List of commands (public functions) of the AP3216_WE library

Function	Parameters	what it does
void Init();	none	initiates the AP3216 with some register values
void setMode(mode);	AP3216_ALS, AP3216_PS, AP3216_ALS_PS, AP3216_ALS_ONCE, AP3216_PS_ONCE, AP3216_ALS_PS_ONCE, AP3216_POWER_DOWN, AP3216_RESET	Continuous or singel measurements of ALS, PS or both. Or switch off or reset the device.
AP3216IntStatus getIntStatus( );	none	resturns the interrupt status: 0 (NO_INT), 1 (ALS_INT), 2 (PS_INT) or 3 (ALS_PS_INT).
void clearInterrupt(*interrupt Status*);	1 (ALS_INT), 2 (PS_INT) or 3 (ALS_PS_INT).	clears interrupts manually
void setIntClearManner(*mode*);	0 (CLR_INT_BY_DATA_READ), 1 (CLR_INT_MANUALLY)	clear interrupts manually or by reading data registers
uint16_t getIRData( );	none	returns ambient infrared light
bool irDataIsOverflowed( );	none	returns if IR data register is overflowed; if true, PS value might not be valid.
float getAmbientLight( );	none	returns ambient light in lux
uint16_t getProximity( );	none	returns proxmity value
bool objectIsNear( );	none	returns if an object is within PS threshold or beyond upper limit; the upper limit has to be crossed once.
void setLuxRange(*range*);	RANGE_20661 (default), RANGE_5162, RANGE_1291, RANGE_323	sets the lux range - smaller range = higher resolution
void setALSIntAfterNConversions(*number*);	1 (default), 4, 8, 12, 16, 20, ..., 52, 56, 60	only if the ALS thresholds are exceeded n times an interrupt will be triggered
void setALSCalibrationFactor(*factor*);	1.0 (default) 3.98	ALS value will be multiplied with the factor. To be used for calibration, e.g. when the sensor is placed behind a window.
void setALSThresholds(*lower thresh., upper thr.*);	Thresholds in lux	sets lower and upper thresholds for ambient light interrupts. Don't exceed the lux range!
void setPSIntegrationTime(*factor*);	1 (default), 2, 3, 4,......., 15, 16	sets PS integration time; higher values will increase max. distance and accuracy
void setPSGain(*factor*);	1, 2 (default), 4, 8	increases proximity value, slightly higher max. distance, higher noise
void setPSIntAfterNConversions(*number*);	1, 2, 4, 8	only if the PS thresholds are exceeded n times an interrupt will be triggered
void setNumberOfLEDPulses(*number*);	0 (makes no sense), 1 (default), 2, 3	number of LED pulses per proximity measurement; increases slightly max. distance.
void setLEDCurrent(*percentage*);	LED_16_7, LED_33_3, LED_66_7, LED_100 (default)	LED current is 100% by default; can be reduced to 66.7, 33.3, 16.7%
void setPSInterruptMode(*mode*);	0 (INT_MODE_ZONE), 1 (INT_MODE_HYSTERESIS)	see datasheet and examples
void setPSMeanTime(*time*);	0 (PS_MEAN_TIME_12_5), 1 (PS_MEAN_TIME_25), 2 (PS_MEAN_TIME_37_5), 3 (PS_MEAN_TIME_50)	Time for PS measurement; default is 12.5 ms; higher values increase accuracy
byte setLEDWaitingTime(*factor*);	0 (default), 1, 2, 3, 4, ... , 60, 61, 63	sets waiting time between measurements; waiting time = n x PS mean time, or: n x (PS mean time + ALS conversion time) if both active
bool setPSCalibration(*PS value*);	0 (default), ..., 511	PS measurement output will be: measured PS value - calibration value