ublox Documentation

Release 0.1.0

Henrik Wahlgren @ Palmlund Wahlgren Innovative Technology AB

Nov 22, 2018

Contents:

CHAPTER 1

Installation

ublox require Python 3.6 or higher.

Install with pip:

```
pip install ublox
```

CHAPTER 2

Using the library

2.1 Supported Modules

We have tested the library agains the following module:

- SARA N211
- SARA R410
- SARA R412

Since Ublox has the same AT commands for alot of their modules it might work on other modules as well.

2.2 Creating a module

You need to connect the module over a serial line and declare the module in the program using the serial port.

For development we have used Sodaq boards where an Arduino is controlling the module. By using a small passthrough program we can connect the serial interface from the board to our computer via the arduino. The Arduino is also responsible for setting the power pins to the module so it starts.

```
# Creating a module
module = SaraR4Module(serial_port='/dev/tty.usbmodem14111')
```

2.3 Configure module

We are running tests in Sweden and have defined settings to configure the module to work for operators in Sweden. You can set the module up using the .setup() method or use lower API method ._at_action() to set the module up exactly as you want.

```
module.setup()
# or
module._at_action('AT+CFUN=1')
```

There are also methods that wrap common at actions that are documented on the module class.

2.4 Connect the module to a network

When connecting it is important to know if you operator is running in home network or in roaming network. This will give different response when waiting for the connection signal.

You use the numerical ID of you operator (MNO) to connect. In Sweden Telia is 24001 and Tre is 24002.

```
module.connect(operator=24001, roaming=True)
```

2.5 Sending Data

When you want to send data you need to create a socket on the module. This is represented in the library as an object that looks and behaves like a normal python socket.

```
sock = module.create_socket(socket_type='UDP', port=1337)
sock.sendto(b'mytestdata', ('195.34.89.241', 7))
# when you don't need the socket you can close it.
sock.close()
```

2.6 Monitor radio environment

On of the things we are using the library for is to monitor coverage and the radio environment of NB-IoT. You can use the .update_radio_statistics() to get new values from the module and access them on the module object.

```
module.update_radio_statistics()
print(module.radio_rsrq)
print(module.radio_rsrp)
```

Note: Different modules support different statistics values.

CHAPTER 3

Indices and tables

- genindex
- modindex
- search