

libknx

Generated by Doxygen 1.8.8

Mon Dec 22 2014 00:25:51

Contents

1	KNX interface library	1
2	Namespace Index	3
2.1	Namespace List	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Namespace Documentation	9
5.1	knx Namespace Reference	9
5.1.1	Detailed Description	10
6	Class Documentation	11
6.1	knx::config Class Reference	11
6.1.1	Detailed Description	11
6.2	knx::connection Class Reference	12
6.2.1	Detailed Description	12
6.2.2	Member Function Documentation	12
6.2.2.1	get	12
6.2.2.2	listen	12
6.2.2.3	set	13
6.2.2.4	start	13
6.2.2.5	stop	13
6.3	knx::dpt_6::data Class Reference	13
6.4	knx::dpt_4::data Class Reference	14
6.5	knx::dpt_9::data Class Reference	14
6.6	knx::dpt_5::data Class Reference	14
6.7	knx::data_point< dpt > Class Template Reference	15
6.8	knx::dpt Class Reference	15
6.9	knx::dpt_1 Class Reference	15

6.9.1 Detailed Description	15
6.10 knx::dpt_10 Class Reference	16
6.10.1 Detailed Description	16
6.11 knx::dpt_10_001 Class Reference	16
6.11.1 Detailed Description	17
6.12 knx::dpt_1_001 Class Reference	17
6.12.1 Detailed Description	17
6.13 knx::dpt_2 Class Reference	17
6.13.1 Detailed Description	18
6.14 knx::dpt_2_001 Class Reference	18
6.14.1 Detailed Description	18
6.15 knx::dpt_3 Class Reference	18
6.15.1 Detailed Description	19
6.16 knx::dpt_3_007 Class Reference	19
6.16.1 Detailed Description	19
6.17 knx::dpt_4 Class Reference	19
6.17.1 Detailed Description	20
6.18 knx::dpt_4_001 Class Reference	20
6.18.1 Detailed Description	20
6.19 knx::dpt_5 Class Reference	20
6.19.1 Detailed Description	21
6.20 knx::dpt_5_001 Class Reference	21
6.20.1 Detailed Description	21
6.21 knx::dpt_6 Class Reference	22
6.21.1 Detailed Description	22
6.22 knx::dpt_6_001 Class Reference	22
6.22.1 Detailed Description	23
6.23 knx::dpt_7 Class Reference	23
6.23.1 Detailed Description	23
6.24 knx::dpt_7_001 Class Reference	23
6.24.1 Detailed Description	24
6.25 knx::dpt_8 Class Reference	24
6.25.1 Detailed Description	24
6.26 knx::dpt_8_001 Class Reference	24
6.26.1 Detailed Description	25
6.27 knx::dpt_9 Class Reference	25
6.27.1 Detailed Description	25
6.28 knx::dpt_9_001 Class Reference	25
6.28.1 Detailed Description	26
6.29 knx::handle Class Reference	26

6.29.1 Detailed Description	26
6.30 knx::dpt_10::time Class Reference	26
6.30.1 Detailed Description	27
7 File Documentation	29
7.1 knx.hpp File Reference	29
7.1.1 Detailed Description	29
Index	30

Chapter 1

KNX interface library

Author

Norbert Schmitz, knx@nagilo.de

Date

November 2014

Version

1.2.2

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 1.2.1

- 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>("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 compilation 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 1.1.0 has the following limitations:

- Only 3-part group addresses are available
- Only UDP communication is available
- Only the following data point types are available: 1.001(rw) 5.001(r) 10.001(w)

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 Mac
- ...

Yours Norbert Schmitz

Chapter 2

Namespace Index

2.1 Namespace List

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

knx	Separated namespace to embed all libknx related classes	9
---------------------	---	---

Chapter 3

Class Index

3.1 Class List

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

knx::config	This class represents the configuration of the knx connection	11
knx::connection	This class handles the ip connection(s) to the knx gateway	12
knx::dpt_6::data	13
knx::dpt_4::data	14
knx::dpt_9::data	14
knx::dpt_5::data	14
knx::data_point< dpt >	15
knx::dpt	15
knx::dpt_1	Major class for all 1.XXX data point types	15
knx::dpt_10	Major class for all 10.XXX data point types	16
knx::dpt_10_001	Class holding a time value	16
knx::dpt_1_001	Data point type 1.001 simple boolean value	17
knx::dpt_2	Major class for all 2.XXX data point types	17
knx::dpt_2_001	Data point type 2.001 control value	18
knx::dpt_3	Major class for all 3.XXX data point types	18
knx::dpt_3_007	Data point type 3.007 dimming control	19
knx::dpt_4	Major class for all 4.XXX data point types	19
knx::dpt_4_001	Data point type 4.001 char ascii	20
knx::dpt_5	Major class for all 5.XXX data point types	20
knx::dpt_5_001	Data point type 5.001 unsigned 8 bit	21
knx::dpt_6	Major class for all 6.XXX signed 8 bit value	22
knx::dpt_6_001	Data point type 6.001 percent v8	22

knx::dpt_7	Major class for all 7.XXX unsigned signed 16 bit value	23
knx::dpt_7_001	Data point type 7.001 unsigned 16bit pulse counter	23
knx::dpt_8	Major class for all 8.XXX signed 16 bit value	24
knx::dpt_8_001	Data point type 8.001 16bit signed pulse counter	24
knx::dpt_9	Major class for all 9.XXX data point types	25
knx::dpt_9_001	Data point type 9.001 value temp	25
knx::handle	Thread management for connection handling	26
knx::dpt_10::time	Internal class storing a time value	26

Chapter 4

File Index

4.1 File List

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

config.hpp	??
connection.hpp	??
data_point.hpp	??
dpt.hpp	??
handle.hpp	??
knx.hpp	29

Chapter 5

Namespace Documentation

5.1 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 [data_point](#)
- class [dpt](#)
- class [dpt_1](#)
major class for all 1.XXX data point types
- class [dpt_10](#)
major class for all 10.XXX data point types
- class [dpt_10_001](#)
class holding a time value
- class [dpt_1_001](#)
data point type 1.001 simple boolean value
- class [dpt_2](#)
major class for all 2.XXX data point types
- class [dpt_2_001](#)
data point type 2.001 control value
- class [dpt_3](#)
major class for all 3.XXX data point types
- class [dpt_3_007](#)
data point type 3.007 dimming control
- class [dpt_4](#)
major class for all 4.XXX data point types
- class [dpt_4_001](#)
data point type 4.001 char ascii
- class [dpt_5](#)
major class for all 5.XXX data point types
- class [dpt_5_001](#)
data point type 5.001 unsigned 8 bit

- class [dpt_6](#)
major class for all 6.XXX signed 8 bit value
- class [dpt_6_001](#)
data point type 6.001 percent v8
- class [dpt_7](#)
major class for all 7.XXX unsigned signed 16 bit value
- class [dpt_7_001](#)
data point type 7.001 unsigned 16bit pulse counter
- class [dpt_8](#)
major class for all 8.XXX signed 16 bit value
- class [dpt_8_001](#)
data point type 8.001 16bit signed pulse counter
- class [dpt_9](#)
major class for all 9.XXX data point types
- class [dpt_9_001](#)
data point type 9.001 value temp
- class [handle](#)
thread management for connection handling

Typedefs

- typedef [dpt_1_001](#) [dpt_switch](#)
data point type 1.001 is better known as switch
- typedef [dpt_5_001](#) [dpt_scaling](#)
5.001 is better known as scaling value
- typedef [dpt_10_001](#) [dpt_timeofday](#)
10.001 is known as time of day value including hours. minutes, seconds and day of the week
- typedef boost::mpl::vector
< [dpt_1_001](#), [dpt_2_001](#),
[dpt_3_007](#), [dpt_4_001](#),
[dpt_5_001](#), [dpt_6_001](#),
[dpt_7_001](#), [dpt_8_001](#),
[dpt_9_001](#), [dpt_10_001](#) > **dpt_types**

Enumerations

- enum **dpt_index** {
DPT_1_001, **DPT_2_001**, **DPT_3_007**, **DPT_4_001**,
DPT_5_001, **DPT_6_001**, **DPT_7_001**, **DPT_8_001**,
DPT_9_001, **DPT_10_001** }

Functions

- std::ostream & **operator**<< (std::ostream &os, const [dpt_10::time](#) &time)
- template<class dpt >
KNX_IMPORT_EXPORT std::ostream & **operator**<< (std::ostream &os, const [data_point](#)< [dpt](#) > &[data_point](#))

5.1.1 Detailed Description

separated namespace to embed all libknx related classes

Chapter 6

Class Documentation

6.1 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 destructor

Public Attributes

- [std::string local_control_host](#)
name or ip address of the local host (control connection)
- [int local_control_port](#)
communication port for control messages
- [std::string local_data_host](#)
name or ip address of the local host (data connection)
- [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

6.1.1 Detailed Description

This class represents the configuration of the knx connection.

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

- [config.hpp](#)

6.2 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](#) (std::string group, typename data_type::major_type::set_type data)
central function to set any knx group value on the bus
- template<typename data_type >
bool [get](#) (std::string group, typename data_type::major_type::set_type &data)
central function to get any knx group value from the bus
- template<typename data_type >
bool [listen](#) (std::string group, boost::function< void(typename data_type::major_type::set_type &) > callback)
central function to continously get any knx group value from the bus

6.2.1 Detailed Description

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

6.2.2 Member Function Documentation

6.2.2.1 template<typename data_type > bool knx::connection::get (std::string group, typename data_type::major_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

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

Returns

true if read was successful – false otherwise

6.2.2.2 template<typename data_type > bool knx::connection::listen (std::string group, boost::function< void(typename data_type::major_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"
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 );
    boost::this_thread::sleep ( boost::posix_time::seconds ( 2 ) );
    connection.listen<knx::dpt_1_001> ( "0/0/1", my_callback );
    boost::this_thread::sleep( boost::posix_time::seconds ( 20 ) );
    return 0;
}

```

Parameters

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

Returns

true if registering was successful – false otherwise

6.2.2.3 `template<typename data_type > void knx::connection::set (std::string group, typename data_type::major_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.

6.2.2.4 `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.

6.2.2.5 `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

6.3 knx::dpt_6::data Class Reference

Public Member Functions

- **data** (int8_t real_signed_value=0)

Public Attributes

- `int8_t real_signed_value`

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

- `dpt.hpp`

6.4 knx::dpt_4::data Class Reference

Public Member Functions

- **data** (`uint8_t character=0`)

Public Attributes

- `uint8_t character`

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

- `dpt.hpp`

6.5 knx::dpt_9::data Class Reference

Public Member Functions

- **data** (`float float_value=0.f`)

Public Attributes

- `uint16_t value`

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

- `dpt.hpp`

6.6 knx::dpt_5::data Class Reference

Public Member Functions

- **data** (`uint8_t unsigned_value=0`)

Public Attributes

- `uint8_t unsigned_value`

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

- `dpt.hpp`

6.7 knx::data_point< dpt > Class Template Reference

Public Types

- typedef dpt::major_type **major_type**
- typedef major_type::set_type **set_type**
- typedef major_type::data_type **data_type**

Public Member Functions

- **data_point** (set_type value)
- std::string **describe** () const
- set_type **get** () const
- void **set** (set_type value)

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

- data_point.hpp

6.8 knx::dpt Class Reference

Static Public Member Functions

- template<typename data_type >
static std::string **describe** (const typename data_type::major_type::set_type &data)

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

- dpt.hpp

6.9 knx::dpt_1 Class Reference

major class for all 1.XXX data point types

```
#include <dpt.hpp>
```

Public Types

- typedef uint8_t **data_type**
- typedef bool **set_type**

Static Public Member Functions

- static void **set** (set_type value, data_type &data)
- static set_type **get** (const data_type &data)

6.9.1 Detailed Description

major class for all 1.XXX data point types

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

- dpt.hpp

6.10 knx::dpt_10 Class Reference

major class for all 10.XXX data point types

```
#include <dpt.hpp>
```

Classes

- class [time](#)
internal class storing a time value

Public Types

- typedef uint32_t **data_type**
- typedef [time](#) **set_type**

Static Public Member Functions

- static void **set** ([set_type](#) value, data_type &data)
- static [set_type](#) **get** (const data_type &data)

6.10.1 Detailed Description

major class for all 10.XXX data point types

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

- dpt.hpp

6.11 knx::dpt_10_001 Class Reference

class holding a time value

```
#include <dpt.hpp>
```

Public Types

- typedef [dpt_10](#) **major_type**
- typedef TYPENAME
[major_type::set_type](#) **set_type**

Static Public Member Functions

- static std::string **describe** (const [major_type::set_type](#) &data)
- static [major_type::time](#) **now** ()

Static Public Attributes

- static const int **INDEX** = DPT_10_001

6.11.1 Detailed Description

class holding a time value

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

- dpt.hpp

6.12 knx::dpt_1_001 Class Reference

data point type 1.001 simple boolean value

```
#include <dpt.hpp>
```

Public Types

- typedef [dpt_1](#) **major_type**
- typedef TYPENAME
major_type::set_type **set_type**

Static Public Member Functions

- static std::string **describe** (const major_type::set_type &data)

Static Public Attributes

- static const int **INDEX** = DPT_1_001
- static const major_type::set_type **ON**
- static const major_type::set_type **OFF**

6.12.1 Detailed Description

data point type 1.001 simple boolean value

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

- dpt.hpp

6.13 knx::dpt_2 Class Reference

major class for all 2.XXX data point types

```
#include <dpt.hpp>
```

Public Types

- typedef uint8_t **data_type**
- typedef std::pair< bool, bool > **set_type**

Static Public Member Functions

- static void **set** (set_type value, data_type &data)
- static set_type **get** (const data_type &data)

6.13.1 Detailed Description

major class for all 2.XXX data point types

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

- dpt.hpp

6.14 knx::dpt_2_001 Class Reference

data point type 2.001 control value

```
#include <dpt.hpp>
```

Public Types

- typedef [dpt_2](#) **major_type**
- typedef TYPENAME
major_type::set_type **set_type**

Static Public Member Functions

- static std::string **describe** (const major_type::set_type &data)

Static Public Attributes

- static const int **INDEX** = DPT_2_001
- static const major_type::set_type **NO_CONTROL**
- static const major_type::set_type **CONTROL_VALUE_ZERO**
- static const major_type::set_type **CONTROL_VALUE_ONE**

6.14.1 Detailed Description

data point type 2.001 control value

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

- dpt.hpp

6.15 knx::dpt_3 Class Reference

major class for all 3.XXX data point types

```
#include <dpt.hpp>
```

Public Types

- typedef uint8_t **data_type**
- typedef std::pair< bool, uint8_t > **set_type**

Static Public Member Functions

- static void **set** (set_type value, data_type &data)
- static set_type **get** (const data_type &data)

6.15.1 Detailed Description

major class for all 3.XXX data point types

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

- dpt.hpp

6.16 knx::dpt_3_007 Class Reference

data point type 3.007 dimming control

```
#include <dpt.hpp>
```

Public Types

- typedef [dpt_3](#) **major_type**
- typedef TYPENAME
major_type::set_type **set_type**

Static Public Member Functions

- static std::string **describe** (const major_type::set_type &data)

Static Public Attributes

- static const int **INDEX** = DPT_3_007

6.16.1 Detailed Description

data point type 3.007 dimming control

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

- dpt.hpp

6.17 knx::dpt_4 Class Reference

major class for all 4.XXX data point types

```
#include <dpt.hpp>
```

Classes

- class [data](#)

Public Types

- typedef `data` **data_type**
- typedef `uint8_t` **set_type**

Static Public Member Functions

- static void **set** (set_type value, `data_type` &data)
- static set_type **get** (const `data_type` &data)

6.17.1 Detailed Description

major class for all 4.XXX data point types

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

- `dpt.hpp`

6.18 knx::dpt_4_001 Class Reference

data point type 4.001 char ascii

```
#include <dpt.hpp>
```

Public Types

- typedef `dpt_4` **major_type**
- typedef `TYPENAME`
major_type::set_type **set_type**

Static Public Member Functions

- static std::string **describe** (const major_type::set_type &data)

Static Public Attributes

- static const int **INDEX** = DPT_4_001

6.18.1 Detailed Description

data point type 4.001 char ascii

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

- `dpt.hpp`

6.19 knx::dpt_5 Class Reference

major class for all 5.XXX data point types

```
#include <dpt.hpp>
```

Classes

- class [data](#)

Public Types

- typedef [data](#) **data_type**
- typedef uint8_t **set_type**

Static Public Member Functions

- static void **set** (set_type value, [data_type](#) &data)
- static set_type **get** (const [data_type](#) &data)

6.19.1 Detailed Description

major class for all 5.XXX data point types

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

- dpt.hpp

6.20 knx::dpt_5_001 Class Reference

data point type 5.001 unsigned 8 bit

```
#include <dpt.hpp>
```

Public Types

- typedef [dpt_5](#) **major_type**
- typedef TYPENAME
major_type::set_type **set_type**

Static Public Member Functions

- static std::string **describe** (const major_type::set_type &data)

Static Public Attributes

- static const int **INDEX** = DPT_5_001

6.20.1 Detailed Description

data point type 5.001 unsigned 8 bit

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

- dpt.hpp

6.21 knx::dpt_6 Class Reference

major class for all 6.XXX signed 8 bit value

```
#include <dpt.hpp>
```

Classes

- class [data](#)

Public Types

- typedef [data](#) **data_type**
- typedef int8_t **set_type**

Static Public Member Functions

- static void **set** (set_type value, [data_type](#) &data)
- static set_type **get** (const [data_type](#) &data)

6.21.1 Detailed Description

major class for all 6.XXX signed 8 bit value

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

- dpt.hpp

6.22 knx::dpt_6_001 Class Reference

data point type 6.001 percent v8

```
#include <dpt.hpp>
```

Public Types

- typedef [dpt_6](#) **major_type**
- typedef TYPENAME
major_type::set_type **set_type**

Static Public Member Functions

- static std::string **describe** (const major_type::set_type &data)

Static Public Attributes

- static const int **INDEX** = DPT_6_001

6.22.1 Detailed Description

data point type 6.001 percent v8

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

- dpt.hpp

6.23 knx::dpt_7 Class Reference

major class for all 7.XXX unsigned signed 16 bit value

```
#include <dpt.hpp>
```

Public Types

- typedef uint16_t **data_type**
- typedef uint16_t **set_type**

Static Public Member Functions

- static void **set** (set_type value, data_type &data)
- static set_type **get** (const data_type &data)

6.23.1 Detailed Description

major class for all 7.XXX unsigned signed 16 bit value

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

- dpt.hpp

6.24 knx::dpt_7_001 Class Reference

data point type 7.001 unsigned 16bit pulse counter

```
#include <dpt.hpp>
```

Public Types

- typedef [dpt_7](#) **major_type**
- typedef TYPENAME
major_type::set_type **set_type**

Static Public Member Functions

- static std::string **describe** (const major_type::set_type &data)

Static Public Attributes

- static const int **INDEX** = DPT_7_001

6.24.1 Detailed Description

data point type 7.001 unsigned 16bit pulse counter

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

- dpt.hpp

6.25 knx::dpt_8 Class Reference

major class for all 8.XXX signed 16 bit value

```
#include <dpt.hpp>
```

Public Types

- typedef int16_t **data_type**
- typedef int16_t **set_type**

Static Public Member Functions

- static void **set** (set_type value, data_type &data)
- static set_type **get** (const data_type &data)

6.25.1 Detailed Description

major class for all 8.XXX signed 16 bit value

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

- dpt.hpp

6.26 knx::dpt_8_001 Class Reference

data point type 8.001 16bit signed pulse counter

```
#include <dpt.hpp>
```

Public Types

- typedef [dpt_8](#) **major_type**
- typedef TYPENAME
major_type::set_type **set_type**

Static Public Member Functions

- static std::string **describe** (const major_type::set_type &data)

Static Public Attributes

- static const int **INDEX** = DPT_8_001

6.26.1 Detailed Description

data point type 8.001 16bit signed pulse counter

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

- dpt.hpp

6.27 knx::dpt_9 Class Reference

major class for all 9.XXX data point types

```
#include <dpt.hpp>
```

Classes

- class [data](#)

Public Types

- typedef [data](#) **data_type**
- typedef float **set_type**

Static Public Member Functions

- static void **set** (set_type value, [data_type](#) &data)
- static set_type **get** (const [data_type](#) &data)

Static Public Attributes

- static const uint16_t **INVALID_DATA** = 0x7fff
- static const uint16_t **MAX** = 0x7ffe
- static const uint16_t **MIN** = 0xf800
- static const float **MAX_FLOAT** = 670760.96f
- static const float **MIN_FLOAT** = -671088.64f

6.27.1 Detailed Description

major class for all 9.XXX data point types

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

- dpt.hpp

6.28 knx::dpt_9_001 Class Reference

data point type 9.001 value temp

```
#include <dpt.hpp>
```

Public Types

- typedef [dpt_9](#) **major_type**
- typedef TYPENAME
major_type::set_type **set_type**

Static Public Member Functions

- static std::string **describe** (const major_type::set_type &data)

Static Public Attributes

- static const int **INDEX** = DPT_9_001

6.28.1 Detailed Description

data point type 9.001 value temp

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

- [dpt.hpp](#)

6.29 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

6.29.1 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](#)

6.30 knx::dpt_10::time Class Reference

internal class storing a time value

```
#include <dpt.hpp>
```


Public Member Functions

- bool **operator==** (const [time](#) &other)
- [time](#) ()

creates a current time value

Public Attributes

- uint8_t **day**
- uint8_t **hour**
- uint8_t **minutes**
- uint8_t **seconds**

Static Public Attributes

- static const uint8_t **NODAY** = 0
- static const uint8_t **MONDAY** = 1
- static const uint8_t **TUESDAY** = 2
- static const uint8_t **WEDNESDAY** = 3
- static const uint8_t **THURSDAY** = 4
- static const uint8_t **FRIDAY** = 5
- static const uint8_t **SATURDAY** = 6
- static const uint8_t **SUNDAY** = 7

Friends

- std::ostream & **operator<<** (std::ostream &os, const [time](#) &time)

6.30.1 Detailed Description

internal class storing a time value

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

- dpt.hpp

Chapter 7

File Documentation

7.1 knx.hpp File Reference

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

7.1.1 Detailed Description

General include header for libknx.

Index

knx, [9](#)