

Commands for EV3-API

Initialization:

command	explanation
#include <ev3.h>	including Header
InitEV3();	Initialization of all EV3-Functions
FreeEV3();	Closing of all EV3-Functions

Display:

command	explanation
bool LcdClean();	Erase Display
LcdPrintf(<Color>,<Text>,...);	Working like printf()

Help:

parameter	type	explanation	value
<Color>	char	Color of text	1: black text 0: white text with black background
<Text>	char*	Pointer to text	e.g. "Hello EV3"

Break:

command	explanation
void Wait(<Zeit_ms>);	Break code for a given time

Help:

parameter	type	explanation	value
<Zeit_ms>	unsigned long	time in ms possible to use the given macros or type directly the value	MS_1...10 :1 10...100 :10 50...500 :50 100...900 :100 SEC_1...10 :1 5...20 :5 30 MIN_1

Inputs (Sensoren):

Commands:

command	explanation
int setAllSensorMode (<Mode>, <Mode>, <Mode>, <Mode>);	Allocate of the sensor types of all 4 ports in the correct order (IN_1, IN_2, IN_3, IN4)
int readSensor (<Input>);	Readout of the actual sensor data

Help:

parameter	type	explanation	value
<Input>	int	Input-Ports	IN_1, IN_2, IN_3, IN_4
<Mode>	char	Name und Mode of the connected Sensors	See next table

Sensor type:

Sensor	<Mode>	explanation	return value
No Sensor	NO_SEN	No sensor to the port connected	-1
Touch sensor	TOUCH_PRESS	Return of state (2 states possible)	Not pressed: 0 pressed: 1
Light sensor	COL_REFLECT	Return of the reflected light intensities in %	0 to 100
	COL_AMBIENT	Return of room light intensities in %	0 to 100
	COL_COLOR	Return of color	0: transparent 1: black 2: blue 3: green 4: yellow 5: red 6: white 7: brown
Sonar sensor	US_DIST_MM	Return of distance in mm	0 to 2550
Gyroscope	GYRO_ANG	Return of angle in °	-180 to 180
	GYRO_RATE	Return of gear rate in °/s	-440 to 440

Outputs (Motoren):

Controlling:

command	explanation
void OnFwdReg (<Output>, <Speed>);	Forwards/backwards with given speed
void OnRevReg (<Output>, <Speed>);	
void OnFwdSync (<Output>, <Speed>);	synchronized forwards/backwards with given speed (only working with two motors)
void OnRevSync (<Output>, <Speed>);	
void Off (<Output>);	Switch off motors
void RotateMotor (<Output>, <Speed>, <Angle>);	Rotate with given speed for a defined angle (Code stops till the angle is reached)

Reading out:

command	explanation
int MotorRotationCount (<Output>);	Rotation angle of the motors in °
void ResetRotationCount (<Output>);	Reset of rotation angle
char MotorPower (<Output>);	Actual motor speed

Help:

parameter	type	explanation	value
<Output>	int	Output-ports	OUT_A, OUT_B, OUT_C, OUT_D, OUT_AB, OUT_AC, OUT_AD, OUT_BC, OUT_BD, OUT_CD, OUT_ABC, OUT_BCD, OUT_ABCD, OUT_ALL
<Speed>	char	speed	0 to 100
<Angle>	int	angle in °	

Buttons und LED:

Functions for LED:

command	explanation
void SetLedPattern(<Pattern>);	Changing color of LED behind buttons
void SetLedWarning (<Value>);	Activate/deaactivate of warning. LED color cannot be changed while warning is set.

Functions for buttons:

command	explanation
word ButtonWaitForAnyPress(<Zeit>);	Waiting for button press for given time
bool ButtonIsUp(<Button>);	Check if button is pressed or not
bool ButtonIsDown(<Button>);	(1: true, 0: false)
void ButtonWaitForPress(<Button>);	Waiting till a specific button is pressed
void ButtonWaitForPressAndRelease(<Button>);	Waiting till a specific button is pressed and released

Help:

parameter	type	explanation	value
<Pattern>	byte	Color and modus of LED	LED_BLACK (0) LED_GREEN (1) LED_RED (2) LED_ORANGE (3) LED_GREEN_FLASH (4) LED_RED_FLASH (5) LED_ORANGE_FLASH (6) LED_GREEN_PULSE (7) LED_RED_PULSE (8) LED_ORANGE_PULSE (9)
<Value>	bool	Activate or deactivate	0: deactivate 1: activate
<Zeit>	uint	time in ms	0: for endless waiting
<Button>	byte	Name of button	BTNEXIT, BTNRIGHT, BTNLEFT, BTNCENTER, BTNUP, BTNDOWN