

Libelium Smart Boards

1.3.1

Generated by Doxygen 1.9.1

1 Main Page	1
1.0.0.1 Welcome!	1
1.0.0.2 Project Description	1
2 Smart Agriculture board	3
2.1 Info	3
3 Smart Events board	5
3.1 Info	5
4 Smart Gas board	7
4.1 Info	7
5 Smart Water board	9
5.1 Info	9
6 File Index	11
6.1 File List	11
7 File Documentation	13
7.1 Codes/lora_join.cpp File Reference	13
7.1.1 Function Documentation	13
7.1.1.1 lora_join()	13
7.2 Codes/lora_join.h File Reference	14
7.2.1 Function Documentation	14
7.2.1.1 lora_join()	14
7.3 Codes/lora_send.cpp File Reference	15
7.3.1 Function Documentation	15
7.3.1.1 lora_send()	15
7.4 Codes/lora_send.h File Reference	16
7.4.1 Function Documentation	16
7.4.1.1 lora_send()	16
7.5 Codes/Smart_Agriculture.cpp File Reference	16
7.5.1 Macro Definition Documentation	18
7.5.1.1 SENDRATEMINUTES	18
7.5.2 Function Documentation	18
7.5.2.1 create_bytes()	18
7.5.2.2 get_measurements()	18
7.5.2.3 loop()	19
7.5.2.4 measureSensors()	19
7.5.2.5 setup()	19
7.5.2.6 wmSensor1()	19
7.5.3 Variable Documentation	19
7.5.3.1 adr_flag	19

7.5.3.2 anemometer	19
7.5.3.3 app_eui	20
7.5.3.4 app_key	20
7.5.3.5 battery_level	20
7.5.3.6 BME_temperature	20
7.5.3.7 bytes	20
7.5.3.8 cbar	20
7.5.3.9 counter	20
7.5.3.10 device_eui	21
7.5.3.11 humidity	21
7.5.3.12 i	21
7.5.3.13 j	21
7.5.3.14 lwSensor	21
7.5.3.15 pending_pulses	21
7.5.3.16 pluviometer1	21
7.5.3.17 pluviometer2	22
7.5.3.18 pluviometer3	22
7.5.3.19 port	22
7.5.3.20 pressure	22
7.5.3.21 pt1000Sensor	22
7.5.3.22 size_of_buffer	22
7.5.3.23 socket	22
7.5.3.24 temperature	23
7.5.3.25 vane	23
7.5.3.26 vane_flag	23
7.5.3.27 vane_str	23
7.5.3.28 vaneSensor	23
7.5.3.29 watermark1	23
7.5.3.30 weather	23
7.5.3.31 wetness	23
7.6 Codes/Smart_Events.cpp File Reference	24
7.6.1 Macro Definition Documentation	25
7.6.1.1 DRY	25
7.6.1.2 VERY_WET	25
7.6.1.3 WET	25
7.6.2 Function Documentation	25
7.6.2.1 liquidLevel()	25
7.6.2.2 liquidPresence()	25
7.6.2.3 loop()	25
7.6.2.4 pir()	26
7.6.2.5 setup()	26
7.6.3 Variable Documentation	26

7.6.3.1 adr_flag	26
7.6.3.2 app_eui	26
7.6.3.3 app_key	26
7.6.3.4 battery_level	26
7.6.3.5 bytes	26
7.6.3.6 data	27
7.6.3.7 device_eui	27
7.6.3.8 i	27
7.6.3.9 liquid_flag	27
7.6.3.10 liquid_level	27
7.6.3.11 pir_value	27
7.6.3.12 port	27
7.6.3.13 resistance	28
7.6.3.14 size_of_buffer	28
7.6.3.15 socket	28
7.6.3.16 value	28
7.6.3.17 voltage	28
7.7 Codes/Smart_Gas.cpp File Reference	28
7.7.1 Macro Definition Documentation	30
7.7.1.1 CALIBRATION_POINTS	30
7.7.1.2 MINUTESTOSEND	30
7.7.1.3 POINT1_PPM_CO	30
7.7.1.4 POINT1_PPM_NO2	31
7.7.1.5 POINT1_RES_CO	31
7.7.1.6 POINT1_RES_NO2	31
7.7.1.7 POINT2_PPM_CO	31
7.7.1.8 POINT2_PPM_NO2	31
7.7.1.9 POINT2_RES_CO	31
7.7.1.10 POINT2_RES_NO2	31
7.7.1.11 POINT3_PPM_CO	32
7.7.1.12 POINT3_PPM_NO2	32
7.7.1.13 POINT3_RES_CO	32
7.7.1.14 POINT3_RES_NO2	32
7.7.2 Function Documentation	32
7.7.2.1 APPSensor()	32
7.7.2.2 APPSensor2()	32
7.7.2.3 loop()	32
7.7.2.4 setup()	33
7.7.3 Variable Documentation	33
7.7.3.1 adr_flag	33
7.7.3.2 app_eui	33
7.7.3.3 app_key	33

7.7.3.4 battery_level	33
7.7.3.5 bytes	34
7.7.3.6 co_particles	34
7.7.3.7 co_resistance	34
7.7.3.8 co_send	34
7.7.3.9 co_voltage	34
7.7.3.10 concentrations_co	34
7.7.3.11 concentrations_no2	34
7.7.3.12 COSensor	35
7.7.3.13 device_eui	35
7.7.3.14 error	35
7.7.3.15 i	35
7.7.3.16 luxes	35
7.7.3.17 luxes_send	35
7.7.3.18 no2_particles	35
7.7.3.19 no2_resistance	35
7.7.3.20 no2_send	36
7.7.3.21 no2_voltage	36
7.7.3.22 NO2Sensor	36
7.7.3.23 polutants2_particles	36
7.7.3.24 polutants2_resistance	36
7.7.3.25 polutants2_send	36
7.7.3.26 polutants2_voltage	36
7.7.3.27 polutants_particles	36
7.7.3.28 polutants_resistance	37
7.7.3.29 polutants_send	37
7.7.3.30 polutants_voltage	37
7.7.3.31 port	37
7.7.3.32 resistances_co	37
7.7.3.33 size_of_buffer	37
7.7.3.34 socket	37
7.7.3.35 voltages_no2	38
7.8 Codes/Smart_Water.cpp File Reference	38
7.8.1 Macro Definition Documentation	39
7.8.1.1 CALIBRATION_POINT_10	39
7.8.1.2 CALIBRATION_POINT_4	39
7.8.1.3 CALIBRATION_POINT_7	39
7.8.1.4 CALIBRATION_TEMPERATURE	39
7.8.2 Function Documentation	39
7.8.2.1 loop()	39
7.8.2.2 setup()	39
7.8.3 Variable Documentation	40

7.8.3.1 adr_flag	40
7.8.3.2 app_eui	40
7.8.3.3 app_key	40
7.8.3.4 battery_level	40
7.8.3.5 bytes	40
7.8.3.6 device_eui	40
7.8.3.7 pH_value	41
7.8.3.8 pH_value_send	41
7.8.3.9 pH_voltage	41
7.8.3.10 pHSensor	41
7.8.3.11 port	41
7.8.3.12 size_of_buffer	41
7.8.3.13 socket	41
7.8.3.14 temperature	42
7.8.3.15 temperature_send	42
7.8.3.16 temperatureSensor	42
7.9 doc_pages/Agriculture.md File Reference	42
7.10 doc_pages/Events.md File Reference	42
7.11 doc_pages/Gas.md File Reference	42
7.12 doc_pages/Main.md File Reference	42
7.13 doc_pages/Water.md File Reference	42
Index	43

Chapter 1

Main Page

1.0.0.1 Welcome!

Here you can find our documentation for the four different codes we have developed in order to monitor our IoT ecosystem.

1.0.0.2 Project Description

This project aims to monitor the condition of the iti's Smart Home wirelessly through the LoRaWAN protocol. For that reason, there has been used four different Smart Boards from Libelium to get data from various sensors and track the condition that exists in the Smart House every some time stamps. Such sensors are for hazardous gases (Gas board), for water quality (Water board), for wheather conditions (Agriculture board) and for various events like water presence (Events board). For sending the data ([lora_send\(\)](#)) and for joining the network ([lora_join\(\)](#)), there have been implemented two libraries and they are imported to each board's code.

For more details for each board open the links below:

For the Agriculture board: [Info](#)

For the Events board: [Info](#)

For the Gas board: [Info](#)

For the Water board: [Info](#)

Author

Ioannis Chatzipaschalis

Chapter 2

Smart Agriculture board

2.1 Info

The Smart Agriculture board is responsible for monitoring the outdoor wheather conditions such as the wetness, the height of rain, the direction of the vane, the temprature and the pressure in order to have the best conditions for growing plants. There are functions that get and print the sensors' measurements, another that prepares the data for sending and two last that are responsible for the connection eshtablishment (`lora_join`) and for the data transmission (`lora_send`), which you can find them in the separate files imported as libraries. The board's IDs such as Device EUl etc have been declared at the top of the code.

Chapter 3

Smart Events board

3.1 Info

The Smart Events board is responsible for monitoring the water presence to avoid a flood in critical rooms like a power room. There are functions that get and print the sensors' measurements, another that prepares the data for sending and two last that are responsible for the connection establishment (`lora_join`) and for the data transmission (`lora_send`), which you can find them in the separate files imported as libraries. There have been voltage definitions at the top of the code for the water presence conditions based on the sensor manufacturer. In that part of the code you can also find the board's IDs such as Device EUI etc that have been declared.

Chapter 4

Smart Gas board

4.1 Info

The Smart Gas board is responsible for monitoring the concentration (in ppm) of hazardous gases in the atmosphere such as CO and NO₂. There are functions that get and print the sensors' measurements, another that prepares the data for sending and two last that are responsible for the connection establishment (`lora_join`) and for the data transmission (`lora_send`), which you can find them in the separate files imported as libraries. You can also see the calibration values as they are recommended from the manufacturer. The board's IDs such as Device EUI etc have been declared at the top of the code.

Chapter 5

Smart Water board

5.1 Info

The Smart Water board is responsible for monitoring the water's temprature and pH. There are functions that get and print the sensors' measurements, another that prepares the data for sending and two last that are responsible for the connection eshtablishment (`lora_join`) and for the data transmission (`lora_send`), which you can find them in the separate files imported as libraries. You can also see the calibration values as they are recommended from the manufacturer. The board's IDs such as Device EUI etc have been declared at the top of the code.

Chapter 6

File Index

6.1 File List

Here is a list of all files with brief descriptions:

Codes/ lora_join.cpp	13
Codes/ lora_join.h	14
Codes/ lora_send.cpp	15
Codes/ lora_send.h	16
Codes/ Smart_Agriculture.cpp	16
Codes/ Smart_Events.cpp	24
Codes/ Smart_Gas.cpp	28
Codes/ Smart_Water.cpp	38

Chapter 7

File Documentation

7.1 Codes/lora_join.cpp File Reference

```
#include <WaspLoRaWAN.h>
#include "Wasmote.h"
#include "lora_join.h"
```

Functions

- void [lora_join](#) (char [device_eui](#)[], char [app_eui](#)[], char [app_key](#)[], uint8_t [socket](#), uint8_t [adr_flag](#))
LoRaWAN Connection Function.

7.1.1 Function Documentation

7.1.1.1 [lora_join\(\)](#)

```
void lora_join (
    char device_eui[],
    char app_eui[],
    char app_key[],
    uint8_t socket,
    uint8_t adr_flag )
```

LoRaWAN Connection Function.

Returns

Nothing

Parameters

<i>device_eui</i>	The EUI of the board.
<i>app_eui</i>	The EUI of the application.
<i>app_key</i>	The key needed for the encryption.
<i>socket</i>	Specifies the socket of the board we are using.
<i>adr_flag</i>	Indicates if we need Adaptive Data Rate or not.

7.2 Codes/lora_join.h File Reference

Functions

- void [lora_join](#) (char [device_eui](#)[], char [app_eui](#)[], char [app_key](#)[], uint8_t [socket](#), uint8_t [adr_flag](#))
LoRaWAN Connection Function.

7.2.1 Function Documentation

7.2.1.1 [lora_join\(\)](#)

```
void lora_join (
    char device_eui[],
    char app_eui[],
    char app_key[],
    uint8_t socket,
    uint8_t adr_flag )
```

LoRaWAN Connection Function.

Returns

Nothing

Parameters

<i>device_eui</i>	The EUI of the board.
<i>app_eui</i>	The EUI of the application.
<i>app_key</i>	The key needed for the encryption.
<i>socket</i>	Specifies the socket of the board we are using.
<i>adr_flag</i>	Indicates if we need Adaptive Data Rate or not.

7.3 Codes/lora_send.cpp File Reference

```
#include <WaspLoRaWAN.h>
#include <WaspUSB.h>
#include <WaspUSB.cpp>
#include "Waspmote.h"
#include "lora_send.h"
```

Functions

- void `lora_send` (char `device_eui`[], char `app_eui`[], char `app_key`[], uint8_t `size_of_buffer`, uint8_t `port`, uint8_t `socket`, uint8_t `bytes`[])
Data Transmission Function.

7.3.1 Function Documentation

7.3.1.1 `lora_send()`

```
void lora_send (
    char device_eui[],
    char app_eui[],
    char app_key[],
    uint8_t size_of_buffer,
    uint8_t port,
    uint8_t socket,
    uint8_t bytes[] )
```

Data Transmission Function.

Returns

Nothing

Parameters

<code>device_eui</code>	The EUI of the board.
<code>app_eui</code>	The EUI of the application.
<code>app_key</code>	The key needed for the encryption.
<code>size_of_buffer</code>	The size of our data vector.
<code>port</code>	Specifies the port we are using.
<code>socket</code>	Specifies the socket of the board we are using.
<code>bytes</code>	The data vector to be sent.

7.4 Codes/lora_send.h File Reference

Functions

- void `lora_send` (char `device_eui`[], char `app_eui`[], char `app_key`[], uint8_t `size_of_buffer`, uint8_t `port`, uint8_t `socket`, uint8_t `bytes`[])↔

Data Transmission Function.

7.4.1 Function Documentation

7.4.1.1 `lora_send()`

```
void lora_send (
    char device_eui[],
    char app_eui[],
    char app_key[],
    uint8_t size_of_buffer,
    uint8_t port,
    uint8_t socket,
    uint8_t bytes[] )
```

Data Transmission Function.

Returns

Nothing

Parameters

<code>device_eui</code>	The EUI of the board.
<code>app_eui</code>	The EUI of the application.
<code>app_key</code>	The key needed for the encryption.
<code>size_of_buffer</code>	The size of our data vector.
<code>port</code>	Specifies the port we are using.
<code>socket</code>	Specifies the socket of the board we are using.
<code>bytes</code>	The data vector to be sent.

7.5 Codes/Smart_Agriculture.cpp File Reference

```
#include <WaspLoRaWAN.h>
#include <WaspSensorAgr_v30.h>
#include <lora_send.h>
#include <lora_join.h>
```


Macros

- #define `SENDERATEMINUTES` 30
Time in minutes.

Functions

- watermarkClass `wmSensor1` (SOCKET_2)
- void `setup` ()
- void `loop` ()
- void `create_bytes` ()
Function for creating the ready to send bytes vector.
- void `get_measurements` ()
Function for getting the measurements from the Sensors.
- void `measureSensors` ()
Function for Reading the wind sensor values.

Variables

- uint8_t `socket` = SOCKET0
Socket Selection for the Sensor Probe.
- char `device_eui` [] = "72c465dc9a6d5390"
Device parameters for Back-End registration.
- char `app_eui` [] = "72c465dc9a6d5390"
- char `app_key` [] = "1503835a0767d925cd3c8f9f2bcbb6b5"
- uint8_t `port` = 10
Define port to use in Back-End: from 1 to 223.
- uint8_t `bytes` [45] = {0}
Define data payload to send (maximum is up to data rate)
- char `vane_str` [10] = {0}
- uint8_t `j` = 0
Variables.
- uint8_t `i` = 0
- int `battery_level` = 0
- float `wetness` = 0
- float `watermark1`
- float `temperature` = 0
- float `anemometer`
- float `pluviometer1`
mm in current hour
- float `pluviometer2`
mm in previous hour
- float `pluviometer3`
mm in last 24 hours
- uint8_t `cbar` = 2
- int `vane`
- float `BME_temperature`
- float `humidity`
- float `pressure`
- int `pending_pulses`
- uint8_t `counter` = 0

- uint8_t [vane_flag](#) = 0
- uint8_t [size_of_buffer](#) = 0
- uint8_t [adr_flag](#) = 1

Adaptive Data Rate indicator for [lora_join\(\)](#)

- leafWetnessClass [lwSensor](#)
- weatherStationClass [vaneSensor](#)
- weatherStationClass [weather](#)
- pt1000Class [pt1000Sensor](#)

7.5.1 Macro Definition Documentation

7.5.1.1 SENDRATEMINUTES

```
#define SENDRATEMINUTES 30
```

Time in minutes.

7.5.2 Function Documentation

7.5.2.1 create_bytes()

```
void create_bytes ( )
```

Function for creating the ready to send bytes vector.

Returns

Nothing

Decoding values to base64 for send preparation

7.5.2.2 get_measurements()

```
void get_measurements ( )
```

Function for getting the measurements from the Sensors.

Returns

Nothing

7.5.2.3 loop()

```
void loop ( )
```

7.5.2.4 measureSensors()

```
void measureSensors ( )
```

Function for Reading the wind sensor values.

Returns

Nothing

7.5.2.5 setup()

```
void setup ( )
```

7.5.2.6 wmSensor1()

```
watermarkClass wmSensor1 (
    SOCKET_2 )
```

7.5.3 Variable Documentation

7.5.3.1 adr_flag

```
uint8_t adr_flag = 1
```

Adaptive Data Rate indicator for [lora_join\(\)](#)

7.5.3.2 anemometer

```
float anemometer
```

7.5.3.3 app_eui

```
char app_eui[] = "72c465dc9a6d5390"
```

7.5.3.4 app_key

```
char app_key[] = "1503835a0767d925cd3c8f9f2bcbb6b5"
```

7.5.3.5 battery_level

```
int battery_level = 0
```

7.5.3.6 BME_temperature

```
float BME_temperature
```

7.5.3.7 bytes

```
uint8_t bytes[45] = {0}
```

Define data payload to send (maximum is up to data rate)

7.5.3.8 cbar

```
uint8_t cbar = 2
```

7.5.3.9 counter

```
uint8_t counter = 0
```

7.5.3.10 device_eui

```
char device_eui[] = "72c465dc9a6d5390"
```

Device parameters for Back-End registration.

7.5.3.11 humidity

```
float humidity
```

7.5.3.12 i

```
uint8_t i = 0
```

7.5.3.13 j

```
uint8_t j = 0
```

Variables.

7.5.3.14 lwSensor

```
leafWetnessClass lwSensor
```

7.5.3.15 pending_pulses

```
int pending_pulses
```

7.5.3.16 pluviometer1

```
float pluviometer1
```

mm in current hour

7.5.3.17 pluviometer2

```
float pluviometer2
```

mm in previous hour

7.5.3.18 pluviometer3

```
float pluviometer3
```

mm in last 24 hours

7.5.3.19 port

```
uint8_t port = 10
```

Define port to use in Back-End: from 1 to 223.

7.5.3.20 pressure

```
float pressure
```

7.5.3.21 pt1000Sensor

```
pt1000Class pt1000Sensor
```

7.5.3.22 size_of_buffer

```
uint8_t size_of_buffer = 0
```

7.5.3.23 socket

```
uint8_t socket = SOCKET0
```

Socket Selection for the Sensor Probe.

7.5.3.24 temperature

```
float temperature = 0
```

7.5.3.25 vane

```
int vane
```

7.5.3.26 vane_flag

```
uint8_t vane_flag = 0
```

7.5.3.27 vane_str

```
char vane_str[10] = {0}
```

7.5.3.28 vaneSensor

```
weatherStationClass vaneSensor
```

7.5.3.29 watermark1

```
float watermark1
```

7.5.3.30 weather

```
weatherStationClass weather
```

7.5.3.31 wetness

```
float wetness = 0
```

7.6 Codes/Smart_Events.cpp File Reference

```
#include <WaspSensorEvent_v30.h>
#include <WaspLoRaWAN.h>
#include <lora_send.h>
#include <lora_join.h>
```

Macros

- #define `DRY` 0.0
Voltage Values.
- #define `WET` 0.2
- #define `VERY_WET` 1.0

Functions

- liquidPresenceClass `liquidPresence` (SOCKET_6)
- pirSensorClass `pir` (SOCKET_1)
- liquidLevelClass `liquidLevel` (SOCKET_4)
- void `setup` ()
- void `loop` ()

Variables

- uint8_t `socket` = SOCKET0
Socket Selection for the Sensor Probe.
- char `device_eui` [] = "72c465dc2f1bcfa7"
Device parameters for Back-End registration.
- char `app_eui` [] = "72c465dc2f1bcfa7"
- char `app_key` [] = "b5351d67909803c983b66484d25f9362"
- uint8_t `port` = 10
Define port to use in Back-End: from 1 to 223.
- uint8_t `bytes` [12] = {0}
Define data payload to send (maximum is up to data rate)
- char `data` [] = "0102030405060708090A0B0C0D0E0F"
Setup the data vector.
- int `i`
Variables.
- int `battery_level` = 0
- uint8_t `value`
- float `voltage`
- float `resistance`
- uint8_t `pir_value` = 0
- uint8_t `liquid_level` = 0
- uint8_t `liquid_flag` = 0
- uint8_t `size_of_buffer` = sizeof(`bytes`) / sizeof(`bytes`[0])
- uint8_t `adr_flag` = 1
Adaptive Data Rate indicator for `lora_join()`

7.6.1 Macro Definition Documentation

7.6.1.1 DRY

```
#define DRY 0.0
```

Voltage Values.

7.6.1.2 VERY_WET

```
#define VERY_WET 1.0
```

7.6.1.3 WET

```
#define WET 0.2
```

7.6.2 Function Documentation

7.6.2.1 liquidLevel()

```
liquidLevelClass liquidLevel (
    SOCKET_4 )
```

7.6.2.2 liquidPresence()

```
liquidPresenceClass liquidPresence (
    SOCKET_6 )
```

7.6.2.3 loop()

```
void loop ( )
```

7.6.2.4 pir()

```
pirSensorClass pir (
    SOCKET_1 )
```

7.6.2.5 setup()

```
void setup ( )
```

7.6.3 Variable Documentation

7.6.3.1 adr_flag

```
uint8_t adr_flag = 1
```

Adaptive Data Rate indicator for [lora_join\(\)](#)

7.6.3.2 app_eui

```
char app_eui[] = "72c465dc2f1bcfa7"
```

7.6.3.3 app_key

```
char app_key[] = "b5351d67909803c983b66484d25f9362"
```

7.6.3.4 battery_level

```
int battery_level = 0
```

7.6.3.5 bytes

```
uint8_t bytes[12] = {0}
```

Define data payload to send (maximum is up to data rate)

7.6.3.6 data

```
char data[] = "0102030405060708090A0B0C0D0E0F"
```

Setup the data vector.

7.6.3.7 device_eui

```
char device_eui[] = "72c465dc2f1bcfa7"
```

Device parameters for Back-End registration.

7.6.3.8 i

```
int i
```

Variables.

7.6.3.9 liquid_flag

```
uint8_t liquid_flag = 0
```

7.6.3.10 liquid_level

```
uint8_t liquid_level = 0
```

7.6.3.11 pir_value

```
uint8_t pir_value = 0
```

7.6.3.12 port

```
uint8_t port = 10
```

Define port to use in Back-End: from 1 to 223.

7.6.3.13 resistance

```
float resistance
```

7.6.3.14 size_of_buffer

```
uint8_t size_of_buffer = sizeof(bytes) / sizeof(bytes[0])
```

7.6.3.15 socket

```
uint8_t socket = SOCKET0
```

Socket Selection for the Sensor Probe.

7.6.3.16 value

```
uint8_t value
```

7.6.3.17 voltage

```
float voltage
```

7.7 Codes/Smart_Gas.cpp File Reference

```
#include <WaspLoRaWAN.h>
#include <WaspSensorGas_v30.h>
#include <lora_send.h>
#include <lora_join.h>
```

Macros

- #define `MINUTESTOSEND` 28
Time in minutes.
- #define `CALIBRATION_POINTS` 3
3 Calibration points
- #define `POINT1_PPM_CO` 100.0
Ro value at this concentration.
- #define `POINT2_PPM_CO` 300.0
- #define `POINT3_PPM_CO` 1000.0
- #define `POINT1_RES_CO` 20.3
Ro Resistance at 100 ppm.
- #define `POINT2_RES_CO` 1.00
- #define `POINT3_RES_CO` 0.25
- #define `POINT1_PPM_NO2` 10.0
Normal concentration in the air.
- #define `POINT2_PPM_NO2` 50.0
- #define `POINT3_PPM_NO2` 100.0
- #define `POINT1_RES_NO2` 2.00
Rs at normal concentration in the air.
- #define `POINT2_RES_NO2` 0.001
- #define `POINT3_RES_NO2` 0.0002

Functions

- APSensorClass `APPSensor` (SOCKET_6)
- APSensorClass `APPSensor2` (SOCKET_7)
- void `setup` ()
- void `loop` ()

Variables

- uint8_t `socket` = SOCKET0
Device parameters for Back-End registration.
- char `device_eui` [] = "72c465dcdb8b2afd"
- char `app_eui` [] = "72c465dcdb8b2afd"
- char `app_key` [] = "c1e90d187fb93834e60ba43c26fb9282"
- uint8_t `port` = 10
Define port to use in Back-End: from 1 to 223.
- char `bytes` [33] = {0}
Define data payload to send (maximum is up to data rate)
- int `battery_level` = 0
General Variables.
- float `luxes`
- uint8_t `error`
- uint8_t `i` = 0
- int `luxes_send` = 0
- uint8_t `size_of_buffer` = sizeof(`bytes`) / sizeof(`bytes`[0])
- uint8_t `adr_flag` = 0
Adaptive Data Rate indicator for `lora_join()`
- float `co_voltage`

Sensor Variables.

- float [co_resistance](#)
- float [co_particles](#)
- int [co_send](#) = 0
- float [no2_voltage](#)
- float [no2_resistance](#)
- float [no2_particles](#)
- int [no2_send](#) = 0
- float [polutants_voltage](#)
- float [polutants_resistance](#)
- float [polutants_particles](#)
- int [polutants_send](#) = 0
- float [polutants2_voltage](#)
- float [polutants2_resistance](#)
- float [polutants2_particles](#)
- int [polutants2_send](#) = 0
- COSensorClass [COSensor](#)
- NO2SensorClass [NO2Sensor](#)
- float [concentrations_co](#) [] = {[POINT1_PPM_CO](#), [POINT2_PPM_CO](#), [POINT3_PPM_CO](#)}
- float [resistances_co](#) [] = {[POINT1_RES_CO](#), [POINT2_RES_CO](#), [POINT3_RES_CO](#)}
- float [concentrations_no2](#) [] = {[POINT1_PPM_NO2](#), [POINT2_PPM_NO2](#), [POINT3_PPM_NO2](#)}
- float [voltages_no2](#) [] = {[POINT1_RES_NO2](#), [POINT2_RES_NO2](#), [POINT3_RES_NO2](#)}

7.7.1 Macro Definition Documentation

7.7.1.1 CALIBRATION_POINTS

```
#define CALIBRATION_POINTS 3
```

3 Calibration points

7.7.1.2 MINUTESTOSEND

```
#define MINUTESTOSEND 28
```

Time in minutes.

7.7.1.3 POINT1_PPM_CO

```
#define POINT1_PPM_CO 100.0
```

Ro value at this concentration.

7.7.1.4 POINT1_PPM_NO2

```
#define POINT1_PPM_NO2 10.0
```

Normal concentration in the air.

7.7.1.5 POINT1_RES_CO

```
#define POINT1_RES_CO 20.3
```

Ro Resistance at 100 ppm.

Necessary value.

7.7.1.6 POINT1_RES_NO2

```
#define POINT1_RES_NO2 2.00
```

Rs at normal concentration in the air.

7.7.1.7 POINT2_PPM_CO

```
#define POINT2_PPM_CO 300.0
```

7.7.1.8 POINT2_PPM_NO2

```
#define POINT2_PPM_NO2 50.0
```

7.7.1.9 POINT2_RES_CO

```
#define POINT2_RES_CO 1.00
```

7.7.1.10 POINT2_RES_NO2

```
#define POINT2_RES_NO2 0.001
```

7.7.1.11 POINT3_PPM_CO

```
#define POINT3_PPM_CO 1000.0
```

7.7.1.12 POINT3_PPM_NO2

```
#define POINT3_PPM_NO2 100.0
```

7.7.1.13 POINT3_RES_CO

```
#define POINT3_RES_CO 0.25
```

7.7.1.14 POINT3_RES_NO2

```
#define POINT3_RES_NO2 0.0002
```

7.7.2 Function Documentation

7.7.2.1 APPSensor()

```
APSensorClass APPSensor (
    SOCKET_6 )
```

7.7.2.2 APPSensor2()

```
APSensorClass APPSensor2 (
    SOCKET_7 )
```

7.7.2.3 loop()

```
void loop ( )
```


7.7.2.4 setup()

```
void setup ( )
```

< Ro value at this concentration

< Ro Resistance at 100 ppm. Necessary value.

< Ro value at this concentration

< Ro Resistance at 100 ppm. Necessary value.

7.7.3 Variable Documentation

7.7.3.1 adr_flag

```
uint8_t adr_flag = 0
```

Adaptive Data Rate indicator for [lora_join\(\)](#)

7.7.3.2 app_eui

```
char app_eui[] = "72c465dcdb8b2afd"
```

7.7.3.3 app_key

```
char app_key[] = "c1e90d187fb93834e60ba43c26fb9282"
```

7.7.3.4 battery_level

```
int battery_level = 0
```

General Variables.

7.7.3.5 bytes

```
char bytes[33] = {0}
```

Define data payload to send (maximum is up to data rate)

7.7.3.6 co_particles

```
float co_particles
```

7.7.3.7 co_resistance

```
float co_resistance
```

7.7.3.8 co_send

```
int co_send = 0
```

7.7.3.9 co_voltage

```
float co_voltage
```

Sensor Variables.

7.7.3.10 concentrations_co

```
float concentrations_co[] = {POINT1_PPM_CO, POINT2_PPM_CO, POINT3_PPM_CO}
```

7.7.3.11 concentrations_no2

```
float concentrations_no2[] = {POINT1_PPM_NO2, POINT2_PPM_NO2, POINT3_PPM_NO2}
```

7.7.3.12 COSensor

```
COSensorClass COSensor
```

7.7.3.13 device_eui

```
char device_eui[] = "72c465dcdb8b2afd"
```

7.7.3.14 error

```
uint8_t error
```

7.7.3.15 i

```
uint8_t i = 0
```

7.7.3.16 luxes

```
float luxes
```

7.7.3.17 luxes_send

```
int luxes_send = 0
```

7.7.3.18 no2_particles

```
float no2_particles
```

7.7.3.19 no2_resistance

```
float no2_resistance
```

7.7.3.20 no2_send

```
int no2_send = 0
```

7.7.3.21 no2_voltage

```
float no2_voltage
```

7.7.3.22 NO2Sensor

```
NO2SensorClass NO2Sensor
```

7.7.3.23 polutants2_particles

```
float polutants2_particles
```

7.7.3.24 polutants2_resistance

```
float polutants2_resistance
```

7.7.3.25 polutants2_send

```
int polutants2_send = 0
```

7.7.3.26 polutants2_voltage

```
float polutants2_voltage
```

7.7.3.27 polutants_particles

```
float polutants_particles
```

7.7.3.28 polutants_resistance

```
float polutants_resistance
```

7.7.3.29 polutants_send

```
int polutants_send = 0
```

7.7.3.30 polutants_voltage

```
float polutants_voltage
```

7.7.3.31 port

```
uint8_t port = 10
```

Define port to use in Back-End: from 1 to 223.

7.7.3.32 resistances_co

```
float resistances_co[] = {POINT1_RES_CO, POINT2_RES_CO, POINT3_RES_CO}
```

7.7.3.33 size_of_buffer

```
uint8_t size_of_buffer = sizeof(bytes) / sizeof(bytes[0])
```

7.7.3.34 socket

```
uint8_t socket = SOCKET0
```

Device parameters for Back-End registration.

7.7.3.35 voltages_no2

```
float voltages_no2[] = {POINT1_RES_NO2, POINT2_RES_NO2, POINT3_RES_NO2}
```

7.8 Codes/Smart_Water.cpp File Reference

```
#include <WaspLoRaWAN.h>
#include <WaspSensorSW.h>
#include <lora_send.h>
#include <lora_join.h>
```

Macros

- #define `CALIBRATION_POINT_10` 1.985
pH - 10, 1.985 Volts
- #define `CALIBRATION_POINT_7` 2.070
pH - 7, 2.070 Volts
- #define `CALIBRATION_POINT_4` 2.227
pH - 4, 2.227 Volts
- #define `CALIBRATION_TEMPERATURE` 23.7

Functions

- void `setup` ()
- void `loop` ()

Variables

- uint8_t `socket` = SOCKET0
Socket Selection for the Sensor Probe.
- char `device_eui` [] = "72c465dcf469a174"
Device parameters for Back-End registration.
- char `app_eui` [] = "72c465dcf469a174"
- char `app_key` [] = "66c203b5c8149813ef44cb88c5bd88be"
- uint8_t `port` = 10
Define port to use in Back-End: from 1 to 223.
- uint8_t `bytes` [11] = {0}
Define data payload to send (maximum is up to data rate)
- int `battery_level` = 0
Variables.
- float `pH_voltage`
- float `temperature`
- float `pH_value`
- int16_t `temperature_send`
- int `pH_value_send`
- uint8_t `size_of_buffer` = sizeof(`bytes`) / sizeof(`bytes`[0])
- uint8_t `adr_flag` = 0
Adaptive Data Rate indicator for `lora_join()`
- pHClass `pHSensor`
- pt1000Class `temperatureSensor`

7.8.1 Macro Definition Documentation

7.8.1.1 CALIBRATION_POINT_10

```
#define CALIBRATION_POINT_10 1.985
```

pH - 10, 1.985 Volts

7.8.1.2 CALIBRATION_POINT_4

```
#define CALIBRATION_POINT_4 2.227
```

pH - 4, 2.227 Volts

7.8.1.3 CALIBRATION_POINT_7

```
#define CALIBRATION_POINT_7 2.070
```

pH - 7, 2.070 Volts

7.8.1.4 CALIBRATION_TEMPERATURE

```
#define CALIBRATION_TEMPERATURE 23.7
```

7.8.2 Function Documentation

7.8.2.1 loop()

```
void loop ( )
```

7.8.2.2 setup()

```
void setup ( )
```

7.8.3 Variable Documentation

7.8.3.1 adr_flag

```
uint8_t adr_flag = 0
```

Adaptive Data Rate indicator for [lora_join\(\)](#)

7.8.3.2 app_eui

```
char app_eui[] = "72c465dcf469a174"
```

7.8.3.3 app_key

```
char app_key[] = "66c203b5c8149813ef44cb88c5bd88be"
```

7.8.3.4 battery_level

```
int battery_level = 0
```

Variables.

7.8.3.5 bytes

```
uint8_t bytes[11] = {0}
```

Define data payload to send (maximum is up to data rate)

7.8.3.6 device_eui

```
char device_eui[] = "72c465dcf469a174"
```

Device parameters for Back-End registration.

7.8.3.7 pH_value

```
float pH_value
```

7.8.3.8 pH_value_send

```
int pH_value_send
```

7.8.3.9 pH_voltage

```
float pH_voltage
```

7.8.3.10 pHSensor

```
pHClass pHSensor
```

7.8.3.11 port

```
uint8_t port = 10
```

Define port to use in Back-End: from 1 to 223.

7.8.3.12 size_of_buffer

```
uint8_t size_of_buffer = sizeof(bytes) / sizeof(bytes[0])
```

7.8.3.13 socket

```
uint8_t socket = SOCKET0
```

Socket Selection for the Sensor Probe.

7.8.3.14 `temperature`

```
float temperature
```

7.8.3.15 `temperature_send`

```
int16_t temperature_send
```

7.8.3.16 `temperatureSensor`

```
pt1000Class temperatureSensor
```

7.9 `doc_pages/Agriculture.md` File Reference

7.10 `doc_pages/Events.md` File Reference

7.11 `doc_pages/Gas.md` File Reference

7.12 `doc_pages/Main.md` File Reference

7.13 `doc_pages/Water.md` File Reference

Index

- adr_flag
 - Smart_Agriculture.cpp, 19
 - Smart_Events.cpp, 26
 - Smart_Gas.cpp, 33
 - Smart_Water.cpp, 40
- anemometer
 - Smart_Agriculture.cpp, 19
- app_eui
 - Smart_Agriculture.cpp, 19
 - Smart_Events.cpp, 26
 - Smart_Gas.cpp, 33
 - Smart_Water.cpp, 40
- app_key
 - Smart_Agriculture.cpp, 20
 - Smart_Events.cpp, 26
 - Smart_Gas.cpp, 33
 - Smart_Water.cpp, 40
- APPSensor
 - Smart_Gas.cpp, 32
- APPSensor2
 - Smart_Gas.cpp, 32
- battery_level
 - Smart_Agriculture.cpp, 20
 - Smart_Events.cpp, 26
 - Smart_Gas.cpp, 33
 - Smart_Water.cpp, 40
- BME_temperature
 - Smart_Agriculture.cpp, 20
- bytes
 - Smart_Agriculture.cpp, 20
 - Smart_Events.cpp, 26
 - Smart_Gas.cpp, 33
 - Smart_Water.cpp, 40
- CALIBRATION_POINT_10
 - Smart_Water.cpp, 39
- CALIBRATION_POINT_4
 - Smart_Water.cpp, 39
- CALIBRATION_POINT_7
 - Smart_Water.cpp, 39
- CALIBRATION_POINTS
 - Smart_Gas.cpp, 30
- CALIBRATION_TEMPERATURE
 - Smart_Water.cpp, 39
- cbar
 - Smart_Agriculture.cpp, 20
- co_particles
 - Smart_Gas.cpp, 34
- co_resistance
 - Smart_Gas.cpp, 34
- co_send
 - Smart_Gas.cpp, 34
- co_voltage
 - Smart_Gas.cpp, 34
- Codes/lora_join.cpp, 13
- Codes/lora_join.h, 14
- Codes/lora_send.cpp, 15
- Codes/lora_send.h, 16
- Codes/Smart_Agriculture.cpp, 16
- Codes/Smart_Events.cpp, 24
- Codes/Smart_Gas.cpp, 28
- Codes/Smart_Water.cpp, 38
- concentrations_co
 - Smart_Gas.cpp, 34
- concentrations_no2
 - Smart_Gas.cpp, 34
- COSensor
 - Smart_Gas.cpp, 34
- counter
 - Smart_Agriculture.cpp, 20
- create_bytes
 - Smart_Agriculture.cpp, 18
- data
 - Smart_Events.cpp, 26
- device_eui
 - Smart_Agriculture.cpp, 20
 - Smart_Events.cpp, 27
 - Smart_Gas.cpp, 35
 - Smart_Water.cpp, 40
- doc_pages/Agriculture.md, 42
- doc_pages/Events.md, 42
- doc_pages/Gas.md, 42
- doc_pages/Main.md, 42
- doc_pages/Water.md, 42
- DRY
 - Smart_Events.cpp, 25
- error
 - Smart_Gas.cpp, 35
- get_measurements
 - Smart_Agriculture.cpp, 18
- humidity
 - Smart_Agriculture.cpp, 21
- i
 - Smart_Agriculture.cpp, 21
 - Smart_Events.cpp, 27

- Smart_Gas.cpp, 35
- j
 - Smart_Agriculture.cpp, 21
- liquid_flag
 - Smart_Events.cpp, 27
- liquid_level
 - Smart_Events.cpp, 27
- liquidLevel
 - Smart_Events.cpp, 25
- liquidPresence
 - Smart_Events.cpp, 25
- loop
 - Smart_Agriculture.cpp, 18
 - Smart_Events.cpp, 25
 - Smart_Gas.cpp, 32
 - Smart_Water.cpp, 39
- lora_join
 - lora_join.cpp, 13
 - lora_join.h, 14
- lora_join.cpp
 - lora_join, 13
- lora_join.h
 - lora_join, 14
- lora_send
 - lora_send.cpp, 15
 - lora_send.h, 16
- lora_send.cpp
 - lora_send, 15
- lora_send.h
 - lora_send, 16
- luxes
 - Smart_Gas.cpp, 35
- luxes_send
 - Smart_Gas.cpp, 35
- lwSensor
 - Smart_Agriculture.cpp, 21
- measureSensors
 - Smart_Agriculture.cpp, 19
- MINUTESTOSEND
 - Smart_Gas.cpp, 30
- no2_particles
 - Smart_Gas.cpp, 35
- no2_resistance
 - Smart_Gas.cpp, 35
- no2_send
 - Smart_Gas.cpp, 35
- no2_voltage
 - Smart_Gas.cpp, 36
- NO2Sensor
 - Smart_Gas.cpp, 36
- pending_pulses
 - Smart_Agriculture.cpp, 21
- pH_value
 - Smart_Water.cpp, 40
- pH_value_send
 - Smart_Water.cpp, 41
- pH_voltage
 - Smart_Water.cpp, 41
- pHSensor
 - Smart_Water.cpp, 41
- pir
 - Smart_Events.cpp, 25
- pir_value
 - Smart_Events.cpp, 27
- pluviometer1
 - Smart_Agriculture.cpp, 21
- pluviometer2
 - Smart_Agriculture.cpp, 21
- pluviometer3
 - Smart_Agriculture.cpp, 22
- POINT1_PPM_CO
 - Smart_Gas.cpp, 30
- POINT1_PPM_NO2
 - Smart_Gas.cpp, 30
- POINT1_RES_CO
 - Smart_Gas.cpp, 31
- POINT1_RES_NO2
 - Smart_Gas.cpp, 31
- POINT2_PPM_CO
 - Smart_Gas.cpp, 31
- POINT2_PPM_NO2
 - Smart_Gas.cpp, 31
- POINT2_RES_CO
 - Smart_Gas.cpp, 31
- POINT2_RES_NO2
 - Smart_Gas.cpp, 31
- POINT3_PPM_CO
 - Smart_Gas.cpp, 31
- POINT3_PPM_NO2
 - Smart_Gas.cpp, 32
- POINT3_RES_CO
 - Smart_Gas.cpp, 32
- POINT3_RES_NO2
 - Smart_Gas.cpp, 32
- polutants2_particles
 - Smart_Gas.cpp, 36
- polutants2_resistance
 - Smart_Gas.cpp, 36
- polutants2_send
 - Smart_Gas.cpp, 36
- polutants2_voltage
 - Smart_Gas.cpp, 36
- polutants_particles
 - Smart_Gas.cpp, 36
- polutants_resistance
 - Smart_Gas.cpp, 36
- polutants_send
 - Smart_Gas.cpp, 37
- polutants_voltage
 - Smart_Gas.cpp, 37
- port
 - Smart_Agriculture.cpp, 22

- Smart_Events.cpp, 27
- Smart_Gas.cpp, 37
- Smart_Water.cpp, 41
- pressure
 - Smart_Agriculture.cpp, 22
- pt1000Sensor
 - Smart_Agriculture.cpp, 22
- resistance
 - Smart_Events.cpp, 27
- resistances_co
 - Smart_Gas.cpp, 37
- SENDRATEMINUTES
 - Smart_Agriculture.cpp, 18
- setup
 - Smart_Agriculture.cpp, 19
 - Smart_Events.cpp, 26
 - Smart_Gas.cpp, 32
 - Smart_Water.cpp, 39
- size_of_buffer
 - Smart_Agriculture.cpp, 22
 - Smart_Events.cpp, 28
 - Smart_Gas.cpp, 37
 - Smart_Water.cpp, 41
- Smart_Agriculture.cpp
 - adr_flag, 19
 - anemometer, 19
 - app_eui, 19
 - app_key, 20
 - battery_level, 20
 - BME_temperature, 20
 - bytes, 20
 - cbar, 20
 - counter, 20
 - create_bytes, 18
 - device_eui, 20
 - get_measurements, 18
 - humidity, 21
 - i, 21
 - j, 21
 - loop, 18
 - lwSensor, 21
 - measureSensors, 19
 - pending_pulses, 21
 - pluviometer1, 21
 - pluviometer2, 21
 - pluviometer3, 22
 - port, 22
 - pressure, 22
 - pt1000Sensor, 22
 - SENDRATEMINUTES, 18
 - setup, 19
 - size_of_buffer, 22
 - socket, 22
 - temperature, 22
 - vane, 23
 - vane_flag, 23
 - vane_str, 23
 - vaneSensor, 23
 - watermark1, 23
 - weather, 23
 - wetness, 23
 - wmSensor1, 19
- Smart_Events.cpp
 - adr_flag, 26
 - app_eui, 26
 - app_key, 26
 - battery_level, 26
 - bytes, 26
 - data, 26
 - device_eui, 27
 - DRY, 25
 - i, 27
 - liquid_flag, 27
 - liquid_level, 27
 - liquidLevel, 25
 - liquidPresence, 25
 - loop, 25
 - pir, 25
 - pir_value, 27
 - port, 27
 - resistance, 27
 - setup, 26
 - size_of_buffer, 28
 - socket, 28
 - value, 28
 - VERY_WET, 25
 - voltage, 28
 - WET, 25
- Smart_Gas.cpp
 - adr_flag, 33
 - app_eui, 33
 - app_key, 33
 - APPSensor, 32
 - APPSensor2, 32
 - battery_level, 33
 - bytes, 33
 - CALIBRATION_POINTS, 30
 - co_particles, 34
 - co_resistance, 34
 - co_send, 34
 - co_voltage, 34
 - concentrations_co, 34
 - concentrations_no2, 34
 - COSensor, 34
 - device_eui, 35
 - error, 35
 - i, 35
 - loop, 32
 - luxes, 35
 - luxes_send, 35
 - MINUTESTOSEND, 30
 - no2_particles, 35
 - no2_resistance, 35
 - no2_send, 35
 - no2_voltage, 36

- NO2Sensor, 36
- POINT1_PPM_CO, 30
- POINT1_PPM_NO2, 30
- POINT1_RES_CO, 31
- POINT1_RES_NO2, 31
- POINT2_PPM_CO, 31
- POINT2_PPM_NO2, 31
- POINT2_RES_CO, 31
- POINT2_RES_NO2, 31
- POINT3_PPM_CO, 31
- POINT3_PPM_NO2, 32
- POINT3_RES_CO, 32
- POINT3_RES_NO2, 32
- polutants2_particles, 36
- polutants2_resistance, 36
- polutants2_send, 36
- polutants2_voltage, 36
- polutants_particles, 36
- polutants_resistance, 36
- polutants_send, 37
- polutants_voltage, 37
- port, 37
- resistances_co, 37
- setup, 32
- size_of_buffer, 37
- socket, 37
- voltages_no2, 37
- Smart_Water.cpp
 - adr_flag, 40
 - app_eui, 40
 - app_key, 40
 - battery_level, 40
 - bytes, 40
 - CALIBRATION_POINT_10, 39
 - CALIBRATION_POINT_4, 39
 - CALIBRATION_POINT_7, 39
 - CALIBRATION_TEMPERATURE, 39
 - device_eui, 40
 - loop, 39
 - pH_value, 40
 - pH_value_send, 41
 - pH_voltage, 41
 - pHSensor, 41
 - port, 41
 - setup, 39
 - size_of_buffer, 41
 - socket, 41
 - temperature, 41
 - temperature_send, 42
 - temperatureSensor, 42
- socket
 - Smart_Agriculture.cpp, 22
 - Smart_Events.cpp, 28
 - Smart_Gas.cpp, 37
 - Smart_Water.cpp, 41
- temperature
 - Smart_Agriculture.cpp, 22
 - Smart_Water.cpp, 41
- temperature_send
 - Smart_Water.cpp, 42
- temperatureSensor
 - Smart_Water.cpp, 42
- value
 - Smart_Events.cpp, 28
- vane
 - Smart_Agriculture.cpp, 23
- vane_flag
 - Smart_Agriculture.cpp, 23
- vane_str
 - Smart_Agriculture.cpp, 23
- vaneSensor
 - Smart_Agriculture.cpp, 23
- VERY_WET
 - Smart_Events.cpp, 25
- voltage
 - Smart_Events.cpp, 28
- voltages_no2
 - Smart_Gas.cpp, 37
- watermark1
 - Smart_Agriculture.cpp, 23
- weather
 - Smart_Agriculture.cpp, 23
- WET
 - Smart_Events.cpp, 25
- wetness
 - Smart_Agriculture.cpp, 23
- wmSensor1
 - Smart_Agriculture.cpp, 19