

Telit website Python's scripts

1vv0300808 Rev. 0 03-Mar-2009



| | | |
|----------|--|-----------|
| 1 | Introduction | 3 |
| 1.1 | Scope..... | 3 |
| 1.2 | Audience..... | 3 |
| 1.3 | Related Documents | 3 |
| 2 | Software Development Tools..... | 4 |
| 2.1 | Python installation | 4 |
| 3 | Python description | 6 |
| 3.1 | Example Scripts list | 6 |
| 3.2 | Scripts description..... | 7 |
| 3.2.1 | ab_test.py..... | 7 |
| 3.2.2 | ADC_test.py | 7 |
| 3.2.3 | fdi.py | 7 |
| 3.2.4 | FSys_mngtst2.py | 8 |
| 3.2.5 | FTP.py | 8 |
| 3.2.6 | gpiout_tst2.py | 8 |
| 3.2.7 | listenSMS.py | 8 |
| 3.2.8 | PowerSaving.py..... | 9 |
| 3.2.9 | send_email.py..... | 9 |
| 3.2.10 | sendSMS_py | 10 |
| 3.2.11 | SER_MDM_bridge.py | 10 |
| 3.2.12 | SER_send_rcv.py | 10 |
| 3.2.13 | SKT_CL_SR.py | 11 |
| 3.2.14 | sock_MDM2.py | 11 |
| 3.2.15 | watchd_test.py..... | 12 |
| 3.2.16 | SERtest.py | 12 |
| 3.2.17 | testStrArg.py | 12 |
| 3.2.18 | traceback_1.py | 13 |
| 3.2.19 | traceonSER.py..... | 14 |
| 4 | Change Log..... | 15 |



1 Introduction

1.1 Scope

This document gives a first level of support to start working with the Telit Python's scripts and features.

1.2 Audience

This document is intended for customers who design products that integrate Telit modules and are interested in developing software using the Telit Python's engine and development tools.

1.3 Related Documents

The following documents are related to:

- Telit_Easy_Script_in_Python_rx.pdf
- Telit_AT_Commands_Reference_Guide_rx.pdf



2 Software Development Tools

The following tools are suggested to start developing with the Telit Python modules:

- PythonWin package 1.5.2+
- Hyper Terminal

2.1 Python installation

In order to have software that works correctly the system requirement is PC running Windows 2000 or XP. To get *PythonWin package 1.5.2+* with the latest version please contacts Technical Support at the email:

TS-EMEA@telit.com

TS-NORTHAMERICA@telit.com

TS-LATINAMERICA@telit.com

TS-APAC@telit.com

Or by means the form in Telit's website -> technical support -> contact.

Getting user name and password it is possible to download it from Telit's website->download zone:

<http://www.telit.com/en/products/download-zone>

At the moment, the latest version available is *TelitPy1.5.2+_V4.1.exe*.

Note: User & Password are available only for Distributors and direct customers.

To install *Telit Python package* the user needs to execute the exe file *TelitPy1.5.2+_V4.1.exe* and let the installer use the default settings. The installation contains the Python compiler package.

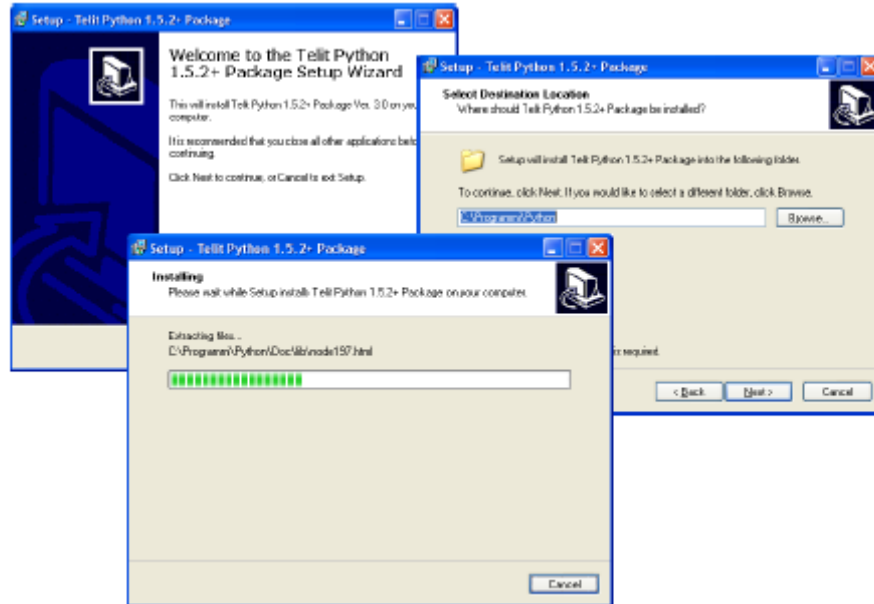
The *Telit Python package* is placed in the folder:

C:\Program Files\Python\



Telit website Python's scripts
1v0300808 Rev. 0 03-Mar-2009

The correct path in the Windows Environmental variables will be set up automatically.



3 Python description

Python scripts are text files stored in NVM inside the [Telit module](#). There's a file system inside the module that allows to write and read files with different names on one single level (no subdirectories are supported).

Attention: it is possible to run only one Python script at the time.

The Python script is executed in a task inside the [Telit module](#) at the lowest priority, making sure this does not interfere with GSM/GPRS normal operations. This allows serial ports, protocol stack etc. to run independently from the Python script.

The Python script interacts with the [Telit module](#) functionality through four build-in interfaces.

3.1 Example Scripts list

- ab_test.py
- ADC_test.py
- fdi.py
- FSys_mngtst2.py
- FTP.py
- gpiout_tst2.py
- listenSMS.py
- PowerSaving.py
- send_email.py
- sendSMS_.py
- SER_MDM_bridge.py
- SER_send_rcv.py
- SKT_CL_SR.py
- sock_MDM2.py
- watchd_test.py
- SERtest.py
- testStrArg.py
- traceback_1.py
- traceonSER.py



3.2 Scripts description

3.2.1 ab_test.py

Script description:

The script tries to open the "test.bin" file. If the file already exists, the program appends some data; otherwise a new file is created with the "wb" parameter.

User inputs: none.

3.2.2 ADC_test.py

Script description:

The script makes a loop with the following operations:

The script measures the ADC value at the ADC_IN1 pin, by issuing AT#ADC and AT#ADC=1,2, then using GPIO.getADC, then prints the result on the debug port, sleeps for 0,5 sec (the permitted Vmax of the input is 2V).

User inputs: none.

3.2.3 fdi.py

Script description:

Unexpected power loss with file left open.

User inputs: none.



3.2.4 FSys_mngtst2.py

Script description:

The script modifies a byte in a file test2.txt of 4000 Bytes stored in NVM, at 5th position.

User inputs: none.

3.2.5 FTP.py

Script description:

Demonstrates FTP operations.

User inputs: none.

3.2.6 gpiout_tst2.py

Script description:

The script sets the GPIO5 as a GPIO output and cycles between 1 and 0 values with a clock period of approximately 2.4 seconds.

User inputs: none.

3.2.7 listenSMS.py

Script description:

The script receives an unsolicited result code if a SMS is received and print it on the debug com port. Then the script collects the content of the SMS arrived and print it on the debug com port.

User inputs: the user should know the mobile number of the SIM card used by the module.



3.2.8 PowerSaving.py

Script description:

The script does a loop with the following operations:

It puts the module in power saving mode for 5 minutes and then wakes it up; the script sends a string to a server and it sleeps for 5 seconds.

User inputs, edit according with local values:

| | |
|---------------------------------|---|
| GPRS_APN = 'xxx.xxxxxx.xx' | <i># GPRS APN, ask your network provider for correct values</i> |
| GPRS_USER = "" | <i># GPRS username</i> |
| GPRS_PASSW = "" | <i># GPRS password</i> |
| SERVER_ADDR = 'xxx.xxx.xxx.xxx' | <i># Target server address</i> |
| SERVER_PORT = xxxx | <i># Target server address port</i> |
| stringtosend = 'testing 123\r' | <i># string to be sent the server</i> |

3.2.9 send_email.py

Script description:

The script sends e-mail using AT#SEMAIL custom command.

User inputs:

| | |
|-----------------------------------|---|
| GPRS_APN = 'xxx.xxxxxx.xx' | <i># GPRS APN, ask your network provider for correct values</i> |
| GPRS_USER = "" | <i># GPRS username</i> |
| GPRS_PASSW = "" | <i># GPRS password</i> |
| SMTP_SERVER = "xxxx.xxxxxxxx.xxx" | <i># SMTP server</i> |
| SMTP_USERID = "xxxx@xxx.xxx" | <i># SMTP username</i> |
| SMTP_PASSW = "xxxxx" | <i># SMTP password</i> |
| FROM_EMAIL_ADDR = "xxxx@xxx.xxx " | <i># SMTP FROM</i> |
| TO_EMAIL_ADDR = "xxxx@xxx.xxx " | <i># SMTP TO</i> |
| SUBJECT = "xxxxxxxxxxxxx" | <i># email subject</i> |
| BODY = "testing 123" | <i># email body</i> |

Note: The AT#EMAILD can be issued instead of AT#SEMAIL. The command sends an e-mail message if GPRS context has already been activated by either AT#EMAILACT=1 or AT#GPRS=1.



Warning: be aware of using an e-mail account that doesn't use the SSL ciphering.

3.2.10 sendSMS_.py

Script description:

The script sends an SMS to a destination number in text mode.

User inputs: recipient number for the SMS.

3.2.11 SER_MDM_bridge.py

Script description:

Idea for a bi-directional SER to MDM bridge, useful in serial cable to wireless applications. It shows only the core of the implementation, other things like module setup and GPRS and serial connections initiation and finalization must be build around. One must play with different settings, flow control, timings etc. to shape on different application details.

User inputs: none.

3.2.12 SER_send_rcv.py

Script description:

The script receives and sends data on SER interface. Type something in terminal after 'Write text:' prompt!

User inputs: none.



3.2.13 SKT_CL_SR.py

Script description:

The script allows selecting a client/server device. The script is composed by the following parts:

- 1) Script configuration (IP address, remote & local ports, APN, PIN, PUK, NET_Operator).
- 2) Check SIM status, insert SIM PIN or PUK when needed, and check for network registration
- 3) Socket configuration and GPRS context activation
- 4) Client or Server selection (external user/application and to send 'c' or 's' character to select one of the 2 options)
- 5a) Start Client mode dialling the remote socket (skt1)
- 5b) Start Server mode configuring firewall and socket listen (skt2)
- 6) GPRS connection; every character sent on the serial port of one module are received on the remote other port in transparent way.
- 7) Suspend the socket with "+++" escape sequence.
- 8) Since MDM-SER bridge is active it is possible to send AT commands (AT#SO=skt to reconnect the socket; AT#SH=skt to close the connection)
- 9) The script is rebooted once DCD low status is detected.

User inputs, edit according with local values:

```

NETWORK    = "vodafone IT" # "I WIND"  "I TIM" # this is the network name to register to in manual mode

PIN         = ""           # SIM card PIN
PUK         = ""           # SIM card PUK
NEW_PIN     = ""           # SIM card PIN
LOCAL_PORT  = xxxxx        # Local server address port
REMOTE_IP   = "xxx.xxx.xxx.xxx" # Remote server address
SUBNET_MASK = "255.0.0.0"
REMOTE_PORT = "xxxx"       # Remote server address port
APN         = "xxx.xxxxxxxx.xx" # GPRS APN, ask your network provider for correct values.

```

3.2.14 sock_MDM2.py

Script description:

The script sends data to a server through a TCP socket on MDM while using MDM2 for other actions.

User inputs, edit according with local values:



Telit website Python's scripts

1v0300808 Rev. 0 03-Mar-2009

```
GPRS_APN = 'xxx.xxxxxxx.xx'  
values  
GPRS_USER = " "  
GPRS_PASSW = ""  
SERVER_ADDR = 'xxx.xxx.xxx.xxx'  
SERVER_PORT = xxxx  
stringtosend = 'testing 123\r'
```

*# GPRS APN, ask your network provider for correct**# GPRS username**# GPRS password**# Target server address**# Target server address port**# string to be sent the server*

3.2.15 watchd_test.py

Script description:

The script is a test for the watchdog function; when the "i" loop counter reaches 2000, the module goes to sleep for 70 seconds starts. After 60 sec. of the 70 sec. the module restarts due to watchdog.

User inputs: none.

3.2.16 SERtest.py

Script description:

The script sends a text string 'TEST' to the serial port (ASC0) every 500ms.

User inputs: none.

3.2.17 testStrArg.py

Script description:

The script is to test the string allocation. It seems that no space is allocated in the names list for the strings delimited by " AND ending with \r. removeCRfrom(string) is a function to use the string without \r.

User inputs: none.



3.2.18 traceback_1.py

Script description:

The script gets exception details and formats them.

User inputs: none.



3.2.19 traceonSER.py

Script description:

The script redirects print output to SER interface; it's useful to debug scripts on GPS modules without the need to rely on CMUX or SSC.

User inputs: none.



4 Change Log

| Revision | Date | Changes |
|----------|----------|-------------|
| ISSUE#0 | 03/03/09 | First ISSUE |

