webdyn

Flexitron group

TITAN

Application Note 61

Scripts - Communication with Android devices

Scripts - Communication with Android devices

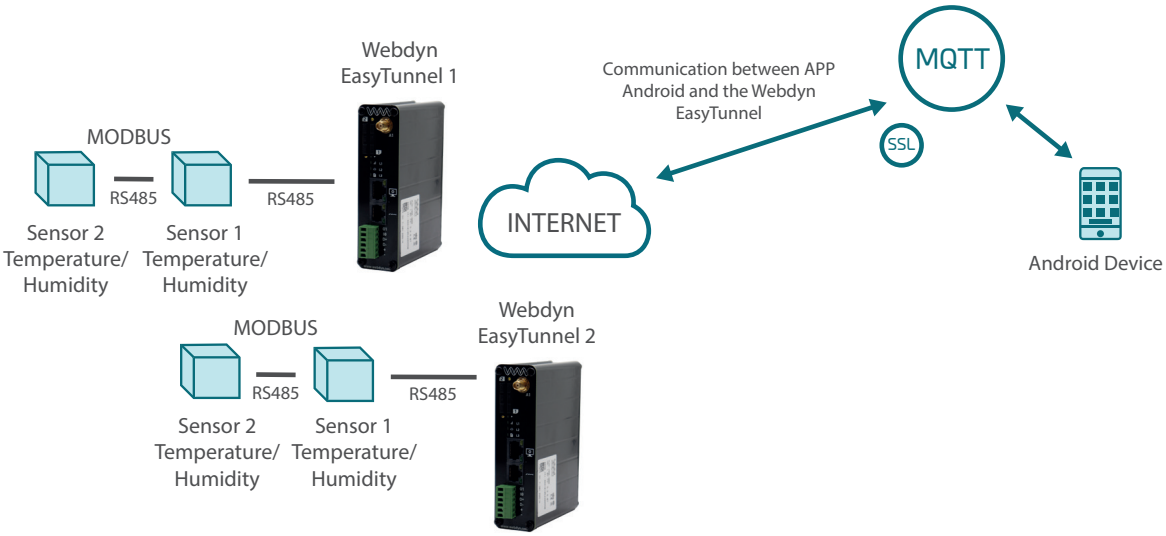
1. Scenario Details

TITAN-based devices have all the typical functionalities of 4G/3G/2G routers, as well as a series of added features that make them one of the most feature-packed routers on the market.

One of the additional features available is the possibility to execute a user script and its communication with applications developed in Android via MQTT/MQTTS communications.

2. Description of the Example

In this example we will implement a user script inside a Webdyn EasyTunnel device to receive commands from an Android app. From the app, the "getdata" command can be sent to the Webdyn EasyTunnel device so that it can read the Modbus RTU temperature and humidity registers from two sensors via its RS485 serial port and return the reading to the Android app for viewing. It will also be possible to send AT commands from the Android app to obtain data such as coverage, read configurations, perform resets, etc. Third party tools (brokers, tools to develop the example Android app, etc.) will be used in this AN50 Demo Application Note.

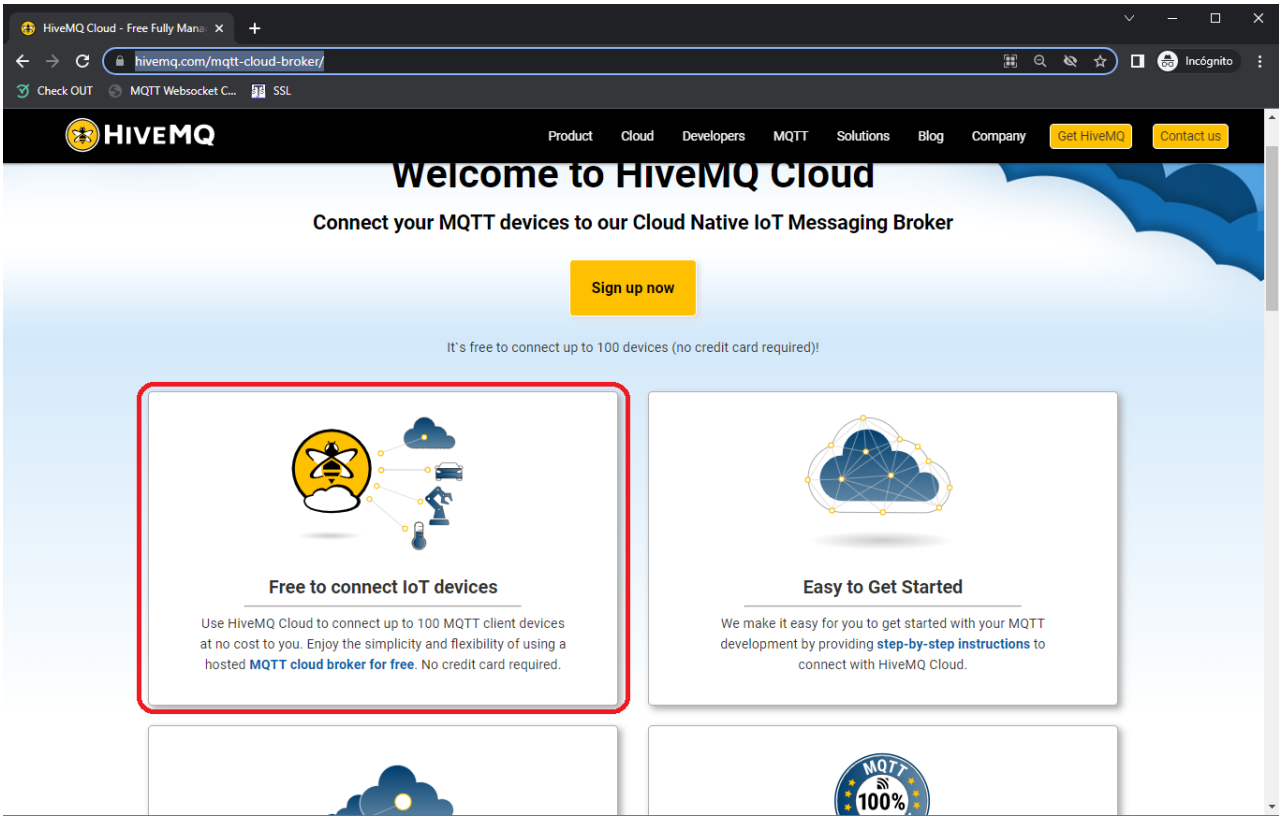


In this example scenario we will assume that Sensor 1 has the modbus RTU address @1 and Sensor 2 has the modbus RTU address @2. For both sensors, the modbus register for temperature will be the register with address "1", and for humidity the register with modbus address "2". RS485 port configuration is 9600,8,N,1

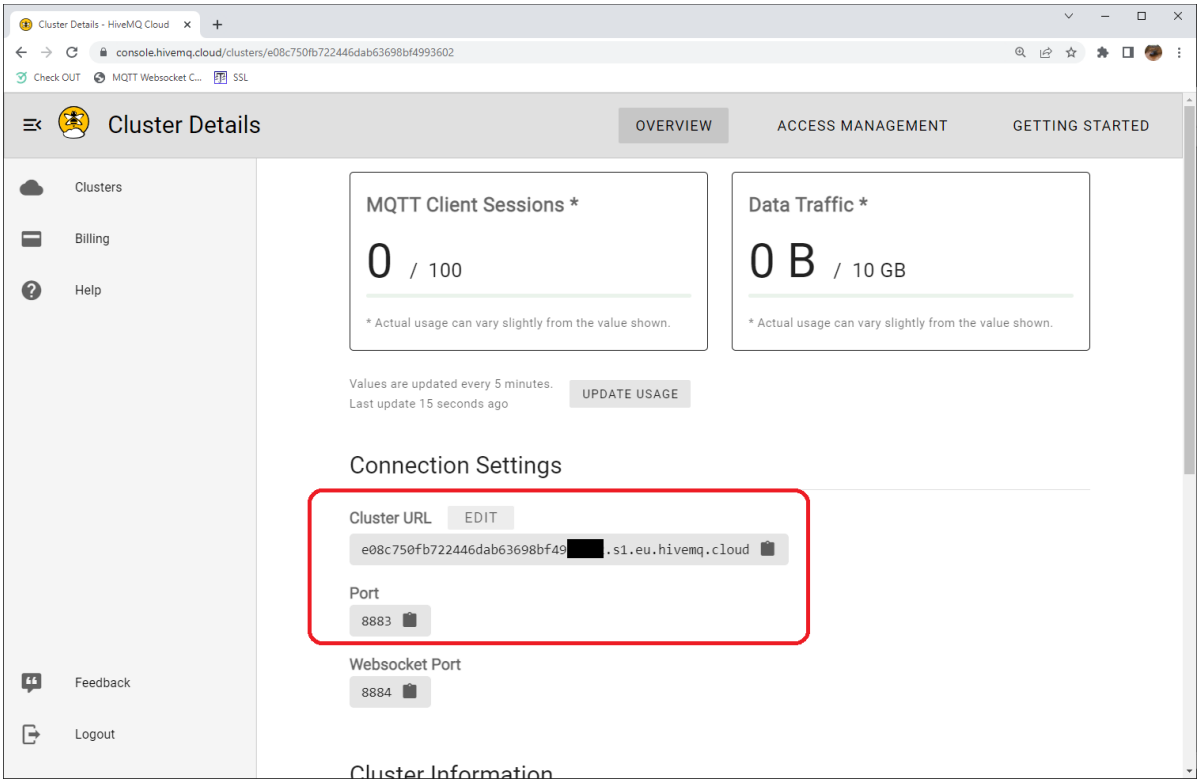
3. Configuring an MQTT broker

Communication between TITAN-based devices and the Android application will take place via MQTT (MQTTS). This requires an MQTT broker. In this example we will create an MQTT broker using a free "HIVEMQ" account (www.hivemq.com), where it is currently possible to create a broker free of charge for the simultaneous connection of up to 100 MQTT/MQTTS devices.

To do so, please go to the URL https://www.hivemq.com/mqtt-cloud-broker/ , select the option "MQTT cloud broker for free" and register for the service to create your account.

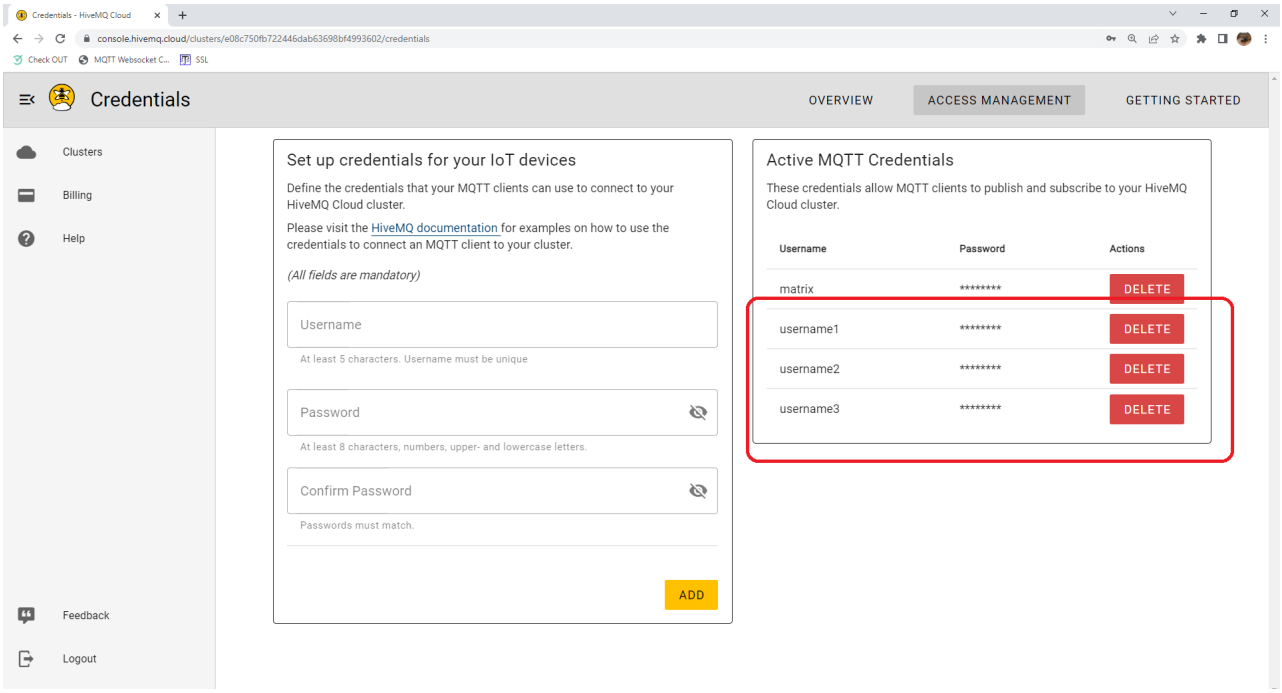


Once the account has been created, the URL of the broker generated will be displayed on the screen shown, which is where we will need to interconnect our Android app and the TITAN-based devices. Note that for this example we will use the TCP port for SSL/TLS 8883 communications.



Then, go to the top menu on this website "ACCESS MANAGEMENT", where you will generate the different usernames and passwords for the devices and apps that you wish to connect to the broker. As in this example, we wish to connect two Webdyn EasyTunnel devices and one Android app, so we will create three user accounts / passwords.

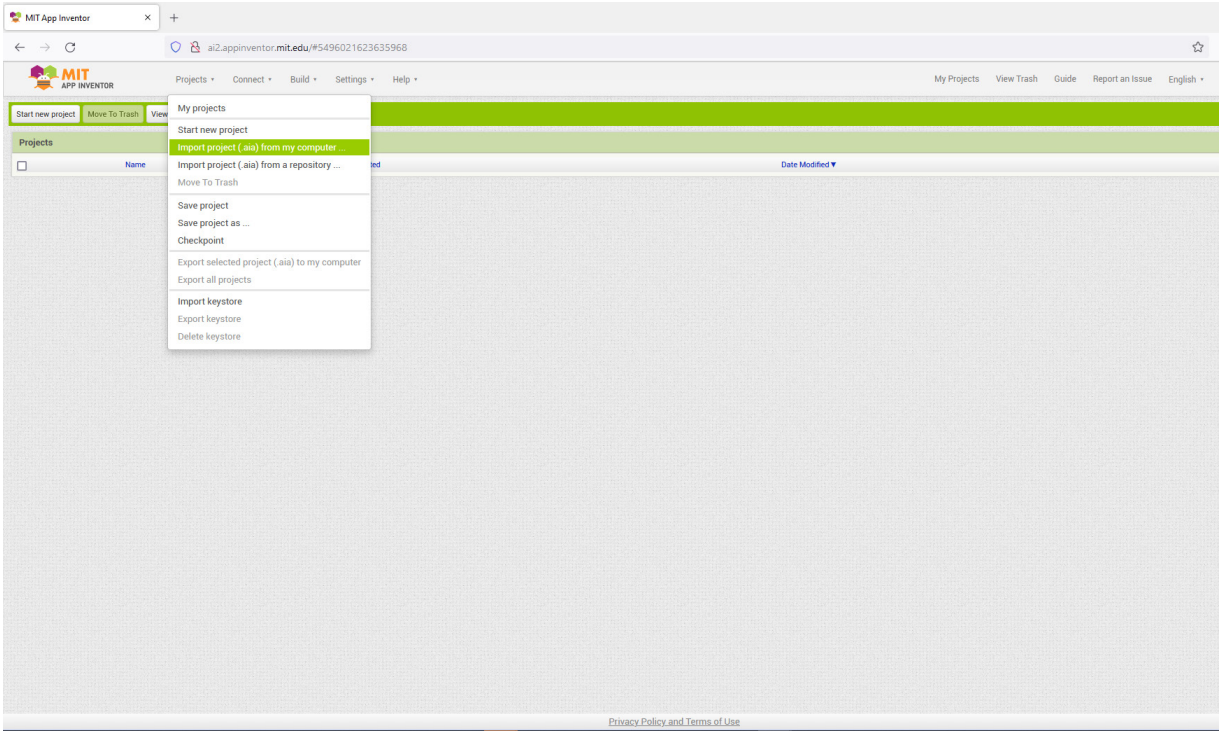
Once this has been done, our free MQTT broker is created and configured for up to 100 devices.



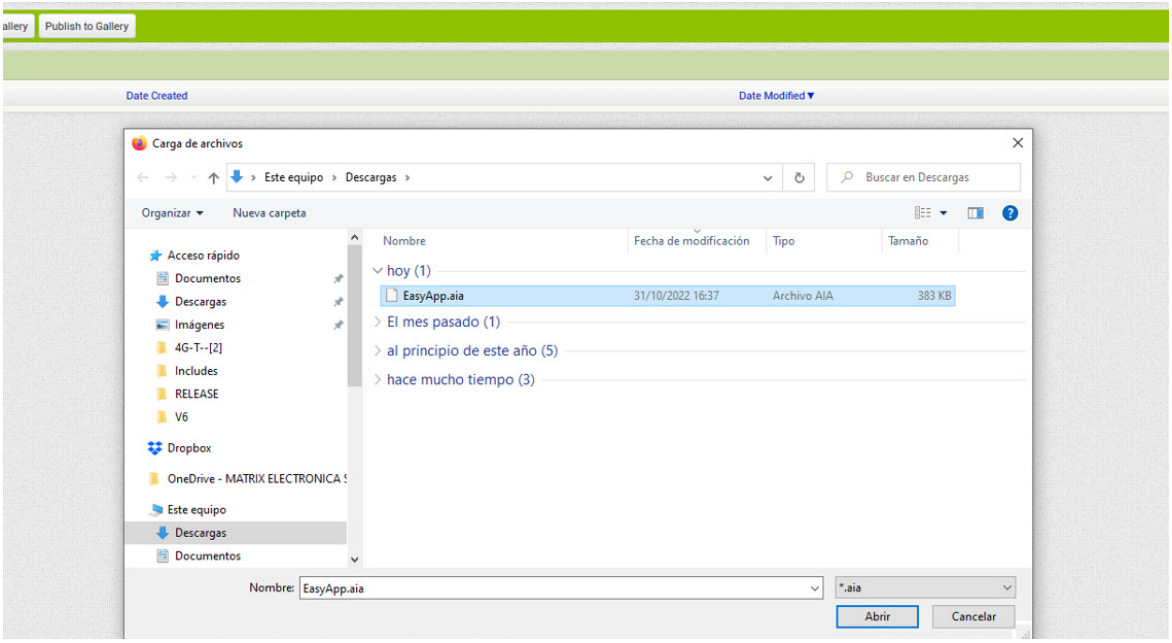
4. Creating a mini test Android APP

It's time to create a test APP. In this application note, we will create a simple app for Android devices using the free MIT tool "App Inventor". To do so, you will need to create an account via the URL https://appinventor.mit.edu

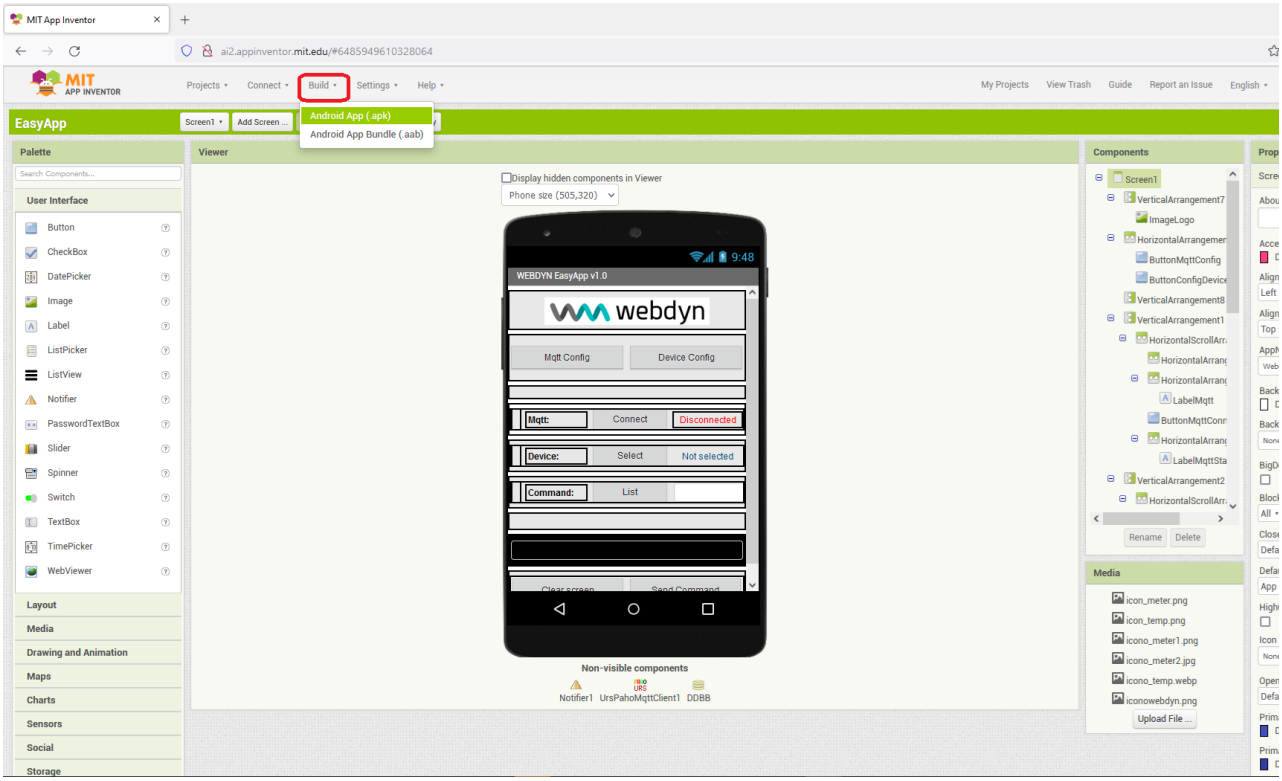
Once you have created your account in App Inventor, you can request from Matrix Electronics, if you wish, the project template file "EasyApp.aia" — a very basic example already built, which you can modify as you wish.



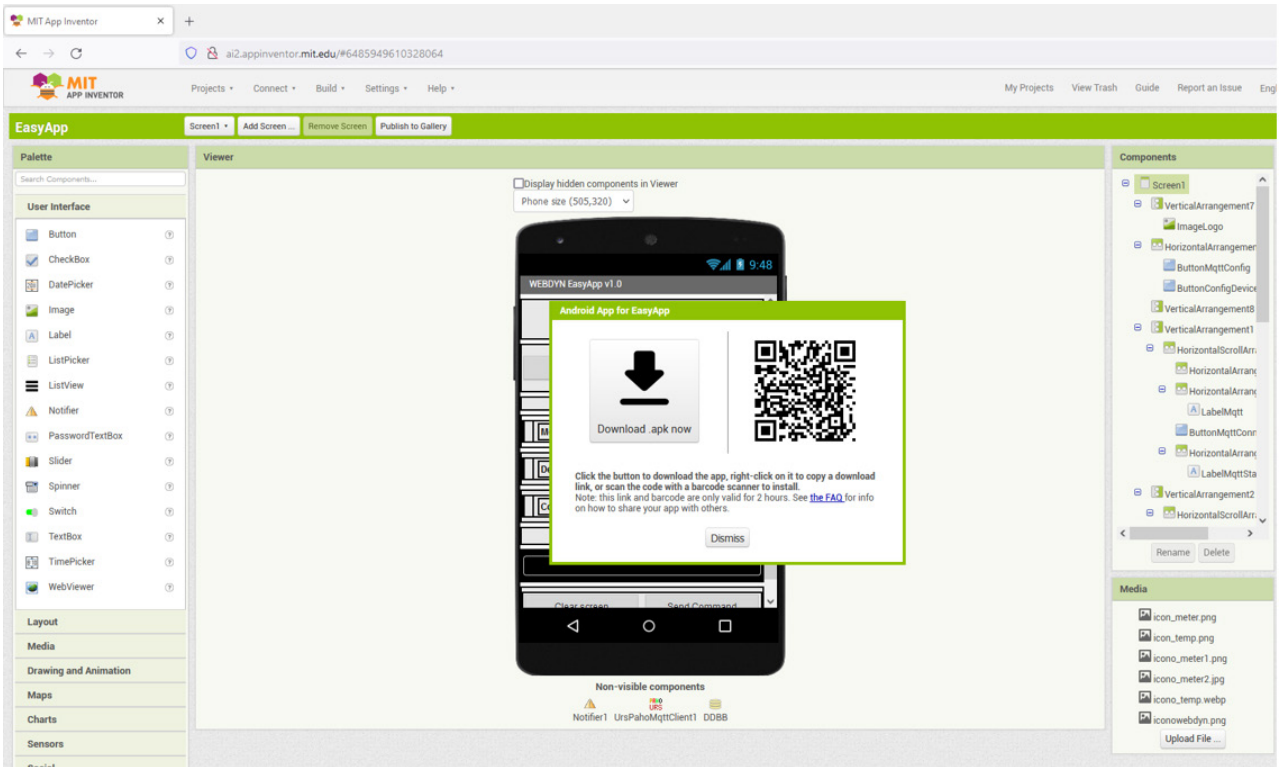
As shown in the image above, select the "Import Project" option in the "Projects" menu and specify the location of the template file "EasyApp.aia".



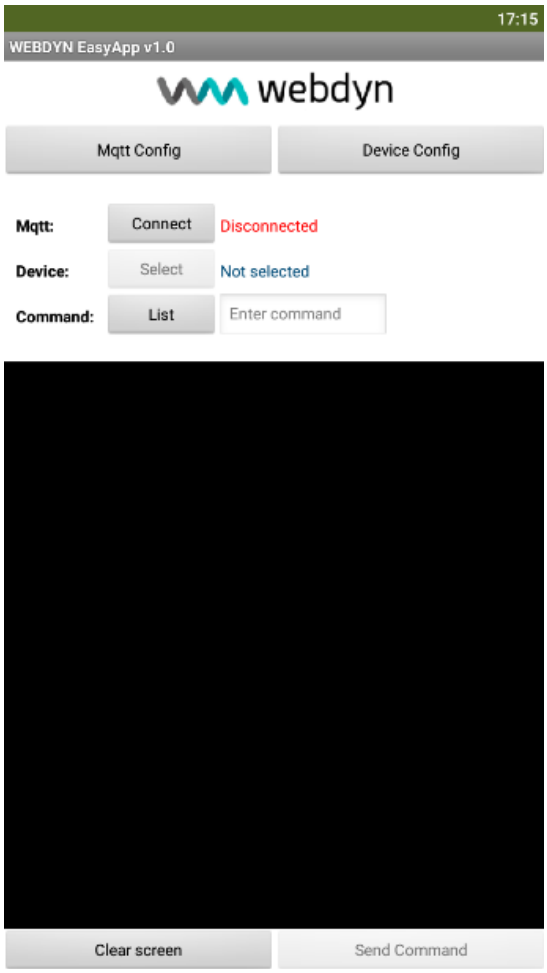
Once the project is loaded (we advise against modifying it this first time to check that everything works correctly), you can compile it from the menu "Build => Android App (.apk)", in order to generate the compiled file to install on an Android mobile phone.



At the end of the compilation process (1-2 mins), a QR code will appear on the screen. Use the camera on your Android phone to read the QR code. This will allow you to install your test app.



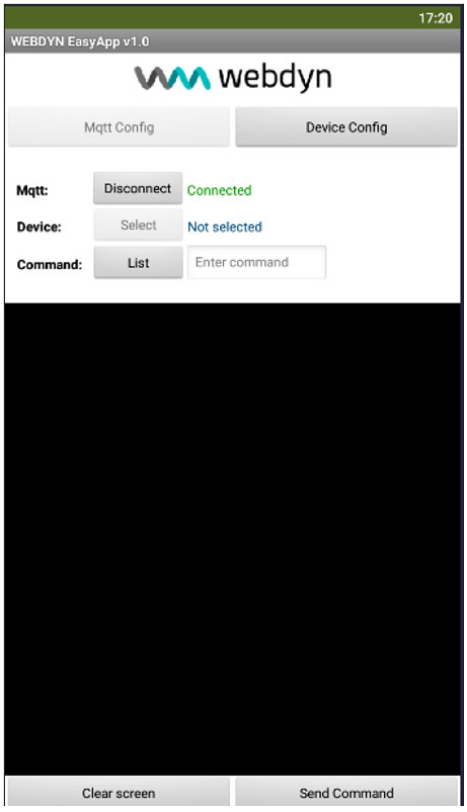
Once the app is installed, you will be able to open it on your mobile phone. This is how it will look.



5. Connecting the test Android APP to the MQTT broker

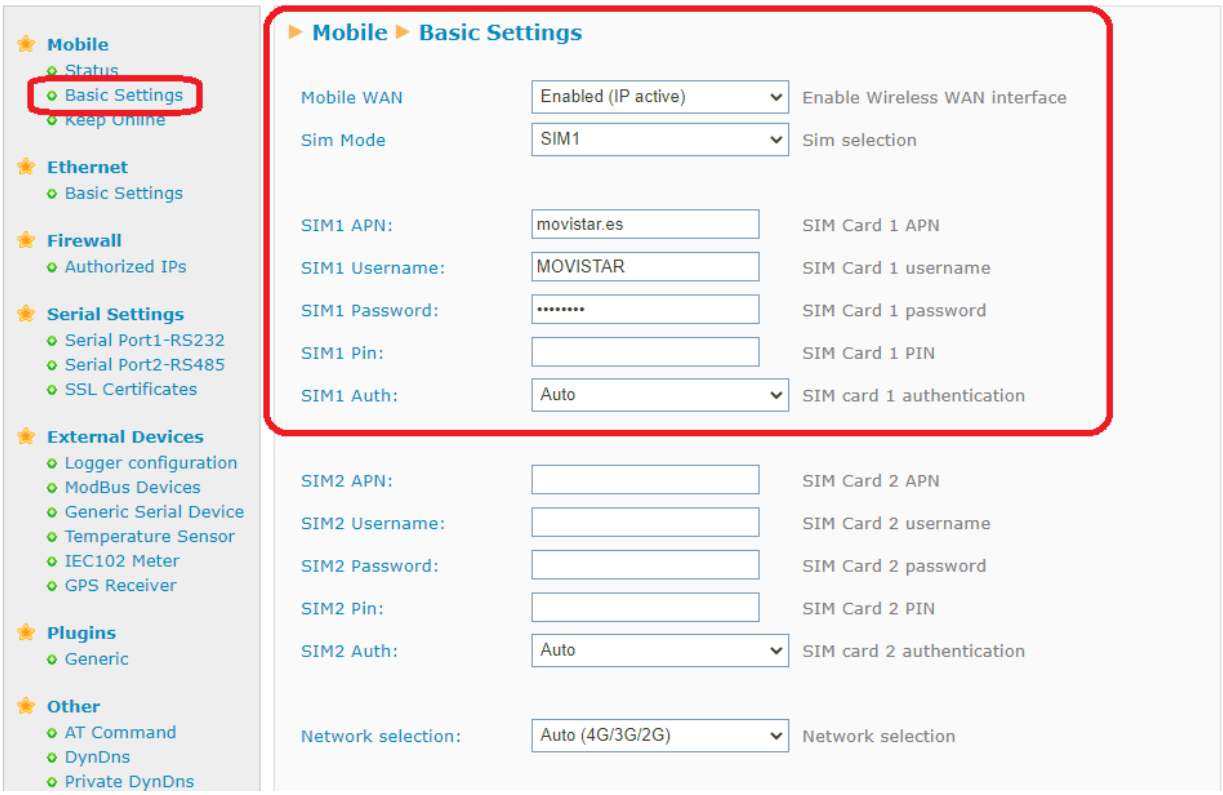
Now, we are going to configure the app to connect it to the MQTT broker that we created earlier. To do this, please click on the “Mqtt Config" button. Now, enter the appropriate settings on the resulting screen: the URL of the broker you got; the port 8883; the connection ID you wish (it must be different on each device/app); one of the usernames/passwords you created earlier for the broker; and, as topic MQTT responses, the text: /EASYAPP/APP1

Once you have configured the MQTT broker configuration data, click on the "Save" button. If you then press the "Connect" button, the text "Connected" should be displayed in green, as shown in the image below. This will indicate that the APP has been successfully connected to the MQTT broker.

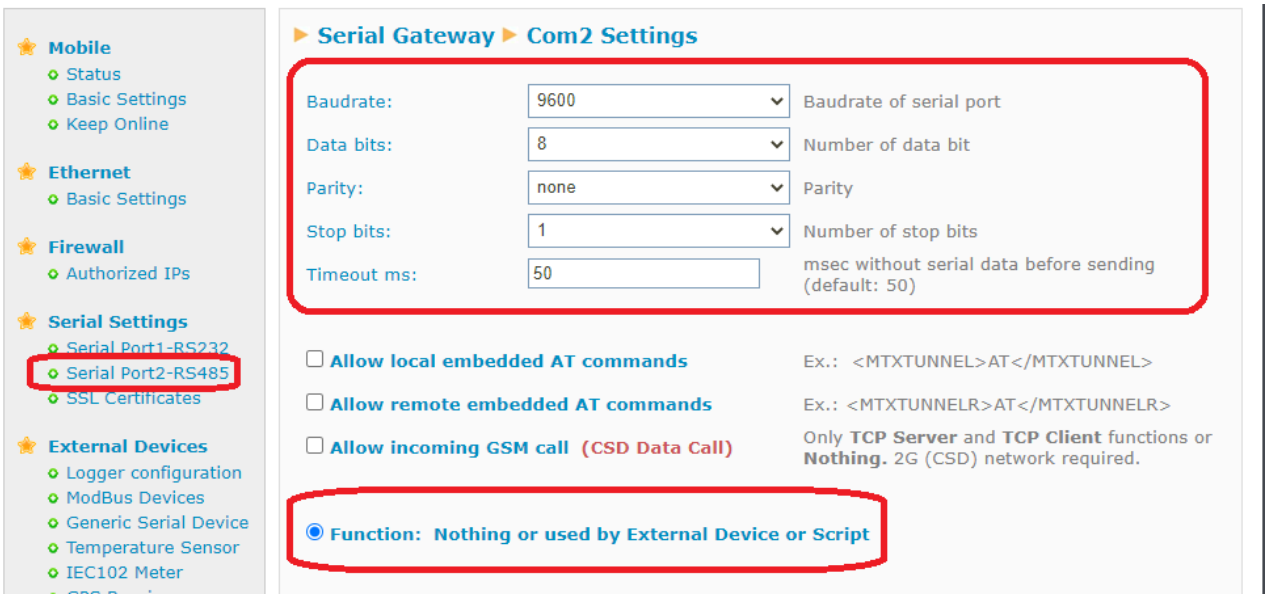


6. Configuring the Webdyn EasyTunnel

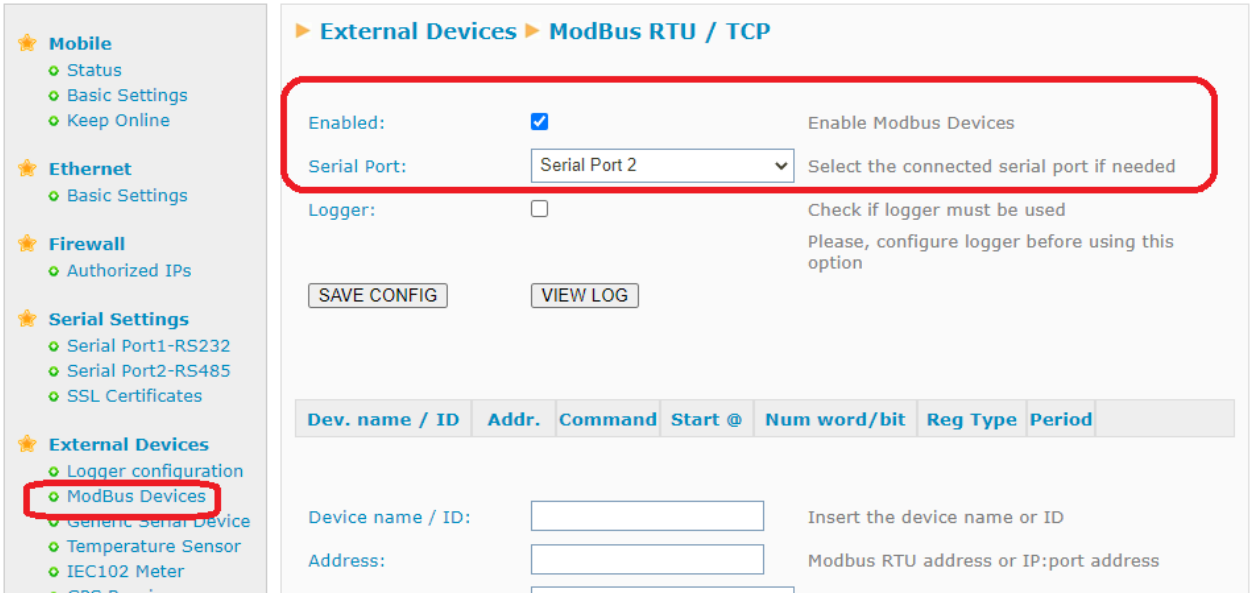
Next, we will fully configure the Webdyn EasyTunnel device. This will need to be done on each of the two devices. You must first configure the "Mobile => Basic Settings" section by entering at least the settings indicated on the following screen.



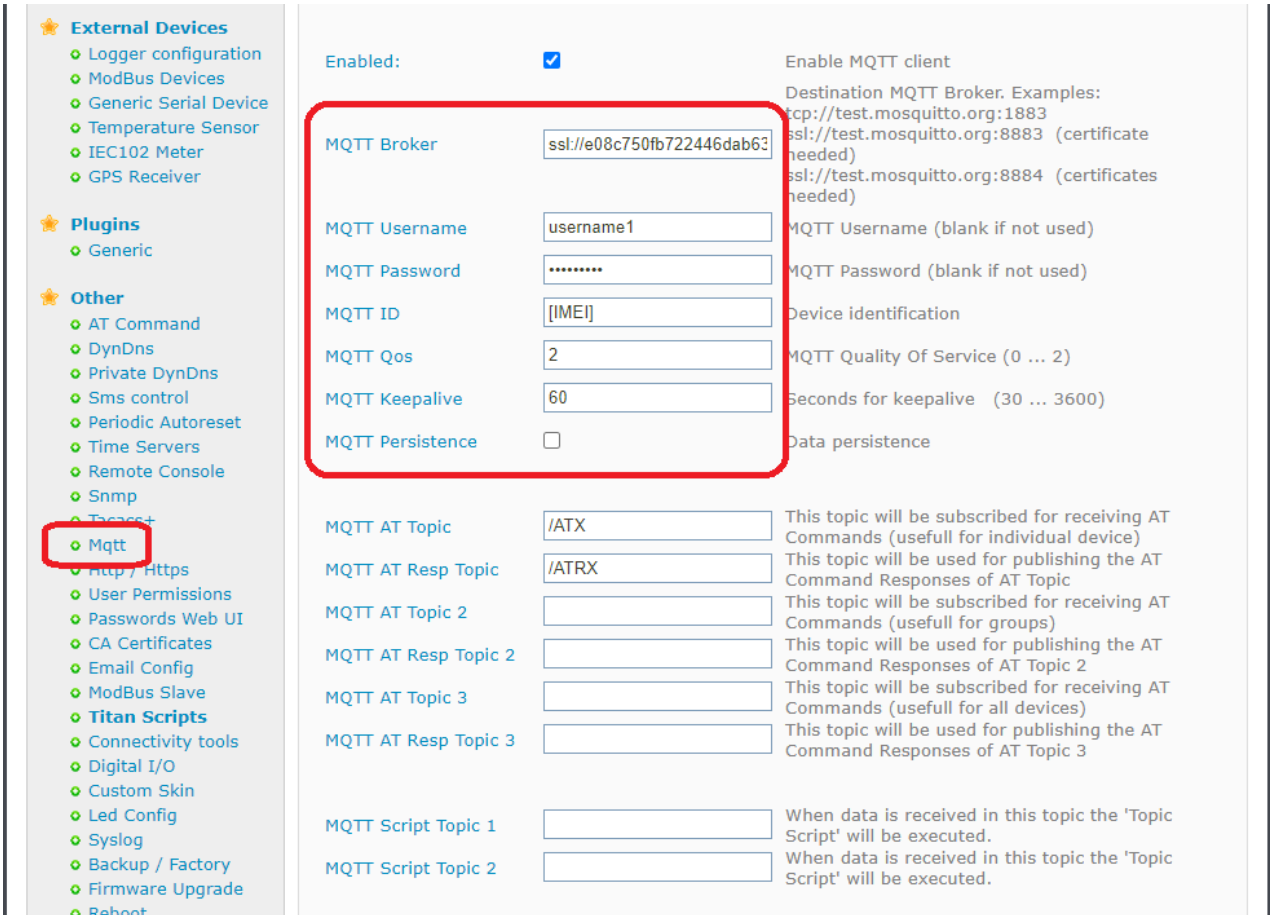
Then, as we are going to use the RS485 port to read Modbus sensors, we will fill in the configuration options in the "Serial Settings => Serial Port2-RS485" section of the menu.



Now, in the "External Devices => Modbus Devices" section, we will indicate that we are going to use the serial port "Serial Port 2" to read modbus RTU devices. We will also enable the Modbus service by checking the "Enabled" box.



Next, we will go to the "Other => Mqtt" section, where we will configure the service to connect to the MQTTS broker created earlier.

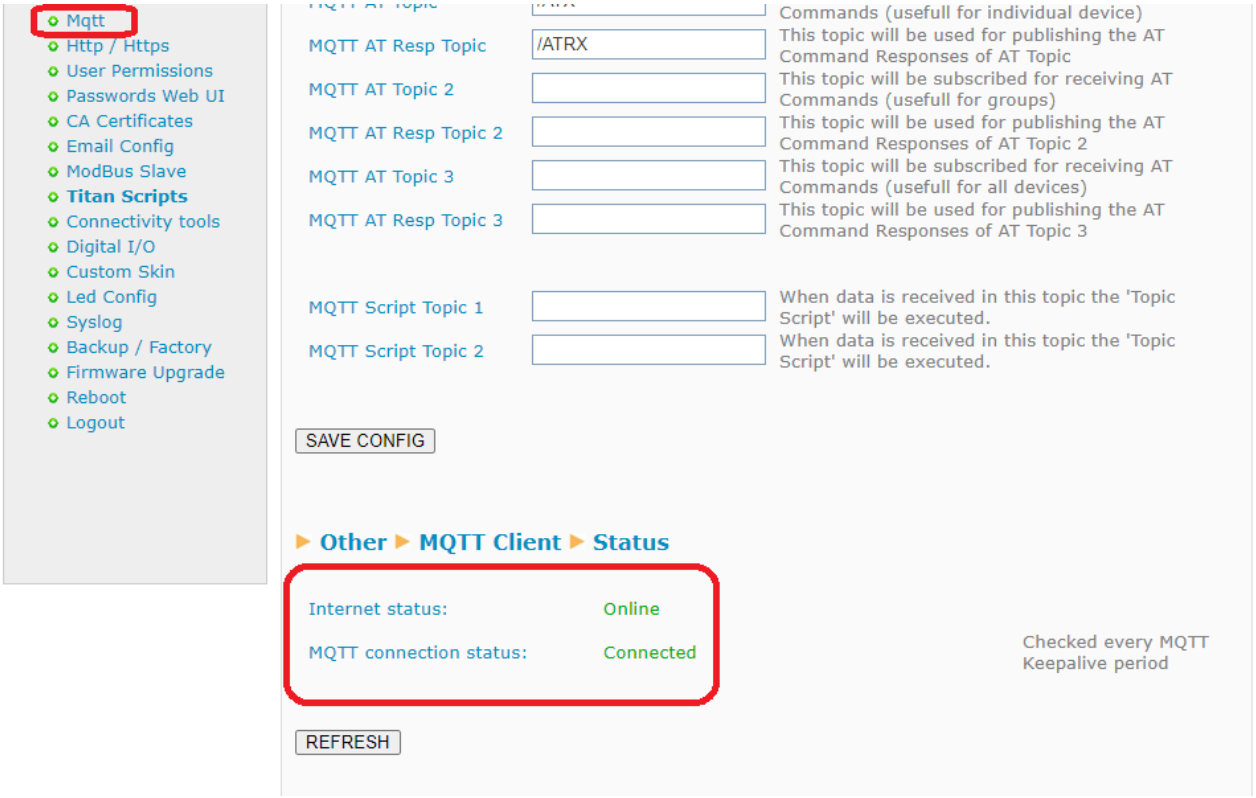


Be careful with the broker's URL, as you will need to specify it as follows:

ssl://<urlBroker>:8883 (so, don't forget the header "ssl://" and the ending with the port "8883")

Then, we will relaunch Webdyn EasyTunnel and after that, in the same menu "Other => Mqtt", we should see that the connection with the broker has been made successfully. This may take a few seconds.

Next, we will go to the "Other => Mqtt" section, where we will configure the service to connect to the MQTTS broker created earlier.

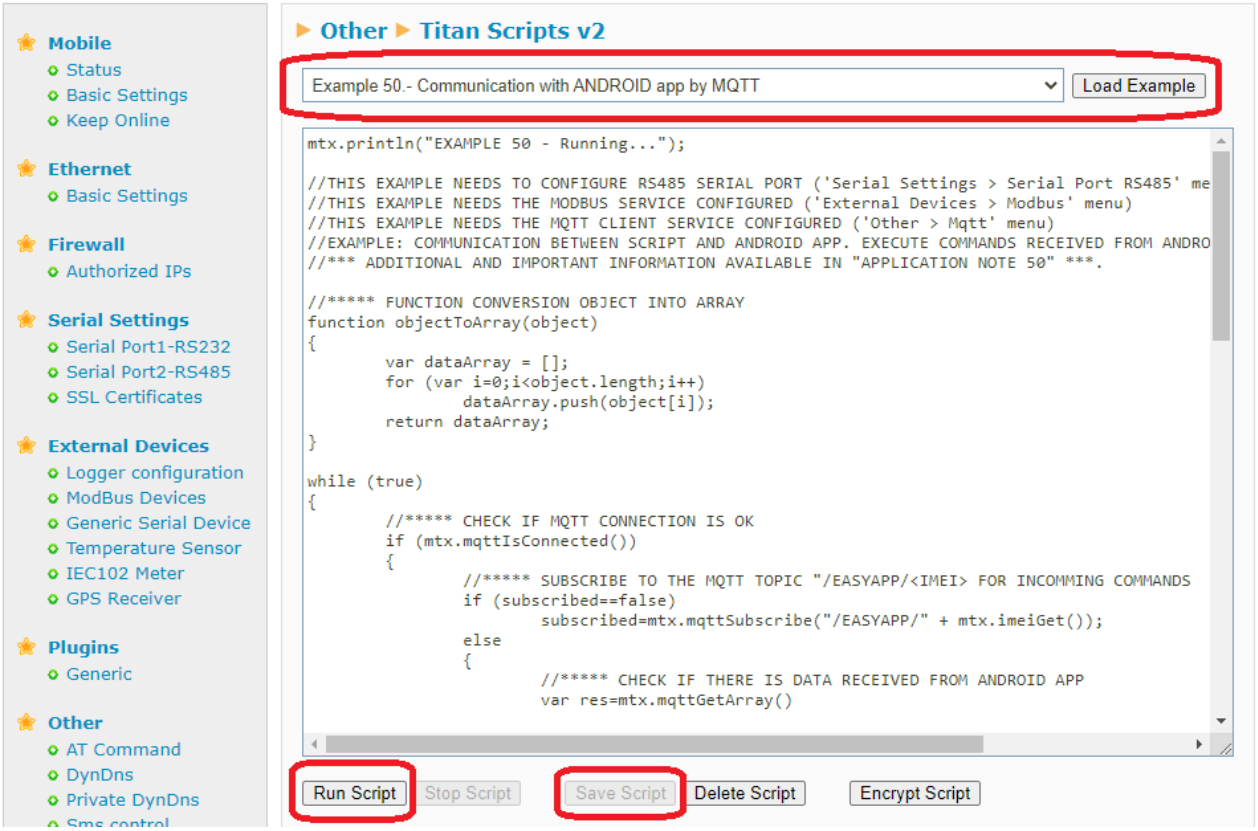


If necessary, at the bottom of this screen, and in the menu "Other => CA Certificates", you can enter the certificates you may need.

7. Running the script

The only thing left to do now is to run the appropriate script on the Webdyn EasyTunnel device. To do so, go to the “Other => Titan Scripts” menu. Once on this screen, select example number 50 (Communication with ANDROID app by MQTT) and click on the "Load Example" button.

We can then test our script by clicking on the “Run script” button.

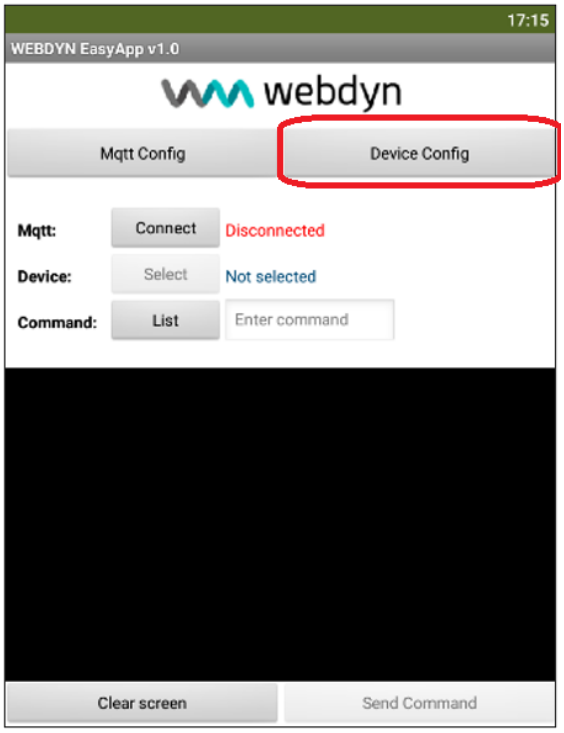


The code in this example script works quite simply. Essentially, the script subscribes to the MQTT topic "/EASYAPP/<IMEI>" and continuously checks if commands have been received in this topic (commands sent by the app). The commands received are in textual form and have a simple structure <topicForResponse>###<command>. In other words, the script will execute the received <command> and send the response to the <topicForResponse> topic, which will be received by the Android app.

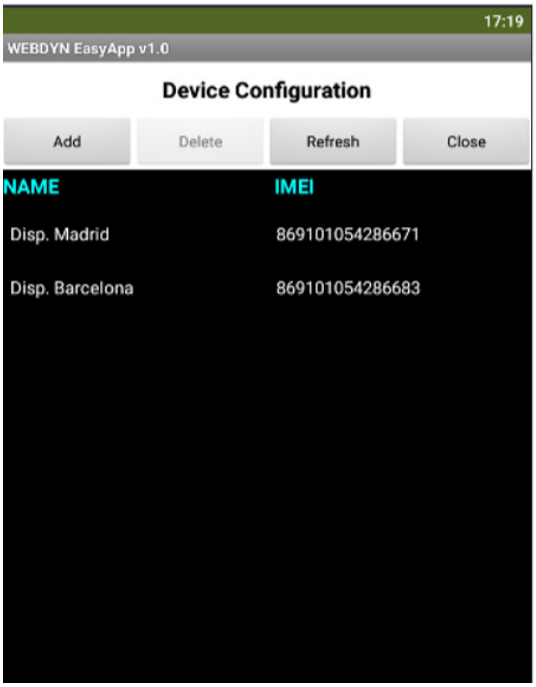
If the command received by the script is "reset", the Webdyn EasyTunnel device will be reset. If the command is "getData", the Webdyn EasyTunnel will read the Modbus RTU temperature and humidity registers of the sensors and return a String with that response. If it is another type of command, it will interpret it as an AT command and attempt to execute it and return the response.

8. Testing the Android App

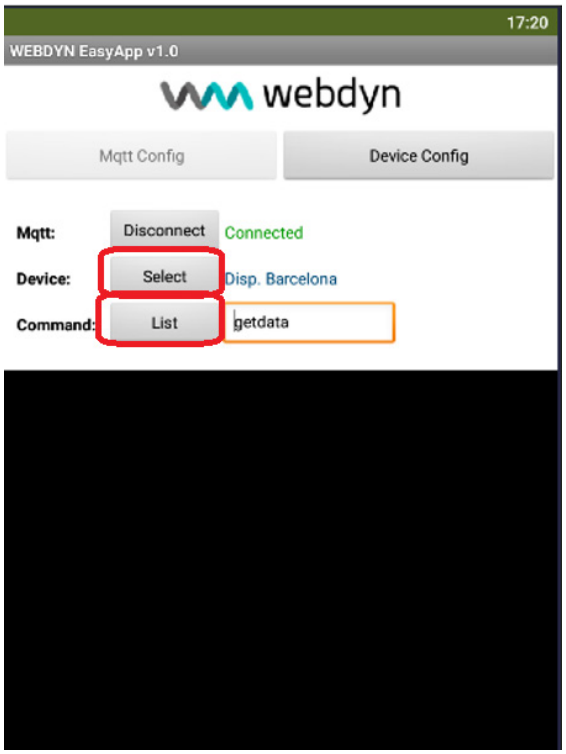
In this final point, point 7, all that remains is to test the App. Start by adding the Webdyn EasyTunnel devices you wish to manage to the App. For this example, we will configure two by clicking on the "Device Config" button.



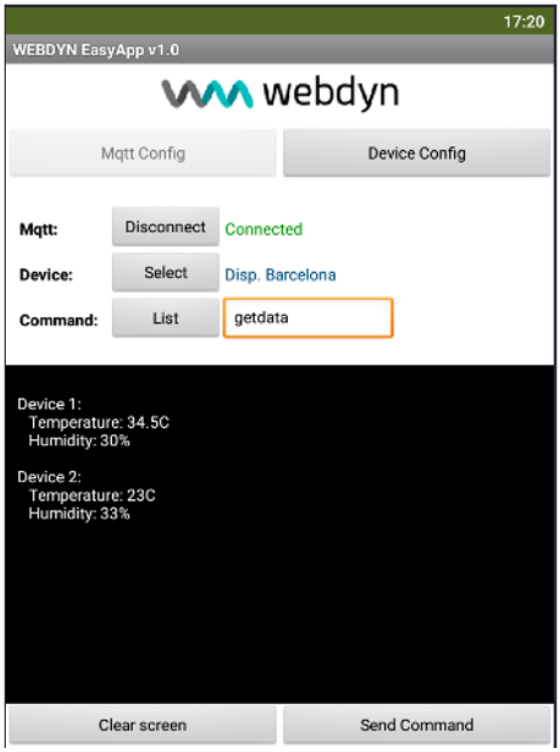
Here, we enter the two devices, indicating a name and its IMEI (the IMEI of each Webdyn EasyTunnel can be obtained in the menu "Mobile => Status") for each one

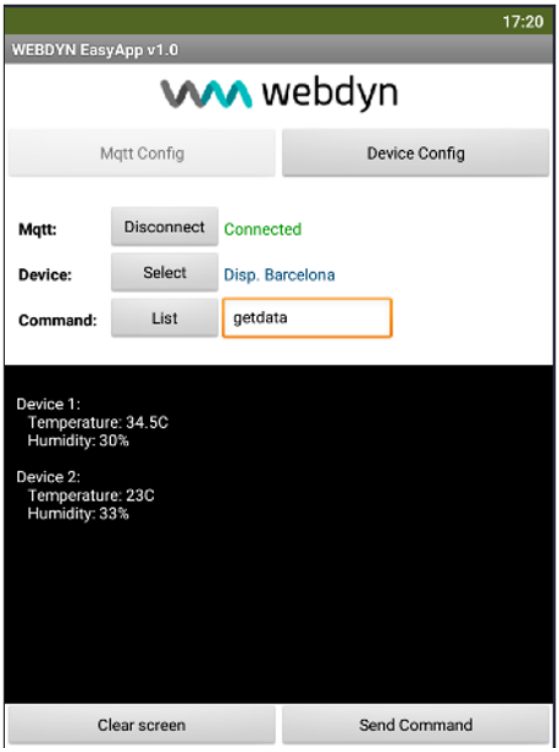


Now, press the "Select" button to choose the device you wish to communicate with. After selecting the device, click on the "List" button and choose the "getdata" command. Finally, click on the "Send Command" button at the bottom of the screen.



If everything works correctly, we will receive a response from the device immediately.





If, with the "List" button, we select the "custom" command, we can also specify AT commands such as the "AT+CSQ" command to obtain coverage, but it would also be possible to execute other AT commands remotely to read a digital input, to change or read the configuration, etc.

Any questions?

Please direct your enquiries to iotsupport@mtxm2m.com