

LESSON 5: GETTING CONNECTED WITH SIGFOX

by Pycom



LESSON OBJECTIVES

- Introduction to Sigfox
- Create a Sigfox account and register the device
- Send Sigfox messages
- Troubleshooting

<https://docs.pycom.io/firmwareapi/pycom/network/sigfox/>

SIGFOX - INTRODUCTION

- Sigfox is a Low Power Wide Area Network that enables remote devices to connect using ultra-narrow band (UNB technology)
- It uses the Industrial, Scientific and Medical radio band
- It uses very narrow bits of spectrum, which minimises the effect of noise

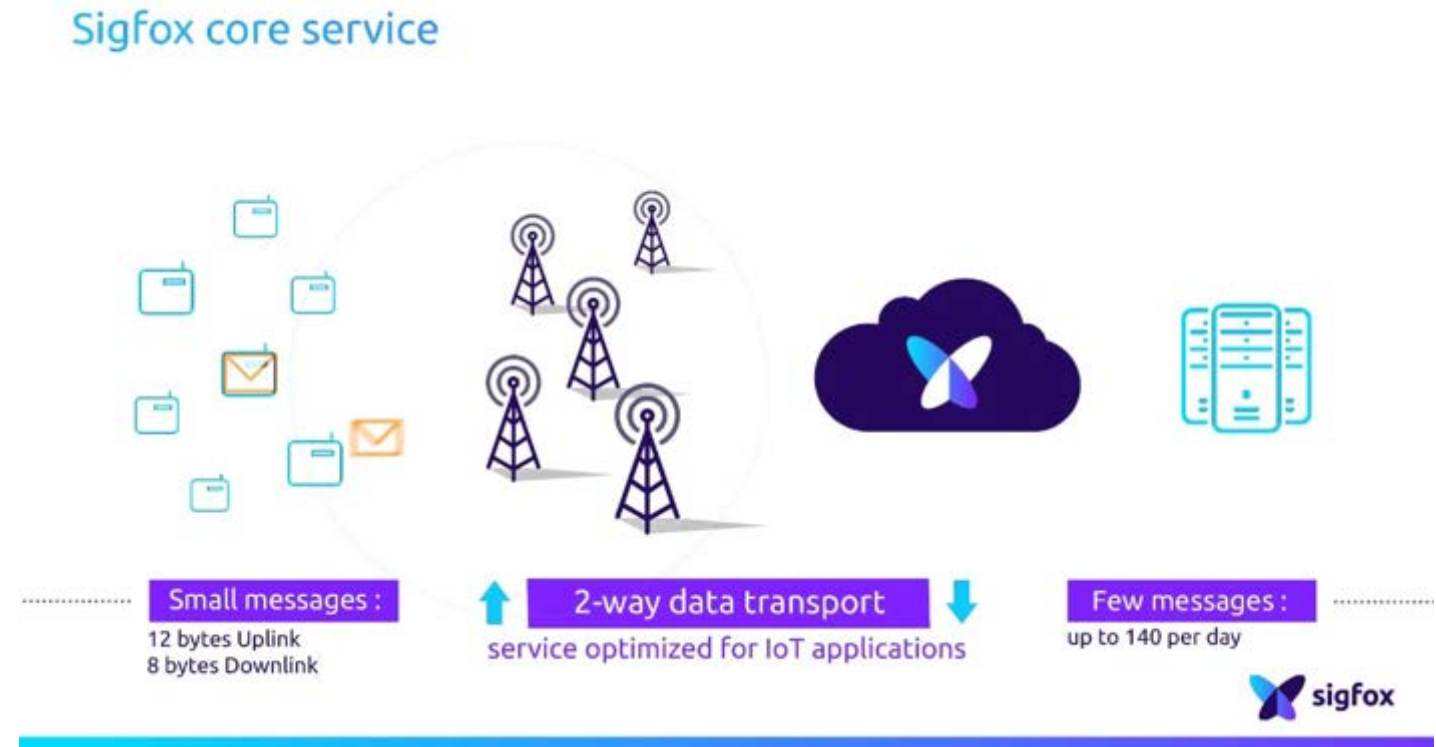


(Image source :[picto_sigfox-EN.jpg](#))

<https://docs.pycom.io/firmwareapi/pycom/network/sigfox/>

SIGFOX - INTRODUCTION

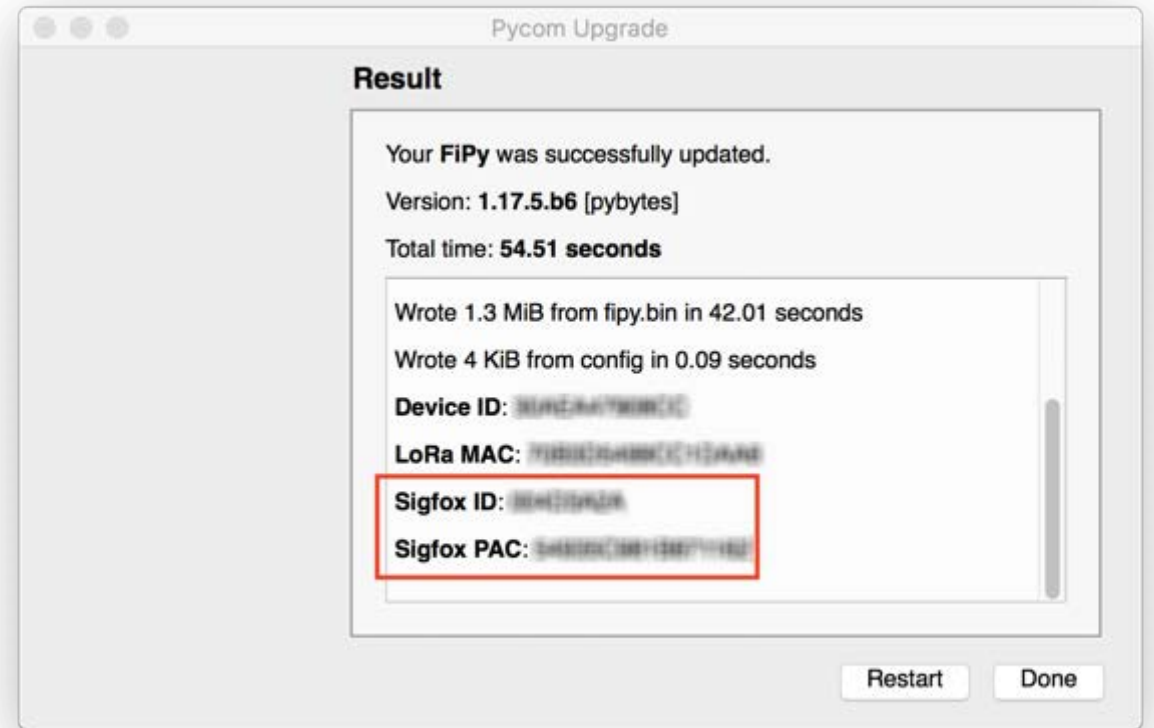
- A Sigfox message is limited to **12-bytes!**
- Also, the accounts are limited to 140 messages per **day**
- So think wisely about what you need to send!
- But rest assured it will be very power efficient



<https://www.elektormagazine.com/news/review-wireless-modules-from-pycom/15521>

FIRMWARE UPDATE AND SIGFOX ID

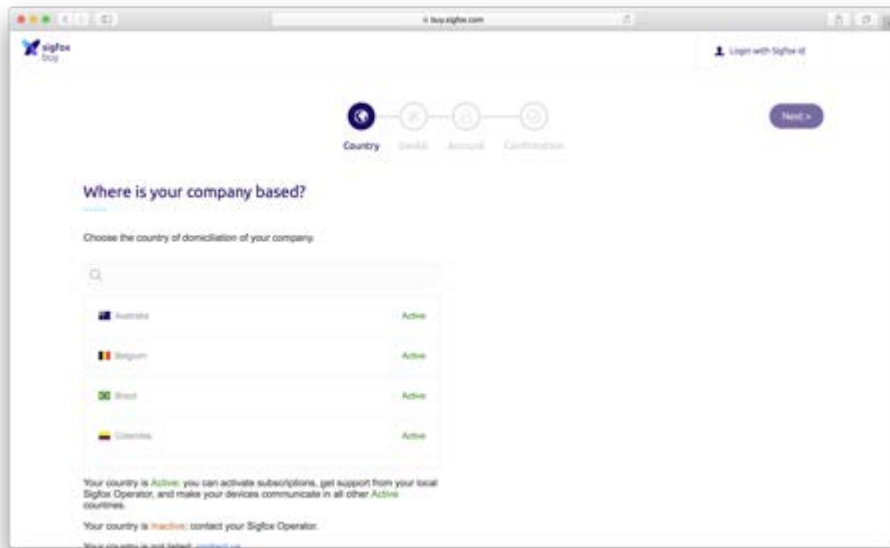
1. Use the Firmware Updater
2. Update your device to the latest firmware
3. At the end of the update
 - a **Sigfox ID** and
 - a **Sigfox PAC** are assigned to your device
4. Copy the **ID** and the **PAC** from the last screen of the Firmware Updater



<https://docs.pycom.io/gettingstarted/registration/sigfox/>

CREATE AN ACCOUNT ON SIGFOX BACKEND

1. Register to the Sigfox Backend - <https://buy.sigfox.com/activate>



2. Enter your country

Where is your company based?

Choose the country of domiciliation of your company.



Your country is **Active**: you can activate subscriptions, get support from your local Sigfox Operator, and make your devices communicate in all other **Active** countries.

Your country is **Inactive**: contact your Sigfox Operator.

WND - UK
United Kingdom
WNDUK is the Sigfox Network Operator for the UK

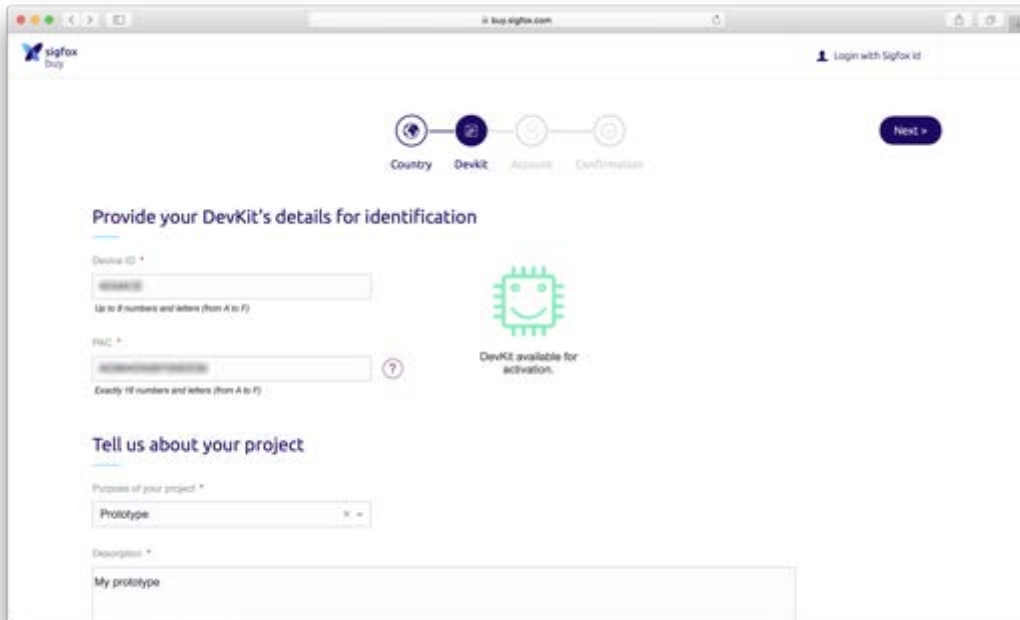
WND
United Kingdom

WND - UK main office
Unit E1, Regent Park, Summerleys Road
HP27 9LE Princes Risborough, Buckinghamshire
<http://www.wnduk.com/>

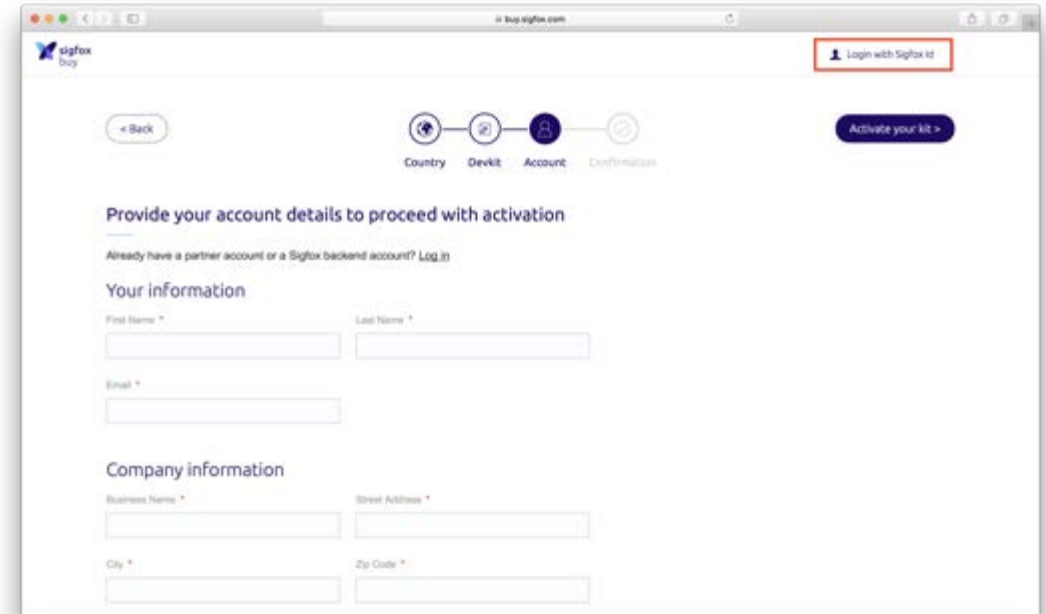
CREATE AN ACCOUNT ON SIGFOX BACKEND

4. The correct **Sigfox ID** and **Sigfox PAC** should prompt a green microchip

• 5. Provide the required information



The screenshot shows the 'Devkit' step of the registration process. The progress bar at the top indicates that the 'Devkit' step is active. The main heading is 'Provide your DevKit's details for identification'. There are two input fields: 'Device ID' and 'PAC'. A green microchip icon is displayed next to the 'PAC' field, with the text 'DevKit available for activation.' below it. The 'PAC' field has a help icon and the text 'Exactly 16 numbers and letters (from A to F)'. Below this, there is a section 'Tell us about your project' with a dropdown menu for 'Purpose of your project' (set to 'Prototype') and a text area for 'Description'.

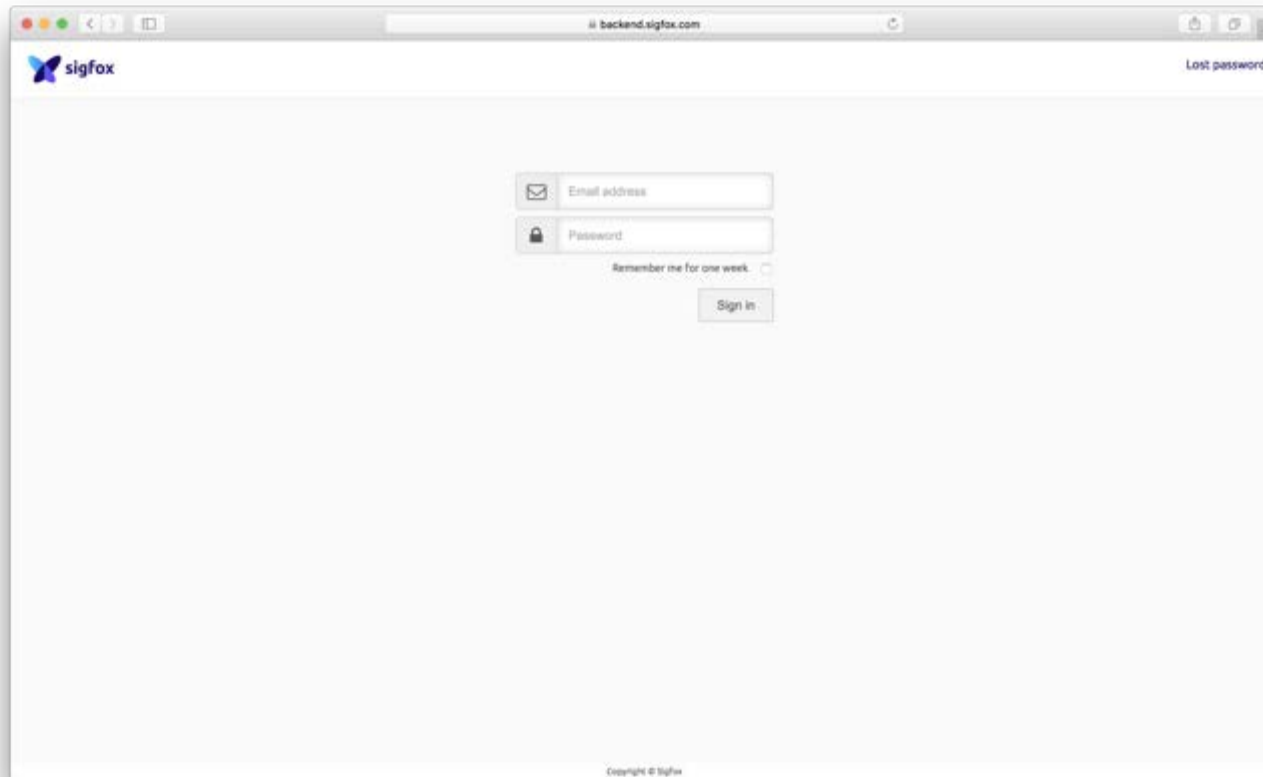


The screenshot shows the 'Account' step of the registration process. The progress bar at the top indicates that the 'Account' step is active. The main heading is 'Provide your account details to proceed with activation'. There is a 'Login with Sigfox id' button in the top right corner. Below the heading, there is a 'Your information' section with input fields for 'First Name', 'Last Name', and 'Email'. There is also a 'Company information' section with input fields for 'Business Name', 'Street Address', 'City', and 'Zip Code'. A 'Back' button is visible on the left side.

<https://docs.pycom.io/gettingstarted/registration/sigfox/>

CREATE AN ACCOUNT ON SIGFOX BACKEND

6. After registration, you will receive a confirmation email with a password to Sigfox's backend (<https://backend.sigfox.com/auth/login>)



IF YOU ALREADY HAVE A SIGFOX ACCOUNT...

- Make sure that you use the same login details
- That way, all of your devices will be added to the same Sigfox Account

SENDING A MESSAGE

- Use this code to send your first message
- set the "Region" depending on your geographic location:
 - RCZ1 for Europe
 - RCZ2 for USA, Mexico & Brazil
 - RCZ3 for Japan
 - RCZ4 for Australia, New Zealand, Singapore, Taiwan, Hong Kong, Colombia & Argentina

```
from network import Sigfox
import socket

# init Sigfox for RCZ1 (Europe)
sigfox = Sigfox(mode=Sigfox.SIGFOX, rcz=Sigfox.RCZ1)

# create a Sigfox socket
s = socket.socket(socket.AF_SIGFOX, socket.SOCK_RAW)

# make the socket blocking
s.setblocking(True)

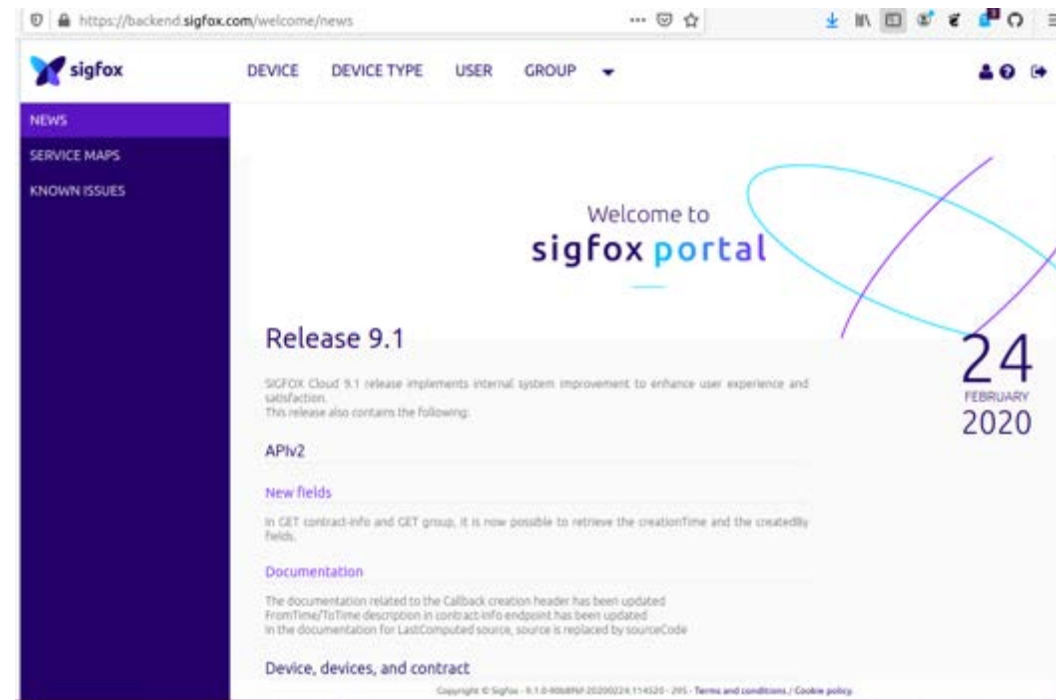
# configure it as uplink only
s.setsockopt(socket.SOL_SIGFOX, socket.SO_RX, False)

# send some bytes
s.send(bytes([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]))
```

<https://docs.pycom.io/firmwareapi/pycom/network/sigfox/>

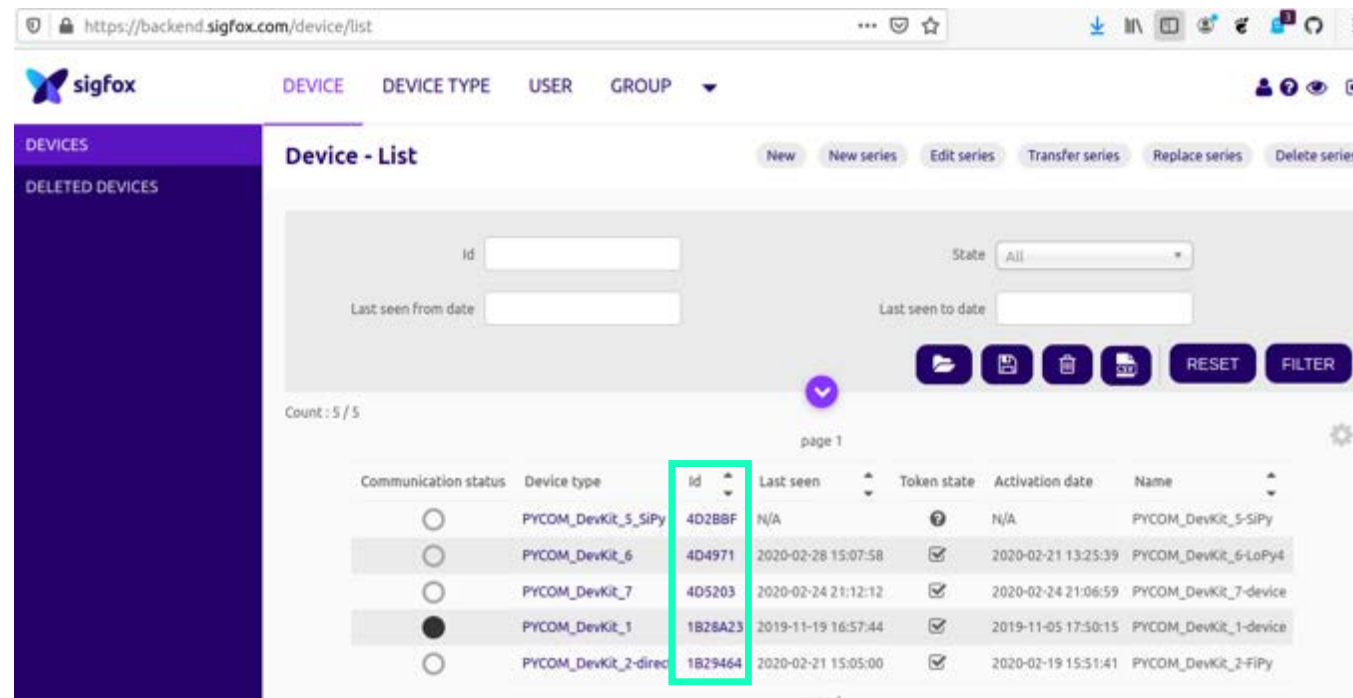
VERIFYING THE SEND

1. Once you log in, you should see a page similar to the one below



VERIFYING THE SEND

2. Click on the 'Device' tab in the Sigfox backend and click on the hexadecimal device ID



The screenshot displays the Sigfox backend interface for managing devices. The page title is 'Device - List'. The left sidebar shows 'DEVICES' and 'DELETED DEVICES'. The main content area includes search filters for 'Id', 'State', 'Last seen from date', and 'Last seen to date'. Below the filters, there are buttons for 'New', 'New series', 'Edit series', 'Transfer series', 'Replace series', and 'Delete series'. A table of devices is shown with the following data:

| Communication status | Device type | Id | Last seen | Token state | Activation date | Name |
|----------------------------------|-----------------------|---------|---------------------|-------------------------------------|---------------------|-----------------------|
| <input type="radio"/> | PYCOM_DevKit_5_SIPy | 4D28BF | N/A | <input type="checkbox"/> | N/A | PYCOM_DevKit_5-SIPy |
| <input type="radio"/> | PYCOM_DevKit_6 | 4D4971 | 2020-02-28 15:07:58 | <input checked="" type="checkbox"/> | 2020-02-21 13:25:39 | PYCOM_DevKit_6-LoPy4 |
| <input type="radio"/> | PYCOM_DevKit_7 | 4D5203 | 2020-02-24 21:12:12 | <input checked="" type="checkbox"/> | 2020-02-24 21:06:59 | PYCOM_DevKit_7-device |
| <input checked="" type="radio"/> | PYCOM_DevKit_1 | 1B28A23 | 2019-11-19 16:57:44 | <input checked="" type="checkbox"/> | 2019-11-05 17:50:15 | PYCOM_DevKit_1-device |
| <input type="radio"/> | PYCOM_DevKit_2-direct | 1B29464 | 2020-02-21 15:05:00 | <input checked="" type="checkbox"/> | 2020-02-19 15:51:41 | PYCOM_DevKit_2-FIPy |

VERIFYING THE SEND

3. Click on *Messages* in the left menu

The screenshot shows the Sigfox web interface. On the left, a dark blue sidebar contains a menu with the following items: INFORMATION, LOCATION, MESSAGES (highlighted with a red circle), EVENTS, STATISTICS, and EVENT CONFIGURATION. The main content area is titled 'Device 1D24D5 - Information' and displays various device details. At the top right of the main area, there are buttons for 'Suspend' and 'Disengage sequence num'. The device information includes:

- Name: LoPy4_DevKit_1-device
- Protocol: V1
- Activable state: ⓘ
- Sequence number: 28 (2019-02-15 13:52:34)
- Trash sequence number: N/A (N/A)
- Last seen: 2019-02-15 13:52:34
- PAC: F31CEA5E1B970F52
- Product certificate: P_00C1_1366_01
- Latitude: 0.000 (degrees)
- Longitude: 0.000 (degrees)
- Device type: Arduino_DevKit_1
- State: OK
- Link Quality Indicator: ⓘ
- Communication status:
- Contract: arduino-_50ca_8220
- Activation date: 2019-02-06 14:25:52
- Token validity: 2020-02-06
- Unsubscription date: N/A ⓘ
- Subscription automatic renewal status: Not allowed ⚠
- Subscription automatic renewal: ⓘ
- Creation date: 2019-02-06 14:13:25

VERIFYING THE SEND

4. Now you should see your messages (hex numbers) together with timestamps and icons indicating the transmission status

The screenshot displays the Sigfox web interface for a specific device. The sidebar on the left contains navigation links: INFORMATION, LOCATION, MESSAGES (highlighted), EVENTS, STATISTICS, and EVENT CONFIGURATION. The main header shows the Sigfox logo and navigation tabs: DEVICE, DEVICE TYPE, USER, and GROUP. The page title is 'Device 1D24D5 - Messages'. Below the title is a filter section with 'From date' and 'To date' input fields, a 'RESET' button, a 'FILTER' button, and a CSV export icon. The table below shows the message data:

| Time | Data / Decoding | LQI | Callbacks | Location |
|---------------------|-----------------|-----|-----------|----------|
| 2019-02-15 13:51:58 | 686968696869 | | | |
| 2019-02-15 13:47:07 | 686968696869 | | | |
| 2019-02-15 13:43:59 | 6869696869 | | | |
| 2019-02-15 13:42:05 | 68696869 | | | |

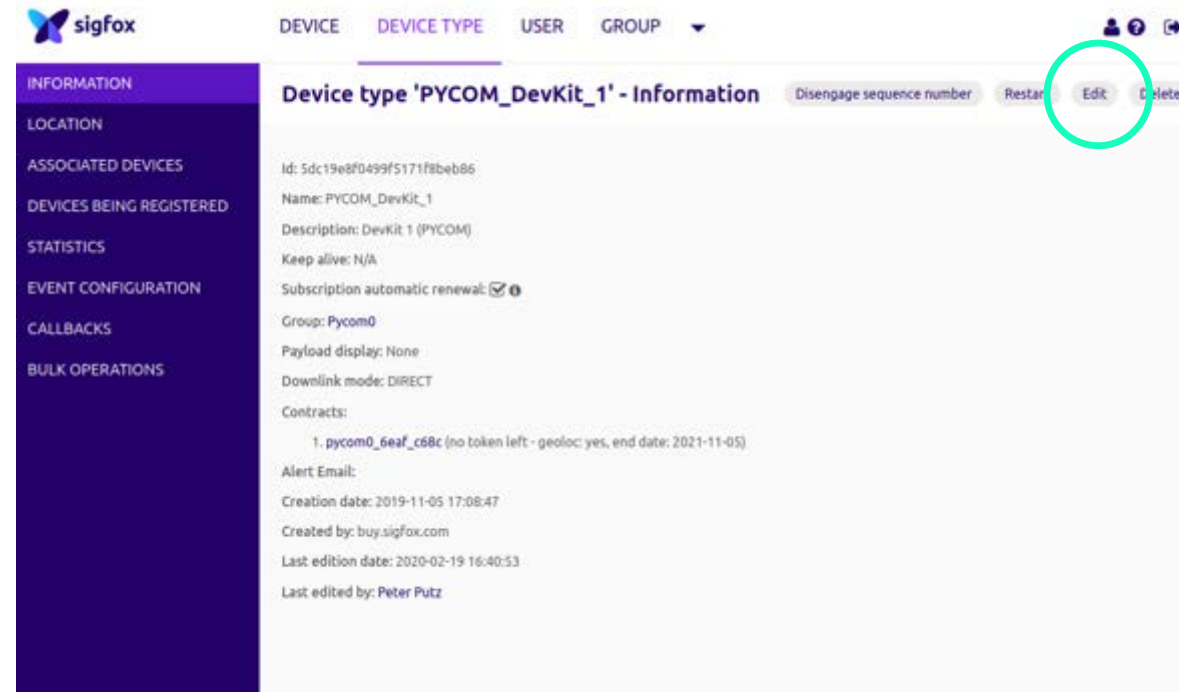
RECEIVING A MESSAGE

- So far, we've **sent** a message from the Pycom device to Sigfox.
- Now, we will **receive** a message from Sigfox.
- We will demo the downlink, by configuring the Sigfox backend to always send one particular message in response

<https://docs.pycom.io/firmwareapi/pycom/network/sigfox/>

CONFIGURE THE DOWNLINK

1. Go to your Device Type and click on Edit



CONFIGURE THE DOWNLINK

2. Change the *Downlink mode* to *DIRECT*

3. Configure the *Downlink data* in hex, e.g.
`{tapId}0000{rssi}`

DEVICE DEVICE TYPE USER GROUP

Device type PYCOM_DevKit_1 - Edition

Contracts:

If we fail to call one of your callbacks, an email will be sent to the address below so that you can take action to fix the problem.

Alert email:

Downlink data

Downlink mode: **DIRECT** For more details on Downlink modes, please refer to [documentation](#).

Expression must either include hexadecimal encoded bytes (ex: deadbeefcafebabe) or the following variables: - (time) 4 bytes - (tapId) 4 bytes - (rssi) 2 bytes - (roaming) 1 byte

Downlink data in hex:

Payload display

Select below the most suitable parsing mode for the display of your payloads in the backend (mostly appropriate for debugging and development)

Payload parsing: **Regular (raw payload)**

Ok Cancel

Copyright © Sigfox - 9.1.0-90b8f6f-20200224.114520 - 295 - Terms and conditions / Cookie policy

RECEIVING A MESSAGE

4. Send a Sigfox message again, but this time:

- Include a downlink request, by using True as the third setsockopt() parameter
- Add a s.recv()

```
# init Sigfox for RCZ1 (Europe)
sigfox = Sigfox(mode=Sigfox.SIGFOX, rcz=Sigfox.RCZ1)

# create a Sigfox socket
s = socket.socket(socket.AF_SIGFOX, socket.SOCK_RAW)

# make the socket blocking
s.setblocking(True)

# configure it as DOWNLINK specified by 'True'
s.setsockopt(socket.SOL_SIGFOX, socket.SO_RX, True)






















# send some bytes and request DOWNLINK
s.send(bytes([1, 2, 3]))

# await DOWNLINK message
s.recv(32)
```

<https://docs.pycom.io/firmwareapi/pycom/network/sigfox/>

VERIFYING THE RECEPTION

- On the Sigfox backend you will see the downlink status as an icon and afterwards,
- You will receive the message on your Pycom device

| | | | | | |
|---------------------|------|---|---|---|---|
| 2020-02-21 13:35:12 | 4257 |  |  |  |  |
| 2020-02-21 13:15:16 | 4259 |  |  |  |  |
| 2020-02-21 13:08:54 | 42b2 |  |  |  |  |
| 2020-02-21 13:07:52 | 0965 |  |  | |  |
| 2020-02-21 13:07:20 | 09cb |  |  | |  |
| 2020-02-21 13:06:54 | 0996 |  |  | |  |

Downlink status - Acked



Congrats, you've sent uplink and downlink messages with Sigfox!

<https://docs.pycom.io/firmwareapi/pycom/network/sigfox/>

(A LITTLE NOTE BEFORE YOU GO...)

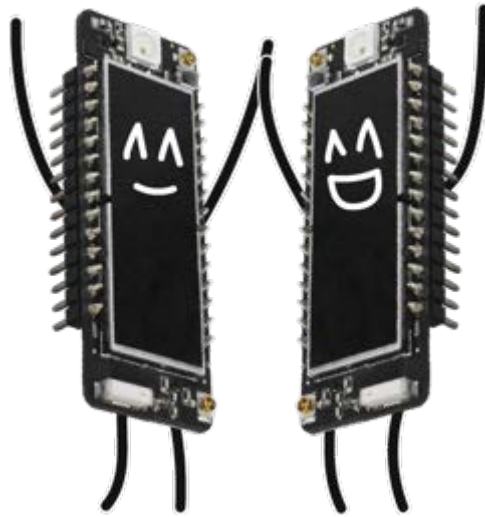
For the next lesson, you'll be connecting to a cellular network. You'll need to get a SIM card from a local provider.

Please note that this might differ from a standard SIM as our devices do not support standard LTE

We recommend contacting your local cellular providers to check out what their plans are surrounding LTE CAT-M1 and NB-IoT.

Click here to find a map of deployed networks and open labs [here](#).

HAVE A PLAY AROUND SEE WHAT HAPPENS!



Problems? Click [here](#)

TROUBLESHOOTING



Problems? Click [here](#)

DISENGAGE THE SEQUENCE NUMBER

- If you are experiencing connectivity issues, this could be due to the sequence number being out of sync
- If there is a large difference between the sequence number sent by the device and the one expected by the backend, the message is dropped by the network

<https://docs.pycom.io/tutorials/sigfox/>

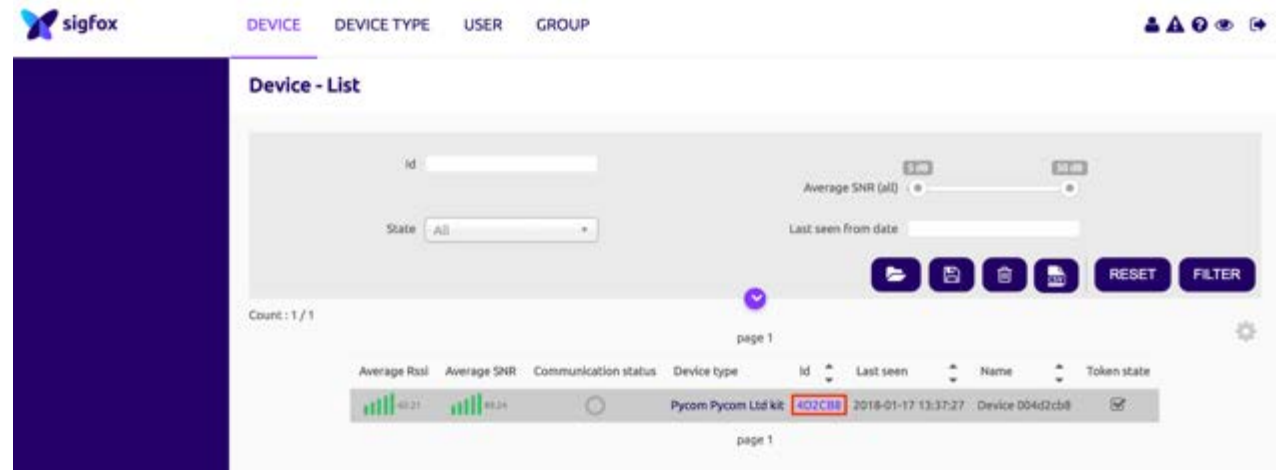
DISENGAGE THE SEQUENCE NUMBER

- You can use the **Disengage Sequence Number** button on *Device Information*
- Alternatively, on the *Device Type* information page it is possible to reset the number expected by the backend

<https://docs.pycom.io/tutorials/sigfox/>

DISENGAGE THE SEQUENCE NUMBER

1. Log into the Sigfox Backend
2. Navigate to **Device**
3. Click on the **Sigfox ID** of the affected device



<https://docs.pycom.io/tutorials/sigfox/>

DISENGAGE THE SEQUENCE NUMBER

4. The **Information** page should now be accessible

5. The 'Device Type' should be followed by a link – follow it

The screenshot shows the Sigfox web interface. At the top left is the Sigfox logo. To the right are navigation tabs: DEVICE, DEVICE TYPE, USER, and GROUP. A dark blue sidebar on the left contains the following menu items: INFORMATION, LOCATION, MESSAGES, EVENTS, STATISTICS, and EVENT CONFIGURATION. The main content area is titled 'Device 4D2CB8 - Information' and displays the following details:

- Name: Device 004d2cb8
- Protocol: V1
- Last seen: 2018-01-17 13:37:27
- Product certificate:
- Latitude: 0.000 (degrees)
- Longitude: 0.000 (degrees)
- Device type: **Pycom Pycom Ltd kit** (highlighted with a red box)
- Average SNR 📶: 89.24 dB
- Average RSSI 📶: -62.21 dBm
- Communication status:

<https://docs.pycom.io/tutorials/sigfox/>

DISENGAGE THE SEQUENCE NUMBER

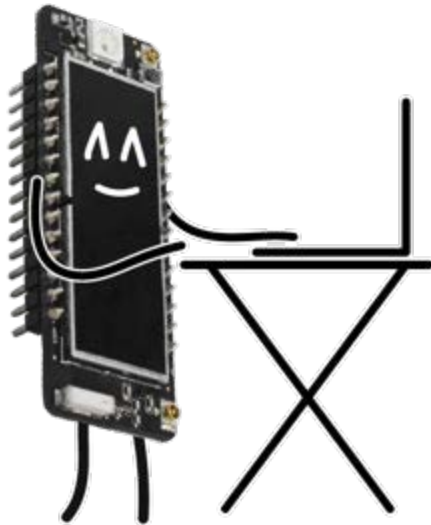
6. On the page, click on **Disengage Sequence Number** (located in the upper right corner)



<https://docs.pycom.io/tutorials/sigfox/>

ID AND PAC AGAIN

- If you want to get your **Sigfox ID** and **Sigfox PAC** after the firmware update, run these commands via the REPL



```
from network import Sigfox
import binascii

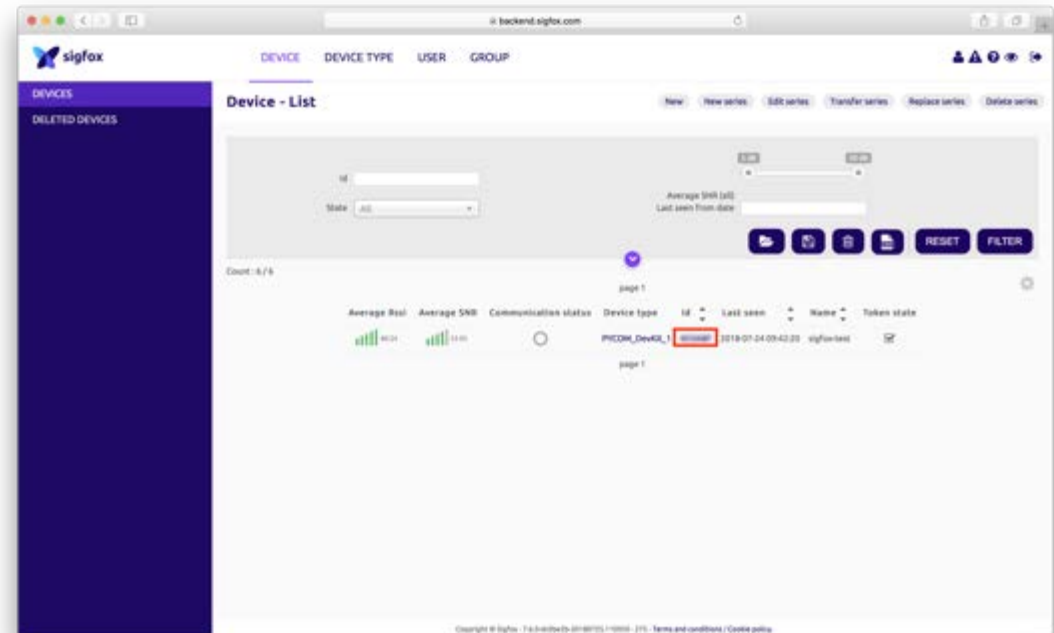
# initialise Sigfox for RCZ1 (You may need a different RCZ Region)
sigfox = Sigfox(mode=Sigfox.SIGFOX, rcz=Sigfox.RCZ1)

# print Sigfox Device ID
print(binascii.hexlify(sigfox.id()))

# print Sigfox PAC number
print(binascii.hexlify(sigfox.pac()))
```

TRANSFERRING A DEVICE TO ANOTHER SIGFOX ACCOUNT

1. Navigate to your device on Sigfox backend
2. Click on **Devices**
3. Click on the **Sigfox ID** of the device you want to transfer

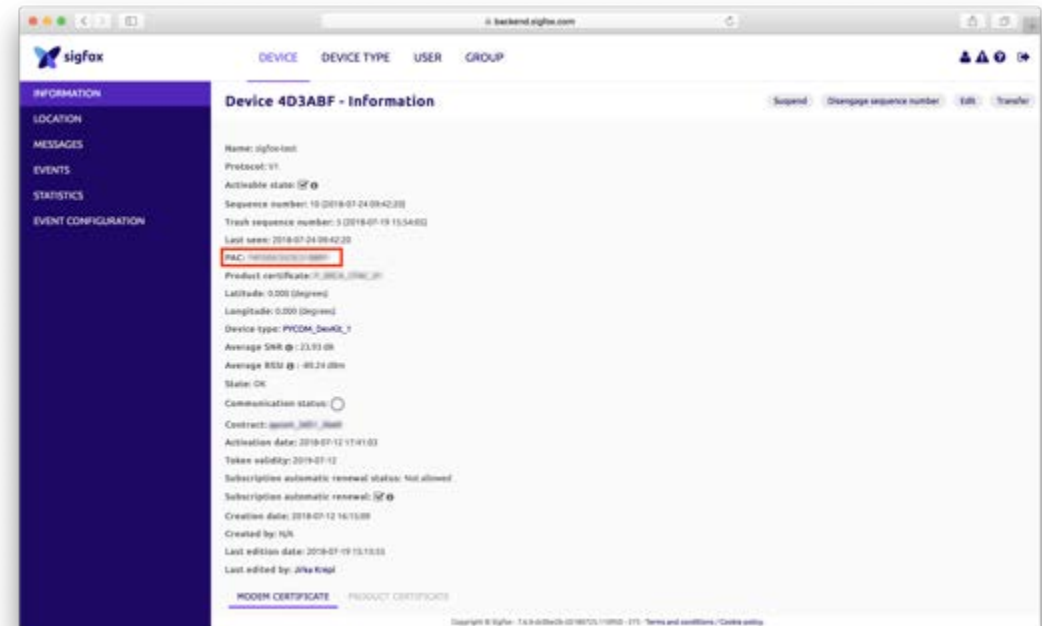


<https://docs.pycom.io/gettingstarted/registration/sigfox/>

TRANSFERRING A DEVICE TO ANOTHER SIGFOX ACCOUNT

4. The new **Sigfox PAC** should be shown

5. Once you know your new **Sigfox PAC** go to <https://buy.sigfox.com/activate> and register with the other account



<https://docs.pycom.io/gettingstarted/registration/sigfox/>

HAVE A PLAY AROUND AND SEE WHAT HAPPENS!

