
smartbus client Python wrapper

Documentation

2.0

lxy@hesong.net

2016 03 16

1		1
1.1	smartbus-client-python	1
1.1.1	1
1.1.2	API	1
1.1.3	1
1.1.4	1
	PYPI	1
	2
1.2	smartbus	2
1.2.1	smartbus package	2
	Submodules	2
	Module contents	12
2		13
Python		15

1.1 smartbus-client-python

smartbus-client-python ()IPSCPython

PythonSmartBus

1.1.1

- SmartBusC
- PythonctypesC/Pythonpypy,ironpython,jythonPythonctypes
- SmartBusC

1.1.2 API

<http://smartbus-client-python.readthedocs.org/>

1.1.3

SmartBus C <https://github.com/Hesong-OpenSource/smartbus-client-sdk>

1.1.4

<https://pypi.python.org/pypi/smartbus-client-python>

PYPI

```
pip install smartbus-client-python
```

:

```
python setup.py
```

```
: Python /DLLSO https://github.com/Hesong-OpenSource/smartbus-client-sdk
```

1.2 smartbus

1.2.1 smartbus package

Submodules

smartbus.errors module

date 2013-6-8

author lxy@hesong.ent

exception smartbus.errors.**AlreadyExistsError**
Exception

exception smartbus.errors.**AlreadyInitializedError**
Exception

exception smartbus.errors.**InvokeFlowIdError**
Exception
ID

exception smartbus.errors.**NotInitializedError**
Exception

exception smartbus.errors.**SmartBusError**(code, message)
Exception

SmartBus

code
SmartBus

message

smartbus.errors.**check**(code, raise_if_err=True)
SmartBus C-API

- **code**(int) –

- **raise_if_err**(bool) –

raise_if_err True None raise_if_err False None

smartbus.head module

```

class smartbus.head.Head(ptr)
    object
    Smartbus
    SMARTBUS_PACKET_HEAD ctypes PacketHeader
    ptr(smartbus._c.mutual.PPacketHeader) -
    cmd
        SmartBus
    cmd_type
        SmartBus
    data_length
    dst_unit_client_id
        ID
    dst_unit_client_type
    dst_unit_id
        ID
    packet_size
    src_unit_client_id
        ID
    src_unit_client_type
    src_unit_id
        ID

```

smartbus.ipc module

IPC API Python

```

class smartbus.ipc.Client(client_id, client_type, user=None, password=None, info=None,
                           lib_path='', event_executor=None)
    smartbus.utils.LoggerMixin

```

IPC

```

: create() get_or_create()

```

- **client_id**(int) -
- **client_type**(int) -
- **user**(str) -
- **password**(str) -
- **info**(str) -
- **lib_path**(str) - SO/DLL *None* : DLL_NAME Python

- **event_executor** (*ThreadPoolExecutor*) – 1

SmartBusError –

Smartbus IPC so/dll Python ,

activate ()

/

: *smartbus /*

client_id

client_type

classmethod create (*args, **kwargs)

Singleton

Client

classmethod destroy ()

dispose ()

classmethod get ()

Singleton

Client

classmethod get_or_create (*args, **kwargs)

Singleton

Client

- *get* ()

- *create* ()

info

launch_flow (*server_unit_id, process_index, project_id, flow_id, mode, timeout, params*)

- **server_unit_id** (*int*) – IPSC *smartbus* ID

- **process_index** (*int*) – IPSC ID IPSC *smartbus* Client ID

- **project_id** (*str*) – ID

- **flow_id** (*str*) – ID

- **mode** (*int*) – 0 1

- **timeout** (*float*) –

- **params** (*list*) – JSON list 32K

invoke_idID

`int`

notify (*server_unit_id*, *process_index*, *project_id*, *title*, *mode*, *expires*, *txt*)

- **server_unit_id** (*int*) – IPSC *smartbus* ID
- **process_index** (*int*) – IPSC IDIPSC *smartbus* client-id
- **project_id** (*str*) – ID
- **title** (*str*) –
- **mode** (*int*) – 0
- **expires** (*float*) –
- **txt** (*str*) – Python2 bytes string, *utf-8*

ID

`int`

Except API

on_connect ()

on_connect_fail (*error_code*)

on_data (*head*, *data*)

- **head** (*Head*) –
- **data** (*bytes*) – bytes

on_disconnect ()

on_flow_ack (*head*, *project_id*, *invoke_id*, *status_code*, *msg*)

- **head** (*Head*) –
- **project_id** (*str*) – ID
- **invoke_id** (*int*) – ID
- **status_code** (*int*) – . 1.
- **msg** (*str*) –

on_flow_error (*head*, *project_id*, *invoke_id*, *error_code*)

- **head** (*Head*) –
- **project_id** (*str*) – ID
- **invoke_id** (*int*) – ID
- **error_code** (*int*) –

on_flow_resp (*head*, *project_id*, *invoke_id*, *params*)

- **head** (*Head*) –
- **project_id** (*str*) – ID
- **invoke_id** (*int*) – ID

- **params** (*list*) – , JSON Array “”

on_flow_timeout (*head, project_id, invoke_id*)

- **head** (*Head*) –
- **project_id** (*str*) – ID
- **invoke_id** (*int*) – ID

on_global_connect_state_changed (*unit_id, client_id, client_type, access_unit_id, status_code, info*)

- **unit_id** (*int*) – SmartbusID
- **client_id** (*int*) – SmartbusID
- **client_type** (*int*) – Smartbus
- **access_unit_id** (*int*) – UnitID
- **status_code** (*int*) – 0 1 2
- **info** (*str*) –

smartbus

password

ping (*dst_unit_id, dst_client_id, dst_client_type, data=None*)

PING

- **dst_unit_id** (*int*) – smartbusID
- **dst_client_id** (*int*) – smartbusID
- **dst_client_type** (*int*) – smartbus
- **data** (*bytes*) –

send_data (*cmd, cmd_type, dst_unit_id, dst_client_id, dst_client_type, data*)

- **cmd** (*int*) –
- **cmd_type** (*int*) –
- **dst_unit_id** (*int*) – ID
- **dst_client_id** (*int*) – ID
- **dst_client_type** (*int*) –
- **data** (*bytes*) – bytes

user

smartbus.net module

NET API Python

```
class smartbus.net.Client(client_id, client_type, master_ip, master_port, slave_ip=None,
                          slave_port=None, user=None, password=None, info=None,
                          event_executor=None)
    smartbus.utils.LoggerMixin
```

NET

```
: create()
```

```
: C-API
```

- **client_id** – client id, ≥ 0 and ≤ 255
- **client_type** – client type
- **master_ip** – IP
- **master_port** –
- **slave_ip** – IP0”“
- **slave_port** – 0xFFFF
- **user** –
- **password** –
- **info** –
- **event_executor** (*ThreadPoolExecutor*) – *I*

```
activate()
/
```

```
: smartbus /
```

client_id

client_type

classmethod create (*args, **kwargs)

classmethod find (client_id, default=None)

- **client_id** – *client_id*
- **default** –

Client

info

classmethod initialize (unit_id, global_connect_callback=None, lib_path='')

- **unit_id** (*int*) – Smartbus ID
- **global_connect_callback** (*callable*) –
- **lib_path** (*str*) – SO/DLL *None* : DLL_NAME Python

```
: unit_id >= 16
```

smartbus

global_connect_callback (*unit_id*, *client_id*, *client_type*, *access_unit_id*, *status_code*, *info*)

- **unit_id** (*int*) – SmartbusID
- **client_id** (*int*) – SmartbusID
- **client_type** (*int*) – Smartbus
- **access_unit_id** (*int*) – UnitID
- **status_code** (*int*) – 0 1 2
- **info** (*str*) –

launch_flow (*server_unit_id*, *process_index*, *project_id*, *flow_id*, *mode*, *timeout*, *params*)

- **server_unit_id** (*int*) – IPSC smartbus ID
- **process_index** (*int*) – IPSC ID IPSC smartbus Client ID
- **project_id** (*str*) – ID
- **flow_id** (*str*) – ID
- **mode** (*int*) – 0 1
- **timeout** (*float*) –
- **params** (*list*) – JSON list 32K

invoke_idID

int

master_ip

master_port

notify (*server_unit_id*, *process_index*, *project_id*, *title*, *mode*, *expires*, *txt*)

- **server_unit_id** (*int*) – IPSC smartbus ID
- **process_index** (*int*) – IPSC IDIPSC smartbus client-id
- **project_id** (*str*) – ID
- **title** (*str*) –
- **mode** (*int*) – 0
- **expires** (*float*) –
- **txt** (*str*) – Python2 bytes string, *utf-8*

ID

int

Except API

on_connect ()

on_connect_fail (*error_code*)

on_data (*head*, *data*)

- **head** (*Head*) –
- **data** (*bytes*) – *bytes*

on_disconnect ()

on_flow_ack (*head, project_id, invoke_id, status_code, msg*)

- **head** (*Head*) –
- **project_id** (*str*) – ID
- **invoke_id** (*int*) – ID
- **status_code** (*int*) – . 1.
- **msg** (*str*) –

on_flow_error (*head, project_id, invoke_id, error_code*)

- **head** (*Head*) –
- **project_id** (*str*) – ID
- **invoke_id** (*int*) – ID
- **error_code** (*int*) –

on_flow_resp (*head, project_id, invoke_id, params*)

- **head** (*Head*) –
- **project_id** (*str*) – ID
- **invoke_id** (*int*) – ID
- **params** (*list*) – , JSON Array “”

on_flow_timeout (*head, project_id, invoke_id*)

- **head** (*Head*) –
- **project_id** (*str*) – ID
- **invoke_id** (*int*) – ID

password

ping (*dst_unit_id, dst_client_id, dst_client_type, data=None*)

PING

- **dst_unit_id** (*int*) – smartbusID
- **dst_client_id** (*int*) – smartbusID
- **dst_client_type** (*int*) – smartbus
- **data** (*bytes*) –

send_data (*cmd, cmd_type, dst_unit_id, dst_client_id, dst_client_type, data*)

- **cmd** (*int*) –

- `cmd_type (int)` –
- `dst_unit_id (int)` – ID
- `dst_client_id (int)` – ID
- `dst_client_type (int)` –
- `data (bytes)` – bytes

slave_ip
slave_port
unit_id
user

smartbus.utils module

Some helper functions

`smartbus.utils.b2s_recode (bs, source_encoding=None, target_encoding=None)`
`bytes str`

- `bs (bytes)` –
- `source_encoding (str)` – bytes *utf-8*
- `target_encoding (str)` – Python2

`str`

`smartbus.utils.s2b_recode (s, source_encoding=None, target_encoding=None)`
`str bytes`

- `s (str)` –
- `source_encoding (str)` – bytes Python2
- `target_encoding (str)` –

`bytes`

`smartbus.utils.to_bytes (s, encoding='utf-8')`
 Convert to *bytes* string.

- `s` – String to convert.
- `encoding (str)` – Encoding codec.

bytes string, it's *bytes* or *str* in Python 2.x, *bytes* in Python 3.x.

`bytes`

- In Python 2, convert *s* to *bytes* if it's *unicode*.

- In Python 2, return original *s* if it's not *unicode*.
- In Python 2, it equals to `to_str()`.
- In Python 3, convert *s* to *bytes* if it's *unicode* or *str*.
- In Python 3, return original *s* if it's neither *unicode* nor *str*.

`smartbus.utils.to_str(s, encoding='utf-8')`

Convert to *str* string.

- **s** – String to convert.
- **encoding** (*str*) – Decoding codec.

str string, it's *bytes* in Python 2.x, *unicode* or *str* in Python 3.x.

str

- In Python 2, convert *s* to *str* if it's *unicode*.
- In Python 2, return original *s* if it's not *unicode*.
- In Python 2, it equals to `to_bytes()`.
- In Python 3, convert *s* to *str* if it's *bytes*.
- In Python 3, return original *s* if it's not *bytes*.
- In Python 3, it equals to `to_unicode()`.

`smartbus.utils.to_unicode(s, encoding='utf-8')`

Convert to *unicode* string.

- **s** – String to convert.
- **encoding** (*str*) – Encoding codec.

unicode string, it's *unicode* in Python 2.x, *str* or *unicode* in Python 3.x.

unicode

- In Python 2, convert *s* to *unicode* if it's *str* or *bytes*.
- In Python 2, return original *s* if it's neither *str* or *bytes*.
- In Python 3, convert *s* to *str* or *unicode* if it's *bytes*.
- In Python 3, return original *s* if it's not *bytes*.
- In Python 3, it equals to `to_str()`.

class `smartbus.utils.LoggerMixin`

object

Mixin Class provide a *logger* property

classmethod `get_logger()`

logger instance.

`logging.Logger`

logger name format is *ModuleName.ClassName*

logger

logger instance.

`logging.Logger`

logger name format is *ModuleName.ClassName*

Module contents

smartbus Python

net ipc Python

date 2013-7-14

author

`smartbus.IpcClient`
Client

`smartbus.NetClient`
Client

-
-
- `genindex`
 - `modindex`
 - `search`

S

`smartbus`, [12](#)
`smartbus.errors`, [2](#)
`smartbus.head`, [3](#)
`smartbus.ipc`, [3](#)
`smartbus.net`, [6](#)
`smartbus.utils`, [10](#)

A

activate() (smartbus.ipc.Client), 4
activate() (smartbus.net.Client), 7
AlreadyExistsError, 2
AlreadyInitializedError, 2

C

client_id (smartbus.ipc.Client), 4
client_id (smartbus.net.Client), 7
client_type (smartbus.ipc.Client), 4
client_type (smartbus.net.Client), 7
cmd (smartbus.head.Head), 3
cmd_type (smartbus.head.Head), 3
code (smartbus.errors.SmartBusError), 2

D

data_length (smartbus.head.Head), 3
dispose() (smartbus.ipc.Client), 4
dst_unit_client_id (smartbus.head.Head), 3
dst_unit_client_type (smartbus.head.Head), 3
dst_unit_id (smartbus.head.Head), 3

I

info (smartbus.ipc.Client), 4
info (smartbus.net.Client), 7
InvokeFlowIdError, 2

L

launch_flow() (smartbus.ipc.Client), 4
launch_flow() (smartbus.net.Client), 8
logger (smartbus.utils.LoggerMixin), 11

M

master_ip (smartbus.net.Client), 8
master_port (smartbus.net.Client), 8
message (smartbus.errors.SmartBusError), 2

N

notify() (smartbus.ipc.Client), 5
notify() (smartbus.net.Client), 8

NotInitializedError, 2

O

on_connect() (smartbus.ipc.Client), 5
on_connect() (smartbus.net.Client), 8
on_connect_fail() (smartbus.ipc.Client), 5
on_connect_fail() (smartbus.net.Client), 8
on_data() (smartbus.ipc.Client), 5
on_data() (smartbus.net.Client), 8
on_disconnect() (smartbus.ipc.Client), 5
on_disconnect() (smartbus.net.Client), 9
on_flow_ack() (smartbus.ipc.Client), 5
on_flow_ack() (smartbus.net.Client), 9
on_flow_error() (smartbus.ipc.Client), 5
on_flow_error() (smartbus.net.Client), 9
on_flow_resp() (smartbus.ipc.Client), 5
on_flow_resp() (smartbus.net.Client), 9
on_flow_timeout() (smartbus.ipc.Client), 6
on_flow_timeout() (smartbus.net.Client), 9
on_global_connect_state_changed() (smartbus.ipc.Client), 6

P

packet_size (smartbus.head.Head), 3
password (smartbus.ipc.Client), 6
password (smartbus.net.Client), 9
ping() (smartbus.ipc.Client), 6
ping() (smartbus.net.Client), 9

S

send_data() (smartbus.ipc.Client), 6
send_data() (smartbus.net.Client), 9
slave_ip (smartbus.net.Client), 10
slave_port (smartbus.net.Client), 10
smartbus (), 12
smartbus.errors (), 2
smartbus.head (), 3
smartbus.ipc (), 3
smartbus.net (), 6
smartbus.utils (), 10

[SmartBusError](#), 2

[src_unit_client_id \(smartbus.head.Head \)](#), 3

[src_unit_client_type \(smartbus.head.Head \)](#), 3

[src_unit_id \(smartbus.head.Head \)](#), 3

U

[unit_id \(smartbus.net.Client \)](#), 10

[user \(smartbus.ipc.Client \)](#), 6

[user \(smartbus.net.Client \)](#), 10