STARS Python Documentation

Release Beta 0.1

KEK Photon Factory

Feb 23, 2017

CONTENTS:

CHAPTER

ONE

STARS PYTHON - STARS.PY

Abstract:

‘stars.py’ is the basic module of STARS written in python. ‘stars.py’ works on Python 2 and 3.

Download stars.py:

1. Goto STARS url: http://stars.kek.jp/
2. Find the section ‘Download and Installation’
3. Click the link of ‘Download Windows version’ or ‘Download Unix version’.
4. Find ‘pythonlib’ then click the link below. You will get the archive file which contains ‘stars.py’.

API reference:

See *stars module*

About the sample programs:

See *STARS Python - the sample programs*

CHAPTER

TWO

STARS PYTHON - SINGLESTARS.PY

Abstract:

'singlestars.py' is the python module derived from the STARS python module 'stars.py'. In addition to the APIs of 'stars.py', the similar APIs supported on the STARS perl module 'stars.pm' are available, for example addcallback(), Mainloop().

In most cases, the STARS client program can be written using only 'stars.py'.

The major difference between 'singlestars.py' and 'stars.py' is the way to run the callback method. 'singlestars.py' runs on the single thread and is able to relate the callback not only to the STARS socket but also to the other sockets. Otherwise 'stars.py' generates the other thread for the callback to the STARS socket.

'singlestars.py' works on Python 2 and 3.

Download singlestars.py:

1. Goto STARS url: http://stars.kek.jp/
2. Find the section 'Download and Installation'
3. Click the link of 'Download Windows version' or 'Download Unix version'.
4. Find 'pythonlib' then click the link below. You will get the archive file which contains 'singlestars.py'.

API reference:

See *singlestars module*

About the sample programs:

See *STARS Python - the sample programs*

CHAPTER

THREE

STARS PYTHON - THE SAMPLE PROGRAMS

Abstract:

The sample programs of the STARS client are available now.

Setup and requirements:

All the sample programs need the connectable STARS server. Before start, please setup the STARS keyfiles to the STARS server. See the file 'README.txt' each of the program.

Download the sample programs of STARS Python:

1. Goto STARS url: http://stars.kek.jp/
2. Find the section 'Download and Installation'
3. Click the link of 'Download Windows version' or 'Download Unix version'.
4. Find 'samples' then click the link below. You will get the archive file which contains the sample programs of STARS Python.

3.1 STARS Python sample programs - stars.py

Abstract:

SampleOfstars.py - The console program using 'stars.py', connecting to STARS server, exchanging the STARS messages, executing the callback method, etc.

3.2 STARS Python sample programs - singlestars.py

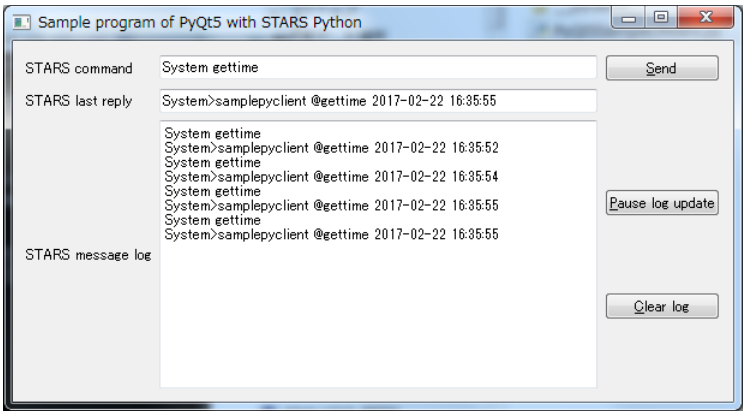
Abstract:

SampleOfsinglestars.py - The console program using 'singlestars.py', connecting to STARS server, exchanging the STARS messages, executing the callback method, etc.

3.3 STARS Python sample programs - using PyQt5 and stars.py

Abstract:

PyQt5SampleOfstars.py - The GUI program of PyQt5 based on Python3 using 'stars.py', connecting to STARS server, exchanging the STARS messages, implementing the callback method on Qthread, etc.



```
System gettime
```

```
System>samplepyclient @gettime 2017-02-22 16:35:55
```

```
System gettime
System>samplepyclient @gettime 2017-02-22 16:35:52
System gettime
System>samplepyclient @gettime 2017-02-22 16:35:54
System gettime
System>samplepyclient @gettime 2017-02-22 16:35:55
System gettime
System>samplepyclient @gettime 2017-02-22 16:35:55
```

CHAPTER FOUR

API REFERENCES

4.1 stars module

STARS Python Interface: basic module of python STARS

History: 0.1 Beta version 2016.05.25 T.Kosuge

0.11 +–bugfix 2016.11.18 Y.Nagatani

0.2 Upgrade 2016.12.20 Y.Nagatani

0.201 +–Change term 'read' to 'receive' 2017.02.17 Y.Nagatani

class `stars.`**`StarsInterface`**(*nodename*, *srvhost*, *keyfile=''*, *srvport=6057*)

Class StarsInterface: Basis of STARS Interface

Parameters: nodename – (string) is so-called 'STARS nodename'.

srvhost – (string) is the IP address or the hostname of STARS server.

keyfile – (optional string) is the filepath of the STARS keyfile. If omitted, nodename + '.key' will be used.

srvport – (optional integer) is the port value of STARS server. If omitted, 6057(=DEFAULT_PORT) will be used.

`DEFAULT_PORT` **= 6057**

`DEFAULT_TIMEOUT` **= 10**

`TCP_BUFFER_SIZE` **= 4096**

`connect`()

connect: Connect to STARS server

Returns: (bool) True if the STARS connection established, otherwise False.

disconnect()
disconnect: Disconnect from STARS server

getdefaultreceivetimeout()
getdefaultreceivetimeout: Return the socket receive timeout value(in seconds).

Returns: (numeric or None) the socket receive timeout value(in seconds).

gethandle()
gethandle: Return STARS socket object.

Returns: (socket object) STARS socket object.

(None) If no socket is assinged for STARS.

getlasterrortext()
getlasterrortext: Return the last error message text.

Returns: (string) the last error message text.

iscallbackrunning()
iscallbackrunning: Return if the callback is running or not. See also start_cb_handler().

Returns: (bool) True if the callback is running, otherwise False.

receive(*timeout=''*, *exceptionret=''*)
receive: Read STARS message from STARS Server.

Parameters: timeout – (optional float,int or None) is the requested timeout(in seconds). If omitted, the value of getdefaultreceivetimeout() will be used. See also getdefaultreceivetimeout(), setdefaultreceivetimeout().

exceptionret – (optional object) is used as the return value when the fatal error deteceted. StarsMessage('') is the default value.

Returns: (StarsMessage object) received data.

(StarsMessage object) if the socket timeout detected, return StarsMessage('').

The value of 'exceptionret' when the fatal error(kind of connection lost, timeout value error...) detected.

send(*arg1*, *arg2=''*, *arg3=''*)
send: Send STARS message to STARS server.

Parameters: arg1 – (string) used as follows.

arg2 – (optional string) used as follows. If omitted, '' will be used.

arg3 – (optional string) used as follows. If omitted, '' will be used.

Use parameters as follows: Send arg1 as STARS message if arg2==''.

Send arg1 + ' ' + arg2 as STARS message if arg2!=''and arg3==''.

Send arg1 + '>' + arg2 + ' ' + arg3 as STARS message if arg2!='' and arg3!=''.

Returns: (bool) True if sended, otherwise False.

setdebug(*b*)

setdebug: debug option.

Parameters: b – (bool) set True to print debug infomation text to stdout.

setdefaultconnectiontimeout(*timeout*)

setdefaultconnectiontimeout: Set the socket connection timeout value referenced in connect().

Parameters:

timeout – (float,int) is used as the socket connection timeout value(in seconds). This value will be used at connect() function.

Returns: (bool) True if the timeout value is valid, otherwise False.

setdefaultreceivetimeout(*timeout*)

setdefaultreceivetimeout: Set the socket receive timeout value(in seconds).

Parameters:

timeout – (float,int or None) is used as the socket receive timeout value(in seconds) This value will be used at receieve() function. See also getdefaultreceivetimeout(), receive() functions.

Returns: (bool) True if the timeout value is valid, otherwise False.

start_cb_handler(*callback*)

Parameters: callback is a python 'callable' function which takes arguments (StarsMessage stmess).

class stars.**StarsMessage**(*message*)

Bases: `str`

Class StarsMessage: STARS Message object.

Properties: allmessage (string) Full text of STARS message.

nodefrom (string) STARS nodename of sender.

nodeto (string) STARS nodename of destination.

message (string) Command and paramters part of STARS message text.

command (string) Command part of STARS message text.

parameters (string) Paramters part of STARS message text.

4.2 singlestars module

STARS Python Interface extended from 'stars.py': Support STARS perl-style functions.

Description: In addition to basic Python STARS features, some APIs are added referring to the STARS perl library 'stars.pm'.

History: 0.1 Beta version 2016.11.09 Yasuko Nagatani

0.2 Support STARS python library ver 0.201 or later 2017.02.17 Yasuko Nagatani

class `singlestars.`**`StarsInterface`**(*nodename*, *srvhost*, *keyfile=''*, *srvport=6057*)

Bases: `stars.StarsInterface`

Class StarsInterface: Derived from basic STARS Interface

Parameters: nodename – (string) is so-called 'STARS nodename'.

srvhost – (string) is the IP address or the hostname of STARS server.

keyfile – (optional string) is the filepath of the STARS keyfile. If omitted, nodename + '.key' will be used.

srvport – (optional integer) is the port value of STARS server. If omitted, 6057(=DEFAULT_PORT) will be used.

`DEFAULT_INTERVALTIME` = **1**

`Mainloop`(*inthandler=None*, *intervaltime=''*)

Mainloop: Start monitoring the readable sockets and executes the assosiated callback function. To terminate this function, use the function terminateMainloop(). See also addcallback() and removecallback().

Attention: Mainloop() runs in the same thread. (Not multi-thread)

Parameters: inthander is a python 'callable' function executed at intervals. If omitted, no function will be executed at intervals. See the 'intervaltime' parameter about the interval time.

intervaltime – (float,int) is the interval time value(in seconds) used for executing the function 'inthandler'. If omitted, the value of the function getintervaltime() will be used. Furthermore the interval time value can be changable by the function setintervaltime() after call Mainloop().

Returns: (bool) False if the error detected before start monitoring the socket, othewise True.

`act`(*sendmesg*, *timeout=''*, *exceptionret=''*)

act: Send STARS message and receive from STARS Server.

Parameters: timeout – (optional float,int) is the requested timeout(in seconds). If omitted, the value of stars.getdefaultreceivetimeout() will be used..

exceptionret – (optional object) is used as the return value when the fatal error deteceted. StarsMessage('') is the default value.

Returns: (StarsMessage object) received data.

(StarsMessage object) if the socket timeout detected, return StarsMessage('').

The value of 'exceptionret' when the fatal error(kind of connection lost, timeout value error...) detected.

`addcallback`(*handler*, *fh=None*, *mode=None*)

addcallback: Assosiates the callback function to the socket readable event. See also removecallback() and Mainloop().

Parameters: hander is a python 'callable' function. See the 'mode' parameter about the arguments of the function.

fh – (optional socket object) is the socket to be monitored. If omitted or None, use the STARS socket. When the socket 'fh' gets readable, the function 'handler' will be called.

mode – (optional) The arguments of the handler depends on the mode value as follows.

If mode value is 'STARS', the handler takes arguments (STARSMessage stmess, (socket object) fh)

If mode value is 'DIRECT', the handler takes arguments (string mess, (socket object) fh)

If mode value is 'DETECT', the handler takes arguments ((socket object) fh)

If omitted or None, same as 'STARS'.

Returns: (bool) True if set properly, otherwise False.

`getintervaltime`()

getintervaltime: Return the current interval time value(in seconds) referenced in the function Mainloop().

Returns: (numeric) the current interval time value(in seconds).

`removecallback`(*fh=None*)

removecallback: Removes from the socket readable monitoring list. See also addcallback() and Mainloop().

Parameters: fh – (optional socket object) is the socket object to be removed from the monitoring list. If omitted or None, use the STARS socket.

Returns: (bool) True if the socket object has removed from the monitoring list. If False returned, it means the socket object 'fh' is not on the monitoring list.

`setintervaltime`(*intervaltime*)

setintervaltime: Change interval time value referenced in the function Mainloop().

Parameters: intervaltime – (float,int) is used as the interval time value(in seconds) at Mainloop(). The initialized value is 1 seconds(=DEFAULT_INTERVALTIME).

Returns: (bool) True if the interval time value is valid, otherwise False.

`terminateMainloop`()

terminateMainloop: Terminate(Exit from) the socket monitoring loop in the function Mainloop().

PYTHON MODULE INDEX

INDEX