libknx

Generated by Doxygen 1.8.13

Contents

Chapter 1

KNX interface library

Author

Norbert Schmitz, knx@nagilo.de

Date

October 2018

Version

1.2.4

This library can be used to access the home automation bus system KNX using an IP gateway. More information on the bus may be found at www.knx.com .

Changes since version 1.2.3

- Complete redesign of the internal library structure with the conclusion to avoid boost includes on the public header files
- Simplification of the integration due to removed boost compile time dependency
- Set default local IP to 0.0.0.0 which now allows to omit the local host address in many cases

Changes since version 1.2.2

- Added dpts. Now supporting: 1_001,2_001,3_007,4_001,5_001,6_001,7_001,8_001,9_001,10_001,11_001,12_001,13_001,14_000,15_000,16_000,17_001,18_001,19_001,20_011,21_001,26_001,219_001,232_600
- Added class knx::group for group address handling
- Supporting two part group addresses (at receive time 3 part addresses are assumed)

Changes since version 1.2.1

- Added support for dpt 7.001, 8.001
- Bugfix for dpt 9 & 10 (incorrect value setting)

Changes since version 1.0.0

- Added listener interface
- Added several new dpts. Now supporting dpt 1.001, 2.001, 3.007, 4.001, 5.001, 6.001, 9.001, 10.001.

The whole program including the header files are free to be used in any non-commercial application. A notification of usage to the author would be very nice.

Commercial use is strictly forbidden. In case you are interested in a commercial license please contact the author.

Example

A minimal code example would look like this when setting group 1/2/3 to on using a data point type of 1.001 .

```
#include "knx.hpp"
int main(int argc, char ** argv) {
  knx::config config(argc, argv);
  knx::connection connection(config);
  knx::handle handle(connection);
  connection.set<knx::dpt_switch>(knx::group("1/2/3"),
      knx::dpt_switch::ON);
  return 0;
}
```

Features

Although there are many eib/knx libraries available I started to develop an new knx library from scratch. The reasons for this are the following:

- Minimal dependencies:

  Many existing libraries contain dependencies to additional libraries which make the compiltion process harder. libknx depends solely on boost. No other dependencies are allowed.

- Platform independence:

  libknx is portable. It is able to run on any linux operating system including Raspberry PI and on any Mac or Windows computer. The main development is realized on a Linux Debian 7.0 32bit machine but other distros and platforms will follow.

- Reduce to the maximum:

  The library on its own does not contain any additional overhead. It is capable of writing and reading knx messages – nothing more and nothing less. Any additional ideas I or other may have should be based on libknx without extending the base system.

Current limitations

libknx is a brand new development which means that many features are currently missing. Most of them will be added in future. Version 1.2.3 has the following limitations:

- Only UDP communication is available

Ideas

The development of such a library directly creates tons of ideas what can be done with it. Besides others these are:

- Control your curtains using a Kinect
- Control your lights using a LeapMotion
- ...

Quick Start

All packages include three quick test binaries called knx_test_setter, knx_test_getter and knx_test_listener. To quick start you can start the listener. Use the options -l with the IP of the computer you are currently using and -r with the IP of the KNXnet/IP gateway.

```
./knx_test_listener.exe --help
allowed options:
  -h [ --help ]                   produce help message
  -r [ --remote_host ] arg        address of the knx ip gateway
  -p [ --remote_port ] arg        port number of the knx ip gateway
  -l [ --local_host ] arg         address of the local control and data daemon
                                  to be used for communication
  -c [ --local_control_port ] arg port of the control daemon
  -d [ --local_data_port ] arg    port of the data daemon
  -f [ --logging_filename ] arg   name of the file to store logging information
```

Yours Norbert Schmitz

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 knx Namespace Reference

separated namespace to embed all libknx related classes

Classes

- class config

  This class represents the configuration of the knx connection.
- class connection

  This class handles the ip connection(s) to the knx gateway.
- class dpt

  Template definition of each dpt.
- class dpt_10

  Base class container.
- class dpt_10_001

  Representation of time with day, hours, minutes and seconds.
- class dpt_11

  Base class container.
- class dpt_11_001

  Representation of date with day, month and year. Year has a value from 0 to 99 where values bigger than 90 are considered to be 1990+. Remark: Natural end in 2089!.
- class dpt_12_001

  Unsigned 32 bit. Here as counter.
- class dpt_13_001

  Signed 32 bit. Here as counter.
- class dpt_14_000

  Float as 32 bit with IEEE 754 notation. Here as acceleration in ms^ -2.
- class dpt_15

  Base class container.
- class dpt_15_000

  Access data with six 4 byte fields.
- class dpt_16

  Base class container.
- class dpt_16_000

An ascii string with max 14 ascii characters.

- class dpt_17

  Base class container.
- class dpt_17_001

  A 6 bit scene number.
- class dpt_18

  Base class container.
- class dpt_18_001

  A 6 bit scene number with control field.
- class dpt_19

  Base class container.
- class dpt_19_001

  Combinaton of date and time as 8 byte value.
- class dpt_1_001

  Binary single bit value. Either on or off.
- class dpt_2

  Base class for dpt 2.
- class dpt_20_001

  Encoding of a status. Here autonomous(0), skave (1) and master (2).
- class dpt_219

  Base class container.
- class dpt_219_001

  Alarm information.
- class dpt_232

  Base class container.
- class dpt_232_600

  Color information.
- class dpt_26

  Base class container.
- class dpt_26_001

  Scene information with filed indicating if scene is active or not.
- class dpt_2_001

  Binary single bit value with additional control bit. Either on or off.
- class dpt_3

  Base class for dpt 3.
- class dpt_3_007

  Dimming control with direction and step value as 2^ (value-1) subdivisions of the interval 0 to 100 percent.
- class dpt_4_001

  Simple single ascii character.
- class dpt_5_001

  Unsigned value with 8 bit. Scaling represents range 0 to 100 percent.
- class dpt_6_001

  Signed value with 8 bit. Percent represents range -128 to 127 percent.
- class dpt_7_001

  Unsigned value with 16 bit. Range from 0 to 65535.
- class dpt_8_001

  Signed value with 16 bit. Range from -32768 to 32767.
- class dpt_9_001

  Float value with 16 bit. Non IEEE definition. Range from -671088.64 to 670760.96. Here interpreted as temperature in degree celsius.
- class dpt_private

Forward declaration of hidden implementation.

- class group

  This class represents a KNX group address which might be either 3 compnent "a/b/c" or 2 component "a/b".
- class handle

  Thread management for connection handling.
- class MappingMajor

Typedefs

- typedef dpt_1_001 dpt_switch

  Alias for dpt 1.001.
- typedef dpt_10_001 dpt_timeofday

  Alias for dpt 10.001.

Functions

- **KNX_MAP_MAJOR** (1, 1, bool)
- **KNX_MAP_MAJOR** (2, 1, dpt_2::data)
- **KNX_MAP_MAJOR** (3, 1, dpt_3::data)
- **KNX_MAP_MAJOR** (4, 2, char)
- **KNX_MAP_MAJOR** (5, 2, uint8_t)
- **KNX_MAP_MAJOR** (6, 2, int8_t)
- **KNX_MAP_MAJOR** (7, 3, uint16_t)
- **KNX_MAP_MAJOR** (8, 3, int16_t)
- **KNX_MAP_MAJOR** (9, 3, float)
- **KNX_MAP_MAJOR** (10, 4, dpt_10::data)
- **KNX_MAP_MAJOR** (11, 4, dpt_11::data)
- **KNX_MAP_MAJOR** (12, 5, uint32_t)
- **KNX_MAP_MAJOR** (13, 5, int32_t)
- **KNX_MAP_MAJOR** (14, 5, float)
- **KNX_MAP_MAJOR** (15, 5, dpt_15::data)
- **KNX_MAP_MAJOR** (16, 15, dpt_16::data)
- **KNX_MAP_MAJOR** (17, 2, dpt_17::data)
- **KNX_MAP_MAJOR** (18, 2, dpt_18::data)
- **KNX_MAP_MAJOR** (19, 8, dpt_19::data)
- **KNX_MAP_MAJOR** (20, 2, uint8_t)
- **KNX_MAP_MAJOR** (26, 2, dpt_26::data)
- **KNX_MAP_MAJOR** (219, 6, dpt_219::data)
- **KNX_MAP_MAJOR** (232, 3, dpt_232::data)
- template<int MAJOR, int MINOR>
  std::ostream & operator<< (std::ostream &os, dpt< MAJOR, MINOR > &data)

  Major function to output the value of a dpt. (see template of dpt)

6.1.1 Detailed Description

separated namespace to embed all libknx related classes

Chapter 7

Class Documentation

7.1 knx::config Class Reference

This class represents the configuration of the knx connection.

`#include <config.hpp>`

Public Member Functions

- config (int argc, char ∗argv[ ])
  constructor taking all command line arguments
- ∼config ()
  empty descructor

Public Attributes

- std::string local_control_host
  name or ip address of the local host (control connection). Default 0.0.0.0.
- int local_control_port
  communication port for control messages
- std::string local_data_host
  name or ip address of the local host (data connection). Default 0.0.0.0.
- int local_data_port
  communication port for data messages
- std::string remote_host
  name or ip address of the knx gateway
- int remote_port
  port of the gateway (default 3671)
- std::string logging_filename
  name of the logging file
- bool logging_activated
  indication if logging is active

7.1.1 Detailed Description

This class represents the configuration of the knx connection.

The documentation for this class was generated from the following file:

- config.hpp

7.2 knx::connection Class Reference

This class handles the ip connection(s) to the knx gateway.

`#include <connection.hpp>`

Public Member Functions

- connection (const knx::config &config)
  creates the connection with given config
- ~connection ()
  simple destructor
- void start ()
  blocking start of the connection background thread
- void stop ()
  non-blocking stop of the background thread
- template<typename data_type >
  void set (knx::group group, typename data_type::set_type data)
  central function to set any knx group value on the bus
- template<typename data_type >
  bool get (knx::group group, typename data_type::set_type &data)
  central function to get any knx group value from the bus
- template<typename data_type >
  bool listen (knx::group group, std::function< void(typename data_type::set_type &) > callback)
  central function to continously get any knx group value from the bus

7.2.1 Detailed Description

This class handles the ip connection(s) to the knx gateway.

7.2.2 Member Function Documentation

7.2.2.1 get()

```
template<typename data_type >
bool knx::connection::get (
            knx::group group,
            typename data_type::set_type & data )
```

central function to get any knx group value from the bus

This function is used to get any value from the knx bus.

Parameters

group	A valid group id as string (e.g. "1/2/3")

Returns

true if read was successful – false otherwise

7.2.2.2 listen()

```
template<typename data_type >
bool knx::connection::listen (
            knx::group group,
            std::function< void(typename data_type::set_type &) > callback )
```

central function to continously get any knx group value from the bus

This function is used to listen for any value from the knx bus.

A minimal example would be:

```
#include "knx.hpp"
#include "debug.hpp"
#include <thread>
#include <chrono>
void my_callback ( knx::dpt_1_001::set_type & data )
{
    std::cout << "listened to " << data << std::endl;
}
int main ( int argc, char ** argv )
{
    knx::config config ( argc, argv );
    knx::connection connection ( config );
    knx::handle handle ( connection );
    std::this_thread::sleep_for(std::chrono::seconds(2));
    connection.listen<knx::dpt_1_001> ( "0/0/1", my_callback );
    std::this_thread::sleep_for(std::chrono::seconds(2));
    return 0;
}
```

Parameters

group	A valid group id as string (e.g. "1/2/3")
callback	A function to be called whenever data has been received

Returns

true if registering was successful – false otherwise

7.2.2.3 set()

```
template<typename data_type >
void knx::connection::set (
```

```
            knx::group group,
            typename data_type::set_type data )
```

central function to set any knx group value on the bus

This function is used to set any value on the knx bus.

7.2.2.4 start()

```
void knx::connection::start ( )
```

blocking start of the connection background thread

This function is a blocking call to start the background thread. It is normally only called by the handler.

Warning

you should not call this function from your code.

7.2.2.5 stop()

```
void knx::connection::stop ( )
```

non-blocking stop of the background thread

This function send the termination signals and waits for the thread to stop.

The documentation for this class was generated from the following file:

- connection.hpp

7.3 knx::dpt_3::data Class Reference

Data container for this dpt.

```
#include <dpt_types.hpp>
```

Public Member Functions

- data (bool c, uint8_t v)
  Standard constructor with values.
- data ()
  Standard constructor.

Public Attributes

- bool control
  Control bit.
- uint8_t value
  Control field.

Friends

- KNX_IMPORT_EXPORT friend std::ostream & operator<< (std::ostream &os, const data &time)
  Output formatting.

7.3.1 Detailed Description

Data container for this dpt.

The documentation for this class was generated from the following file:

- dpt_types.hpp

7.4 knx::dpt_10::data Class Reference

Data container for this dpt.

`#include <dpt_types.hpp>`

Public Member Functions

- bool operator== (const data &other)
  Comparison for time objects.
- data ()
  creates a current time value

Public Attributes

- uint8_t day
  day of the week monday (1) till sunday (7)
- uint8_t hour
  From 0 to 23.
- uint8_t minutes
  From 0 to 59.
- uint8_t seconds
  From 0 to 59.

Static Public Attributes

- static const uint8_t NODAY = 0
  No day assigned or unknown.
- static const uint8_t MONDAY = 1
  Monday.
- static const uint8_t TUESDAY = 2
  Tuesday.
- static const uint8_t WEDNESDAY = 3
  Wednesday.
- static const uint8_t THURSDAY = 4
  Thursday.
- static const uint8_t FRIDAY = 5
  Friday.
- static const uint8_t SATURDAY = 6
  Saturday.
- static const uint8_t SUNDAY = 7
  Sunday.

Friends

- KNX_IMPORT_EXPORT friend std::ostream & operator<< (std::ostream &os, const data &time)
  Output formatting.

7.4.1 Detailed Description

Data container for this dpt.

The documentation for this class was generated from the following file:

- dpt_types.hpp

7.5 knx::dpt_11::data Class Reference

Data container for this dpt.

`#include <dpt_types.hpp>`

Public Member Functions

- bool operator== (const data &other)
  Compare the date.
- data ()
  creates a current date value

Public Attributes

- uint8_t day

  Day of month.
- uint8_t month

  Month of year.
- uint8_t year

  Year with two digits. >=90 is expected to start with 19XX.

Friends

- KNX_IMPORT_EXPORT friend std::ostream & operator<< (std::ostream &os, const data &date)

  Output formatting.

7.5.1 Detailed Description

Data container for this dpt.

The documentation for this class was generated from the following file:

- dpt_types.hpp

7.6 knx::dpt_15::data Class Reference

Data container for this dpt.

```
#include <dpt_types.hpp>
```

Public Member Functions

- bool operator== (const data &other)

  Compare access data.
- data ()

  Default constructor setting to zero.

Public Attributes

- struct {
  unsigned int d6: 4
  digit6
  unsigned int d5: 4
  digit5
  unsigned int d4: 4
  digit4
  unsigned int d3: 4
  digit3
  unsigned int d2: 4
  digit2
  unsigned int d1: 4
  digit1
  bool e: 1
  Error detection (0 is no error).
  bool p: 1
  Permission (1 is accepted).
  bool c: 1
  Encryption (1 is yes).
  bool d: 1
  Read direction (0 is left to right).
  unsigned int index: 4
  Index of access identification code.
  } storage

  The data storage.

Friends

- KNX_IMPORT_EXPORT friend std::ostream & operator<< (std::ostream &os, const data &data)

  Output formatting.

7.6.1 Detailed Description

Data container for this dpt.

The documentation for this class was generated from the following file:

- dpt_types.hpp

7.7 knx::dpt_16::data Class Reference

Data container for this dpt.

`#include <dpt_types.hpp>`

Public Member Functions

- bool operator== (const data &other)

  Compare it.
- data ()

  Setting all to zero.

Public Attributes

- struct {
  char ascii [14]
  The string value.
  } storage

  Storage.

Friends

- KNX_IMPORT_EXPORT friend std::ostream & operator<< (std::ostream &os, const data &data)

  Output formatting.

7.7.1 Detailed Description

Data container for this dpt.

The documentation for this class was generated from the following file:

- dpt_types.hpp

7.8 knx::dpt_2::data Class Reference

Data container for this dpt.

`#include <dpt_types.hpp>`

Public Member Functions

- data (bool c, bool v)

  Constructor with control and value.
- data ()

  Standard Constructor setting all to zero.

Public Attributes

- bool control
  Control bit.
- bool value
  Value bit.

Friends

- KNX_IMPORT_EXPORT friend std::ostream & operator<< (std::ostream &os, const data &time)
  Output formatting.

7.8.1 Detailed Description

Data container for this dpt.

The documentation for this class was generated from the following file:

- dpt_types.hpp

7.9 knx::dpt_17::data Class Reference

Data container for this dpt.

`#include <dpt_types.hpp>`

Public Member Functions

- bool operator== (const data &other)
  Compare.
- data ()
  Set to zero.

Public Attributes

-
  struct {
    unsigned int r: 2
      Reserved do not use!.
    unsigned int number: 6
      Scene number with 6 bits.
  } storage

  Data storage.

Friends

- KNX_IMPORT_EXPORT friend std::ostream & operator<< (std::ostream &os, const data &data)

  Output formatting.

7.9.1 Detailed Description

Data container for this dpt.

The documentation for this class was generated from the following file:

- dpt_types.hpp

7.10 knx::dpt_18::data Class Reference

Data container for this dpt.

`#include <dpt_types.hpp>`

Public Member Functions

- bool operator== (const data &other)

  Compare.
- data ()

  Set to zero.

Public Attributes

-
  struct {
    unsigned int c: 1
      0 means activate and 1 learn the scene
    unsigned int r: 1
      do not use
    unsigned int number: 6
      scene nuber
  } storage

  Data storage.

Friends

- KNX_IMPORT_EXPORT friend std::ostream & operator<< (std::ostream &os, const data &data)

  Output formatting.

7.10.1 Detailed Description

Data container for this dpt.

The documentation for this class was generated from the following file:

- dpt_types.hpp

7.11 knx::dpt_19::data Class Reference

Data container for this dpt.

`#include <dpt_types.hpp>`

Public Member Functions

- bool operator== (const data &other)
  Compare.
- data ()
  Set to zero.

Public Attributes

- struct {
  unsigned int year: 8
  Add 1900 to get real year.
  unsigned int res0: 4
  do not use
  unsigned int month: 4
  January is 1, February 2, ...
  unsigned int res1: 3
  do not use
  unsigned int day_of_month: 5
  From 1st to 31st day.
  unsigned int day_of_week: 3
  Monday is 1, Tuesday is 2, ..
  unsigned int hour_of_day: 5
  As usual.
  unsigned int res2: 2
  do not use
  unsigned int minutes: 6
  Minutes till 59.
  unsigned int res3: 2
  do not use
  unsigned int seconds: 6
  Seconds till 59.
  unsigned int f: 1
  Fault with 1 is true.
  unsigned int wd: 1
  Set to 1 if it is a working day.

unsigned int nwd: 1
Indicates if wd field is valid. 1 means not valid.
unsigned int ny: 1
Set to one if year is not valid.
unsigned int nd: 1
Set to one if day and month are not valid.
unsigned int ndow: 1
Set to one if day of week is not valid.
unsigned int nt: 1
Set to one if time is not valid.
unsigned int suti: 1
Summer time field. Add one hour if set to one.
unsigned int clq: 1
Quality of clock. 1 says with external signal.
unsigned int res4: 7
do not use
} storage

Data storage.

Friends

- KNX_IMPORT_EXPORT friend std::ostream & operator<< (std::ostream &os, const data &data)
  Output formatting.

7.11.1 Detailed Description

Data container for this dpt.

The documentation for this class was generated from the following file:

- dpt_types.hpp

7.12 knx::dpt_26::data Class Reference

Data container for this dpt.

`#include <dpt_types.hpp>`

Public Member Functions

- bool operator== (const data &other)
  Compare.
- data ()
  Set to zero.

Public Attributes

- struct {
  unsigned int reserved: 1
  Do not use.
  unsigned int active: 1
  1 says scene is inactive.
  unsigned int scene_number: 6
  The number of the scene.
  } storage

  Data storage.

Friends

- KNX_IMPORT_EXPORT friend std::ostream & operator<< (std::ostream &os, const data &data)

  Output formatting.

7.12.1 Detailed Description

Data container for this dpt.

The documentation for this class was generated from the following file:

- dpt_types.hpp

7.13 knx::dpt_219::data Class Reference

Data container for this dpt.

`#include <dpt_types.hpp>`

Public Member Functions

- bool operator== (const data &other)

  Compare.
- data ()

  Set to zero.

Public Attributes

- struct {
  unsigned char log_number
  Number of the current log.
  unsigned char alarm_priority
  Priority with 0 is highest.
  unsigned char application_area
  Area of error (there is a table where 0 is no fault and 20 is lighting (for example)
  unsigned char error_class
  Class of error. 0 is no fault. 4 is hw fault (table exists)
  unsigned int reserverd0: 2
  Do not use.
  unsigned int attributes: 6
  Bit field with 0 (ack), 1 (timestamp), 2 (alarmtext), 3 (errorcode) supressed.
  unsigned int reserverd1: 5
  Do not use.
  unsigned int alarm_status_attributes: 3
  Alarm status attributes with 0 (inalarm), 1 (alarmunack) and 2 (locked)
  } storage

  Data storage.

Friends

- KNX_IMPORT_EXPORT friend std::ostream & operator<< (std::ostream &os, const data &data)
  Output formatting.

7.13.1 Detailed Description

Data container for this dpt.

The documentation for this class was generated from the following file:

- dpt_types.hpp

7.14 knx::dpt_232::data Class Reference

Data container for this dpt.

`#include <dpt_types.hpp>`

Public Member Functions

- bool operator== (const data &other)
  Compare.
- data ()
  Set to zero.

Public Attributes

- struct {
  unsigned char **red**
  unsigned char **green**
  unsigned char **blue**
  } storage

  Data storage.

Friends

- KNX_IMPORT_EXPORT friend std::ostream & operator<< (std::ostream &os, const data &data)

  Output formatting.

7.14.1 Detailed Description

Data container for this dpt.

The documentation for this class was generated from the following file:

- dpt_types.hpp

7.15 knx::dpt< MAJOR, MINOR > Class Template Reference

Template definition of each dpt.

`#include <dpt.hpp>`

Public Types

- typedef MappingMajor< MAJOR >::set_type set_type

  Fetching the set value type from mapping.
- typedef dpt< MAJOR, MINOR > own_type

  Assigning the type for later access.
- typedef dpt_private< MAJOR, MINOR > private_type

  Assigning type of private implementation.

Public Member Functions

- dpt ()

  Constructor usually setting all values to zero.
- ~dpt ()

  Destructor for the dpt.
- template<class TYPE >
  void set (const TYPE value)

  Major function to set the value of a dpt.
- template<class TYPE >
  void get (TYPE &value) const

  Major function to get the value of a dpt.

Static Public Member Functions

- static unsigned int get_index ()

  Function returning an index of the class for quick type access.

Friends

- template<int MAJOR2, int MINOR2>
  std::ostream & operator<< (std::ostream &os, dpt< MAJOR2, MINOR2 > &data)

  Major function to output the value of a dpt.

7.15.1 Detailed Description

template<int MAJOR, int MINOR>
class knx::dpt< MAJOR, MINOR >

Template definition of each dpt.

The documentation for this class was generated from the following file:

- dpt.hpp

7.16 knx::dpt_10 Class Reference

Base class container.

`#include <dpt_types.hpp>`

Classes

- class data

  Data container for this dpt.

7.16.1 Detailed Description

Base class container.

The documentation for this class was generated from the following file:

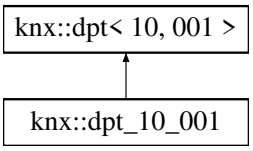
- dpt_types.hpp

7.17 knx::dpt_10_001 Class Reference

Representation of time with day, hours, minutes and seconds.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_10_001:



Public Types

- typedef dpt_10::data **time**

Additional Inherited Members

7.17.1 Detailed Description

Representation of time with day, hours, minutes and seconds.

The documentation for this class was generated from the following file:

- dpt.hpp

7.18 knx::dpt_11 Class Reference

Base class container.

`#include <dpt_types.hpp>`

Classes

- class data

  Data container for this dpt.

7.18.1 Detailed Description

Base class container.

The documentation for this class was generated from the following file:

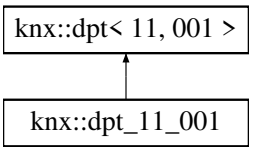
- dpt_types.hpp

7.19 knx::dpt_11_001 Class Reference

Representation of date with day, month and year. Year has a value from 0 to 99 where values bigger than 90 are considered to be 1990+. Remark: Natural end in 2089!.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_11_001:



Public Types

- typedef dpt_11::data **date**

Additional Inherited Members

7.19.1 Detailed Description

Representation of date with day, month and year. Year has a value from 0 to 99 where values bigger than 90 are considered to be 1990+. Remark: Natural end in 2089!.

The documentation for this class was generated from the following file:

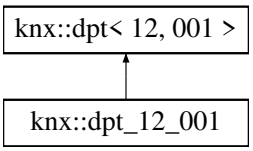
- dpt.hpp

7.20 knx::dpt_12_001 Class Reference

Unsigned 32 bit. Here as counter.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_12_001:



Additional Inherited Members

7.20.1 Detailed Description

Unsigned 32 bit. Here as counter.

The documentation for this class was generated from the following file:

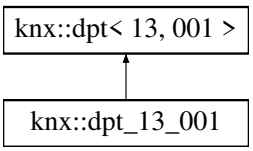
- dpt.hpp

7.21 knx::dpt_13_001 Class Reference

Signed 32 bit. Here as counter.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_13_001:



Additional Inherited Members

7.21.1 Detailed Description

Signed 32 bit. Here as counter.

The documentation for this class was generated from the following file:

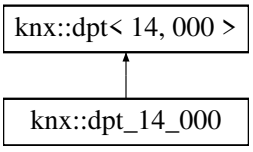
- dpt.hpp

7.22 knx::dpt_14_000 Class Reference

Float as 32 bit with IEEE 754 notation. Here as acceleration in ms^-2.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_14_000:



Additional Inherited Members

7.22.1 Detailed Description

Float as 32 bit with IEEE 754 notation. Here as acceleration in ms^-2.

The documentation for this class was generated from the following file:

- dpt.hpp

7.23 knx::dpt_15 Class Reference

Base class container.

`#include <dpt_types.hpp>`

Classes

- class data

  Data container for this dpt.

7.23.1 Detailed Description

Base class container.

The documentation for this class was generated from the following file:

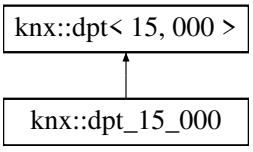
- dpt_types.hpp

7.24 knx::dpt_15_000 Class Reference

Access data with six 4 byte fields.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_15_000:



Additional Inherited Members

7.24.1 Detailed Description

Access data with six 4 byte fields.

The documentation for this class was generated from the following file:

- dpt.hpp

7.25 knx::dpt_16 Class Reference

Base class container.

`#include <dpt_types.hpp>`

Classes

- class data

  Data container for this dpt.

7.25.1 Detailed Description

Base class container.

The documentation for this class was generated from the following file:

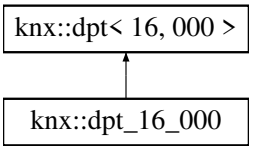
- dpt_types.hpp

7.26 knx::dpt_16_000 Class Reference

An ascii string with max 14 ascii characters.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_16_000:



Additional Inherited Members

7.26.1 Detailed Description

An ascii string with max 14 ascii characters.

The documentation for this class was generated from the following file:

- dpt.hpp

7.27 knx::dpt_17 Class Reference

Base class container.

`#include <dpt_types.hpp>`

Classes

- class data

  Data container for this dpt.

7.27.1 Detailed Description

Base class container.

The documentation for this class was generated from the following file:

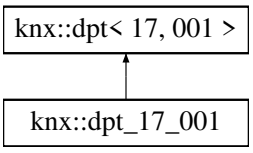
- dpt_types.hpp

7.28 knx::dpt_17_001 Class Reference

A 6 bit scene number.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_17_001:



Additional Inherited Members

7.28.1 Detailed Description

A 6 bit scene number.

The documentation for this class was generated from the following file:

- dpt.hpp

7.29 knx::dpt_18 Class Reference

Base class container.

`#include <dpt_types.hpp>`

Classes

- class data

  Data container for this dpt.

7.29.1 Detailed Description

Base class container.

The documentation for this class was generated from the following file:

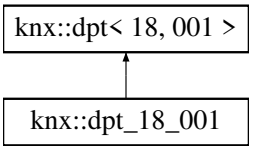
- dpt_types.hpp

7.30 knx::dpt_18_001 Class Reference

A 6 bit scene number with control field.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_18_001:



Additional Inherited Members

7.30.1 Detailed Description

A 6 bit scene number with control field.

The documentation for this class was generated from the following file:

- dpt.hpp

7.31 knx::dpt_19 Class Reference

Base class container.

`#include <dpt_types.hpp>`

Classes

- class data

  Data container for this dpt.

7.31.1 Detailed Description

Base class container.

The documentation for this class was generated from the following file:

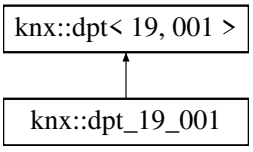
- dpt_types.hpp

7.32 knx::dpt_19_001 Class Reference

Combinaton of date and time as 8 byte value.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_19_001:



Additional Inherited Members

7.32.1 Detailed Description

Combinaton of date and time as 8 byte value.

The documentation for this class was generated from the following file:

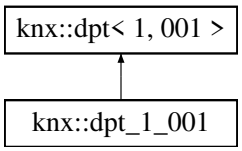
- dpt.hpp

7.33 knx::dpt_1_001 Class Reference

Binary single bit value. Either on or off.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_1_001:



Static Public Attributes

- static KNX_IMPORT_EXPORT const dpt< 1, 001 >::set_type ON
  Static variable for switching on.
- static KNX_IMPORT_EXPORT const dpt< 1, 001 >::set_type OFF
  Static variable for switching off.

Additional Inherited Members

7.33.1 Detailed Description

Binary single bit value. Either on or off.

The documentation for this class was generated from the following file:

- dpt.hpp

7.34 knx::dpt_2 Class Reference

Base class for dpt 2.

`#include <dpt_types.hpp>`

Classes

- class data

  Data container for this dpt.

7.34.1 Detailed Description

Base class for dpt 2.

The documentation for this class was generated from the following file:

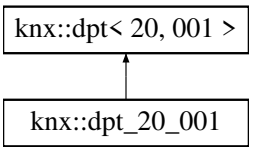
- dpt_types.hpp

7.35 knx::dpt_20_001 Class Reference

Encoding of a status. Here autonomous(0), skave (1) and master (2).

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_20_001:



Additional Inherited Members

7.35.1 Detailed Description

Encoding of a status. Here autonomous(0), skave (1) and master (2).

The documentation for this class was generated from the following file:

- dpt.hpp

7.36 knx::dpt_219 Class Reference

Base class container.

`#include <dpt_types.hpp>`

Classes

- class data

  Data container for this dpt.

7.36.1 Detailed Description

Base class container.

The documentation for this class was generated from the following file:

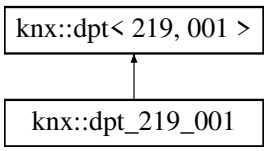
- dpt_types.hpp

7.37 knx::dpt_219_001 Class Reference

Alarm information.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_219_001:



Additional Inherited Members

7.37.1 Detailed Description

Alarm information.

The documentation for this class was generated from the following file:

- dpt.hpp

7.38 knx::dpt_232 Class Reference

Base class container.

`#include <dpt_types.hpp>`

Classes

- class data

  Data container for this dpt.

7.38.1 Detailed Description

Base class container.

The documentation for this class was generated from the following file:

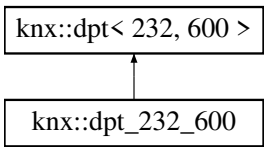
- dpt_types.hpp

7.39 knx::dpt_232_600 Class Reference

Color information.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_232_600:



Additional Inherited Members

7.39.1 Detailed Description

Color information.

The documentation for this class was generated from the following file:

- dpt.hpp

7.40 knx::dpt_26 Class Reference

Base class container.

`#include <dpt_types.hpp>`

Classes

- class data

  Data container for this dpt.

7.40.1 Detailed Description

Base class container.

The documentation for this class was generated from the following file:

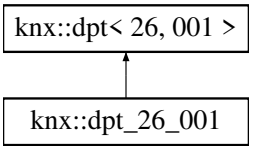
- dpt_types.hpp

7.41 knx::dpt_26_001 Class Reference

Scene information with filed indicating if scene is active or not.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_26_001:



Additional Inherited Members

7.41.1 Detailed Description

Scene information with filed indicating if scene is active or not.

The documentation for this class was generated from the following file:

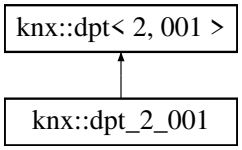
- dpt.hpp

7.42 knx::dpt_2_001 Class Reference

Binary single bit value with additional control bit. Either on or off.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_2_001:



Static Public Attributes

- static KNX_IMPORT_EXPORT const dpt< 2, 001 >::set_type NO_CONTROL
  Control bit is off so value is not used.
- static KNX_IMPORT_EXPORT const dpt< 2, 001 >::set_type CONTROL_VALUE_ZERO
  Control bit is on and value is off.
- static KNX_IMPORT_EXPORT const dpt< 2, 001 >::set_type CONTROL_VALUE_ONE
  Control bit is on and value is on.

Additional Inherited Members

7.42.1 Detailed Description

Binary single bit value with additional control bit. Either on or off.

The documentation for this class was generated from the following file:

- dpt.hpp

7.43 knx::dpt_3 Class Reference

Base class for dpt 3.

`#include <dpt_types.hpp>`

Classes

- class data
  Data container for this dpt.

7.43.1 Detailed Description

Base class for dpt 3.

The documentation for this class was generated from the following file:

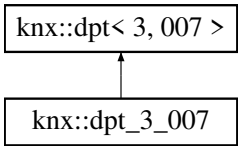
- dpt_types.hpp

7.44 knx::dpt_3_007 Class Reference

Dimming control with direction and step value as 2^(value-1) subdivisions of the interval 0 to 100 percent.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_3_007:



Additional Inherited Members

7.44.1 Detailed Description

Dimming control with direction and step value as 2^(value-1) subdivisions of the interval 0 to 100 percent.

The documentation for this class was generated from the following file:

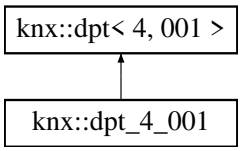
- dpt.hpp

7.45 knx::dpt_4_001 Class Reference

Simple single ascii character.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_4_001:



Additional Inherited Members

7.45.1 Detailed Description

Simple single ascii character.

The documentation for this class was generated from the following file:

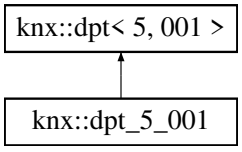
- dpt.hpp

7.46 knx::dpt_5_001 Class Reference

Unsigned value with 8 bit. Scaling represents range 0 to 100 percent.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_5_001:



Additional Inherited Members

7.46.1 Detailed Description

Unsigned value with 8 bit. Scaling represents range 0 to 100 percent.

The documentation for this class was generated from the following file:

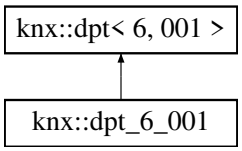
- dpt.hpp

7.47 knx::dpt_6_001 Class Reference

Signed value with 8 bit. Percent represents range -128 to 127 percent.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_6_001:



Additional Inherited Members

7.47.1 Detailed Description

Signed value with 8 bit. Percent represents range -128 to 127 percent.

The documentation for this class was generated from the following file:

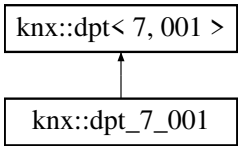
- dpt.hpp

7.48 knx::dpt_7_001 Class Reference

Unsigned value with 16 bit. Range from 0 to 65535.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_7_001:



Additional Inherited Members

7.48.1 Detailed Description

Unsigned value with 16 bit. Range from 0 to 65535.

The documentation for this class was generated from the following file:

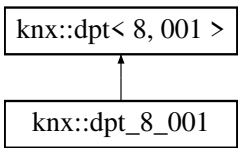
- dpt.hpp

7.49 knx::dpt_8_001 Class Reference

Signed value with 16 bit. Range from -32768 to 32767.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_8_001:



Additional Inherited Members

7.49.1 Detailed Description

Signed value with 16 bit. Range from -32768 to 32767.

The documentation for this class was generated from the following file:

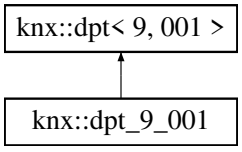
- dpt.hpp

7.50 knx::dpt_9_001 Class Reference

Float value with 16 bit. Non IEEE definition. Range from -671088.64 to 670760.96. Here interpreted as temperature in degree celsius.

`#include <dpt.hpp>`

Inheritance diagram for knx::dpt_9_001:



Static Public Attributes

- static KNX_IMPORT_EXPORT const uint16_t INVALID_DATA
  Invalid value.
- static KNX_IMPORT_EXPORT const uint16_t MAX
  Maximal value in binary.
- static KNX_IMPORT_EXPORT const uint16_t MIN
  Minimal value in binary.
- static KNX_IMPORT_EXPORT const float MAX_FLOAT
  Maximal value as float.
- static KNX_IMPORT_EXPORT const float MIN_FLOAT
  Minimal value as float.

Additional Inherited Members

7.50.1 Detailed Description

Float value with 16 bit. Non IEEE definition. Range from -671088.64 to 670760.96. Here interpreted as temperature in degree celsius.

The documentation for this class was generated from the following file:

- dpt.hpp

7.51 knx::dpt_private< MAJOR, MINOR > Class Template Reference

Forward declaration of hidden implementation.

`#include <dpt.hpp>`

7.51.1 Detailed Description

template<int MAJOR, int MINOR>
class knx::dpt_private< MAJOR, MINOR >

Forward declaration of hidden implementation.

The documentation for this class was generated from the following file:

- dpt.hpp

7.52 knx::group Class Reference

This class represents a KNX group address which might be either 3 compnent "a/b/c" or 2 component "a/b".

`#include <group.hpp>`

Public Member Functions

- group ()
  Sets group to 0/0/0 by default.
- group (std::string name)
  Assigns a group address by name. Sets default 0/0/0 if string is invalid. 2 component addresses may be used.
- group (uint8_t a, uint8_t b, uint8_t c)
  Binary assign a 3 component group address.
- group (uint8_t a, uint8_t b)
  Binary assign a 2 component group address.
- std::string get_name ()
- uint16_t get_binary ()
- void set_by_number (uint16_t a, uint16_t b, uint16_t c)
- void set_by_number (uint16_t a, uint16_t b)
- void set_by_name (std::string name)
- void set_by_binary_2 (uint16_t binary)
- void set_by_binary_3 (uint16_t binary)
- bool operator== (knx::group other)
  Comparison operator for rapid equality checks.

7.52.1 Detailed Description

This class represents a KNX group address which might be either 3 compnent "a/b/c" or 2 component "a/b".

7.52.2 Member Function Documentation

7.52.2.1 get_binary()

```
uint16_t knx::group::get_binary ( )
```

Get the binary group address value as 16 bit int.

Returns

Binary representation of the group address.

7.52.2.2 get_name()

```
std::string knx::group::get_name ( )
```

Gives the current set address as string.

Returns

The group address as string.

7.52.2.3 set_by_binary_2()

```
void knx::group::set_by_binary_2 (
            uint16_t binary )
```

Sets the whole address from binary value guessing a 2 comonent address in the form "a/b".

Parameters

binary	The address as 16 bit uint.

7.52.2.4 set_by_binary_3()

```
void knx::group::set_by_binary_3 (
            uint16_t binary )
```

Sets the whole address from binary value guessing a 3 comonent address in the form "a/b/c".

Parameters

binary	The address as 16 bit uint.

7.52.2.5 set_by_name()

```
void knx::group::set_by_name (
            std::string name )
```

Set the address by name either as "1/2" or "1/2/3".

7.52.2.6 set_by_number() [1/2]

```
void knx::group::set_by_number (
            uint16_t a,
            uint16_t b,
            uint16_t c )
```

Set the address by a set of components in the form "a/b/c". If any value exceeds the range higer bits are ignored.

Parameters

a	First component 5 bits (0-31).
b	Second component 3 bits (0-7).
c	Third component 8 bits (0-255).

7.52.2.7 set_by_number() [2/2]

```
void knx::group::set_by_number (
            uint16_t a,
            uint16_t b )
```

Set the address by a set of components in the form "a/b". If any value exceeds the range higer bits are ignored.

Parameters

a	First component 5 bits (0-31).
b	Second component 11 bits (0-2047).

The documentation for this class was generated from the following file:

- group.hpp

7.53 knx::handle Class Reference

Thread management for connection handling.

`#include <handle.hpp>`

Public Member Functions

- handle (knx::connection &connection)
  Creates new background thread.
- ~handle ()
  Stops and joins the thread.

7.53.1 Detailed Description

Thread management for connection handling.

This class handles all required background threads for asynchronous knx connection handling.

The documentation for this class was generated from the following file:

- handle.hpp

7.54 knx::MappingMajor< MAJOR > Class Template Reference

The documentation for this class was generated from the following file:

- dpt.hpp

Chapter 8

File Documentation

8.1 config.hpp File Reference

```
#include "debug.hpp"
#include <string>
```

Classes

- class knx::config

  This class represents the configuration of the knx connection.

Namespaces

- knx

  separated namespace to embed all libknx related classes

8.1.1 Detailed Description

This class holds the current configuration of the knx connection.

Definition of all known dpt types.

8.2 connection.hpp File Reference

```
#include <functional>
#include "debug.hpp"
#include "config.hpp"
#include "dpt.hpp"
#include "group.hpp"
```

Classes

- class knx::connection

  This class handles the ip connection(s) to the knx gateway.

Namespaces

- knx

  separated namespace to embed all libknx related classes

8.2.1 Detailed Description

User interface class for setting, getting and listening.

8.3 group.hpp File Reference

```
#include <stdint.h>
#include <string>
#include <sstream>
```

Classes

- class knx::group

  This class represents a KNX group address which might be either 3 compnent "a/b/c" or 2 component "a/b".

Namespaces

- knx

  separated namespace to embed all libknx related classes

8.3.1 Detailed Description

Class file for group address handling.

8.4 handle.hpp File Reference

```
#include "debug.hpp"
#include "connection.hpp"
#include <thread>
```

Classes

- class knx::handle

  Thread management for connection handling.

Namespaces

- knx

  separated namespace to embed all libknx related classes

8.4.1 Detailed Description

This file contains the helper class for handling knx access.

8.5 knx.hpp File Reference

```
#include "config.hpp"
#include "connection.hpp"
#include "handle.hpp"
#include "dpt.hpp"
```

8.5.1 Detailed Description

General include header for libknx.

Index