3
 - :port_4
- 戻り値
 - nil

GyroSensor

rate -> Fixnum

- ジャイロセンサで角速度を測定する
- 引数
 - なし
- 戻り値
 - 角速度（単位は度/秒）

GyroSensor

angle -> Fixnum

- ジャイロセンサで角位置を測定する.
- 引数
 - なし
- 戻り値
 - 角位置 (単位は度)

GyroSensor

reset -> nil

- ジャイロセンサの角位置をゼロにリセットする.
- 引数
 - なし
- 戻り値
 - nil

GyroSensor

calibrate (n=200) -> Float | Symbol

- ジャイロセンサのキャリブレーション
複数回測定した値の平均値
- 引数
 - n …測定回数：デフォルトは200
- 戻り値
 - offset:測定回数の平均値
 - : E_OBJ:測定値の最大・最小の値が5以上の場合

UltrasonicSensorクラス

- initialize(port)
- distance
- listen

UltrasonicSensor

initialize(port) -> nil

- 超音波センサポートを設定する
- 引数
 - port…センサポート番号
 - :port_1
 - :port_2
 - :port_3
 - :port_4
- 戻り値
 - nil

UltrasonicSensor

distance -> Fixnum

- 超音波センサで距離を測定する.
- 引数
 - なし
- 戻り値
 - 距離 (単位はセンチ)

UltrasonicSensor

listen -> bool

- 超音波センサで超音波信号を検出する.
- 引数
 - なし
- 戻り値
 - true…超音波信号を検出した
 - false…超音波信号を検出しなかった

TouchSensorクラス

- initialize(port)
- pressed?

TouchSensor

initialize(port) -> nil

- タッチセンサポートを設定する
- 引数
 - port…ポート番号
 - :port_1
 - :port_2
 - :port_3
 - :port_4
- 戻り値
 - nil

TouchSensor

pressed? -> bool

- タッチセンサの状態を検出する.
- 不正のセンサポート番号を指定した場合, 常に *false* を返す (エラーログが出力される) .
- 引数
 - なし
- 戻り値
 - true…タッチセンサが押されている状態
 - false…タッチセンサが押されていない状態

Parameterクラス

- `Parameter.set(touch_sensor_port, color_sensor_port, gyro_sensor_port, sonar_sensor_port, tail_motor_port, right_motor_port, left_motor_port)`
- `Parameter.touch_sensor_port`
- `Parameter.color_sensor_port`
- `Parameter.gyro_sensor_port`
- `Parameter.sonar_sensor_port`
- `Parameter.tail_motor_port`
- `Parameter.right_motor_port`
- `Parameter.left_motor_port`

```
Parameter.set(touch_sensor_port,  
color_sensor_port, gyro_sensor_port,  
sonar_sensor_port, tail_motor_port,  
right_motor_port, left_motor_port)->nil
```

- 各種センサ、各モータのポート番号を記録する
- 引数
 - touch_sensor_port…タッチセンサポート番号(Symbol)
 - color_sensor_port…カラーセンサポート番号(Symbol)
 - gyro_sensor_port…ジャイロセンサーポート番号(Symbol)
 - Sonar_sensor_port…超音波センサーポート番号(Symbol)
 - tail_motor_port…尻尾モータポート番号(Symbol)
 - right_motor_port…右モータポート番号(Symbol)
 - left_motor_port…左モータポート番号(Symbol)
- 戻り値
 - nil

ポート番号シンボル一覧

:port_1	ポート1
:port_2	ポート2
:port_3	ポート3
:port_4	ポート4
:port_a	ポートA
:port_b	ポートB
:port_c	ポートC
:port_d	ポートD

Parameter

touch_sensor_port->Symbol

- 記録したタッチセンサのポート番号を取得する
- 引数
 - なし
- 戻り値
 - Symbol

Parameter

color_sensor_port->Symbol

- 記録したカラーセンサのポート番号を取得する
- 引数
 - なし
- 戻り値
 - Symbol

Parameter

gyro_sensor_port->Symbol

- 記録したジャイロセンサのポート番号を取得する
- 引数
 - なし
- 戻り値
 - Symbol

Parameter

sonar_sensor_port->Symbol

- 記録した超音波センサのポート番号を取得する
- 引数
 - なし
- 戻り値
 - Symbol

Parameter

tail_motor_port->Symbol

- 記録したテイルモータのポート番号を取得する
- 引数
 - なし
- 戻り値
 - Symbol

Parameter

right_motor_port->Symbol

- 記録した右モータのポート番号を取得する
- 引数
 - なし
- 戻り値
 - Symbol

Parameter

left_motor_port->Symbol

- 記録した左モータのポート番号を取得する
- 引数
 - なし
- 戻り値
 - Symbol

改版履歴

- 2015年6月2日 alpha 1.0.0
 - 安積卓也（大阪大学）
 - 長谷川涼（大阪大学）
- 2015年7月5日 alpha 1.0.2
 - 安積卓也（大阪大学）
 - 長谷川涼（大阪大学）
- 2016年8月30日 beta 1.0.0
 - 安積卓也（大阪大学）
 - 長谷川涼（大阪大学）
- 2017年5月10日 beta 2.0.1
 - 安積卓也（大阪大学）
 - 長谷川涼（大阪大学）
 - 小南靖雄（TOPPERS個人会員）
- 2018年5月7日 beta 2.1.0
 - 安積卓也（大阪大学）
 - 長谷川涼（大阪大学）
 - 山本拓朗（大阪大学）
 - 小南靖雄（TOPPERS個人会員）