UTIM Documentation

Connax OY

Feb 14, 2019

User Documentation

The code is open source, released under the terms of Apache License Version 2.0 and available on GitHub.

You can read more on our page *About UTIM and Uhost*. Check now how to *get started* with UTIM and take a look at our *tutorials*, which showcase some demo applications.

The main documentation for the project is organized into different sections:

- *User Documentation*
- *Project Documentation*

CHAPTER 1

About UTIM and Uhost

UTIM is a library for IoT devices that automatically configures devices and securely connecting them.

CHAPTER 2

Getting Started

1. **Install server side (Uhost)**

   The *Uhost installation guide* explains how Uhost can be installed and configured.

1. **Install client side (UTIM)**

   The *UTIM installation guide* explains how UTIM can be installed and configured.

CHAPTER 3

Uhost Installation Guide

3.1 Before the start

First of all, to start you need to have:

1. MySQL database to store UTIMs
2. One of messaging broker for communicating between server and client sides

   2.1 Mosquitto

   2.2 RabbitMQ

3.2 Installation

Use `pip` for python3:

```
pip3 install --extra-index-url https://test.pypi.org/simple/ uhost
```

3.3 Launch

Example of Uhost launcher is *here </user/about>*.

Before you run launcher you need:

1. Set environment variable `UHOST_MASTER_KEY`. Value of this variable is in hex format. For example:

```
UHOST_MASTER_KEY=6b6579
```

2. Edit `config.ini` file (in the same folder), or create config file in the other place and set environment variable `UHOST_CONFIG`. Value of this variable is a absolute path to `config.ini`.

```
; Configuration
; Sections (required):
; * UHOST:
;    * uhostname - name of uhost in hex format (for example, uhostname=74657374 for␣
↪value 'test')
;    * messaging_protocol - MQTT or AMQP
; * MYSQLDB
; Sections (optional, according UHOST.messaging_protocol):
; * MQTT
; * AMQP

[UHOST]
uhostname = 74657374
messaging_protocol = MQTT

[MQTT]
hostname = localhost
username = test
password = test
reconnect_time = 60

[AMQP]
hostname = localhost
username = test
password = test
reconnect_time = 60

[MYSQLDB]
hostname = localhost
username = test
password = test
```

3. After running in output you should see config for Utim like that:

```
#########################################################

Use this configuration to start Utim:

UHOST_NAME=74657374
MASTER_KEY=6b6579
MESSAGING_PROTOCOL=MQTT
MESSAGING_HOSTNAME=localhost
MESSAGING_USERNAME=test
MESSAGING_PASSWORD=test

NOTE: UHOST_NAME and MASTER_KEY are in hex format

#########################################################
```

Note: Do this steps before launch any UTIM instance**

1. Connect to database (from your `config.ini`)
2. Select schema which name is `uhost_{UHOST_NAME}`
3. Add Utim ID in hex format to `device_id` column of `udata` table

CHAPTER 4

UTIM Installation Guide

4.1 Installation

Use `pip` for python3:

```
pip3 install --extra-index-url https://test.pypi.org/simple/ utim
```

4.2 Launch

Example of UTIM launcher is *here </user/about>*.

Before you run launcher you need:

1. Set environment variable `UTIM_MASTER_KEY`. Value of this variable is in hex format. For example:

```
UTIM_MASTER_KEY=6b6579
```

2. Edit `config.ini` file (in the same folder), or create config file in the other place and set environment variable `UTIM_CONFIG`. Value of this variable is a absolute path to `config.ini`.

```
; Configuration
; Sections (required):
; * UTIM:
;   * utimname - name of UTIM in hex format (for example, utimname=74657374 for value
↪'test')
;   * messaging_protocol - MQTT or AMQP
; * MYSQLDB
; Sections (optional, according UTIM.messaging_protocol):
; * MQTT
; * AMQP

[UTIM]
```

(continues on next page)

(continued from previous page)

```
uhostname = 74657374
utimname = 7574696d
messaging_protocol = MQTT

[MQTT]
hostname = localhost
username = test
password = test
reconnect_time = 60

[AMQP]
hostname = localhost
username = test
password = test
reconnect_time = 60
```

CHAPTER 5

UTIM Tutorials

This section contains tutorials showing how to use UTIM and Uhost libraries

5.1 Hello World

This is the simplest application you can create with UTIM and Uhost libraries. It shows how to get UTIM session key.

5.1.1 Python Tutorial - Hello World

This tutorial describes the simplest implementation of UTIM-Uhost usage

Uhost

Code of this example here.

Steps:

1. Import from Uhost library

```
from uhost import uhost
from uhost.utilities.exceptions import UtimConnectionException,␣
↪UtimInitializationError
```

2. Create Uhost object and run it

```
uh1 = uhost.Uhost()
uh1.run()
```

3. Finally, stop Utim before exit

```
uh1.stop()
```

UTIM

Code of this example here.

Steps:

1. Import from UTIM library

```
from utim.connectivity.manager import ConnectivityConnectError
from utim.utim import Utim
from utim.connectivity import DataLinkManager, TopDataType
from utim.connectivity.manager import ConnectivityManager
from utim.utilities.tag import Tag
from utim.utilities.exceptions import UtimConnectionException,␣
↪UtimInitializationError
```

2. Initialize two queues - first is for receiving and second is for transmitting

```
rx_queue = queue.Queue()
tx_queue = queue.Queue()
```

3. Initialize ConnectivityManager - utility to read data from queues. To use queues to send and receive data you should set argument `dl_type=DataLinkManager.TYPE_QUEUE` to ConnectifityManager

```
cm1 = ConnectivityManager()
cm1.connect(dl_type=DataLinkManager.TYPE_QUEUE, rx=tx_queue, tx=rx_queue)
```

3. Create UTIM object and run it

```
concrete_utim = Utim()
concrete_utim.connect(dl_type=DataLinkManager.TYPE_QUEUE, rx=rx_queue, tx=tx_
↪queue)
concrete_utim.run()
```

4. Send data to start communication:

```
data1 = [TopDataType.DEVICE, Tag.INBOUND.NETWORK_READY]
cm1.send(data1)
```

5. Wait for session key and stop it when the key is received

```
while True:
    data = cm1.receive()
    if data:
        session_key = data[1]
        concrete_utim.stop()
        break
```

6. Finally, stop Utim (if not stopped) and ConnectivityManager before exit

```
concrete_utim.stop()

cm1.stop()
```

5.2 SSH password authentication

This example shows how the key can be applied to connect via ssh

5.2.1 Python Tutorial - SSH password authentication

This tutorial describes the how you can use generated UTIM session key to connect to server via ssh

Uhost

Run Uhost as described in Python Tutorial - Hello World.

Creating users

Code of this example here.

This script creates new user for linux machine using UTIM ID's as username and UTIM's session key as password

Step:

1. Connect to database
2. Select IDs and keys of UTIMs
3. Create or delete (if exists) and create new user.

UTIM

Code of this example here.

Steps:

1. Import from UTIM library

```
from utim.connectivity.manager import ConnectivityConnectError
from utim.utim import Utim
from utim.connectivity import DataLinkManager, TopDataType
from utim.connectivity.manager import ConnectivityManager
from utim.utilities.tag import Tag
from utim.utilities.exceptions import UtimConnectionException,␣
↪UtimInitializationError
```

2. Initialize two queues - first is for receiving and second is for transmitting

```
rx_queue = queue.Queue()
tx_queue = queue.Queue()
```

3. Initialize ConnectivityManager - utility to read data from queues. To use queues to send and receive data you should set argument `dl_type=DataLinkManager.TYPE_QUEUE` to ConnectifityManager

```
cm1 = ConnectivityManager()
cm1.connect(dl_type=DataLinkManager.TYPE_QUEUE, rx=tx_queue, tx=rx_queue)
```

3. Create UTIM object and run it

```
concrete_utim = Utim()
concrete_utim.connect(dl_type=DataLinkManager.TYPE_QUEUE, rx=rx_queue, tx=tx_
↪queue)
concrete_utim.run()
```

4. Send data to start communication:

```
data1 = [TopDataType.DEVICE, Tag.INBOUND.NETWORK_READY]
cm1.send(data1)
```

5. Wait for session key and stop it when the key is received

```
while True:
    data = cm1.receive()
    if data:
        session_key = data[1]
        concrete_utim.stop()
        break
```

6. Use `paramiko` library to connect via ssh, execute command and print result of command executing:

```
ssh = paramiko.SSHClient()
ssh.load_system_host_keys()
ssh.connect(_SERVER, username=_UTIM_NAME.hex().upper(), password=session_key.
↪hex())
ssh_stdin, ssh_stdout, ssh_stderr = ssh.exec_command('ifconfig')
for line in iter(ssh_stdout.readline, ""):
    print(line, end="")
ssh.close()
```

7. Finally, stop Utim (if not stopped) and ConnectivityManager before exit

```
concrete_utim.stop()

cm1.stop()
```

CHAPTER 6

uTeam - UTIM team

The UTIM development team is formed by Connax.

CHAPTER 7

Glossary

This is a glossary of terms.

UTIM Universal Thing Identity Module (UTIM) for IoT devices

Uhost Universal Host for UTIM working with any remote computer

CHAPTER 8

Indices and tables

- genindex
- search

Index

U