Tutorial : IFTTT and VERA Integration with IFTTT

Introduction

- IFTTT (https://ifttt.com) is a well-known internet software service which enables the creation of basic “applets” tighting together sources of events and actions on various objects or other services. So IFTTT applets react to “triggers” and perform “actions”. There are tons of supported triggers and actions like sending notifications , sending emails, editing documents on your preferred cloud storage, acting on IOT devices etc.

- VERA so far is not one of of the supported IFTTT “services”. GetVera company is working on an VERA integration as an IFTTT action part (so IFTTT can act on a device or run a scene on vera) but there is no support for incoming data integrations where VERA would act as a “triggers” in a applet. The focus of this tutorial is to show how easy it is to use VERA as a trigger using ALTUI new IFTTT integration.

- ALTUI supports since a while the concept of “Data Storage” integration to enable a user to send some data externally, via a web service call, whenever a particular device variable changes. Supported integrations so far were thingspeak , emoncms, Datayours and the list is open to extension by third party. In the latest ALTUI release a new internal “Data Storage” integration is added for IFTTT thus enabling any chosen device/variable change to trigger some IFTTT actions

Typical example of accomplishments using VERA as a trigger

1. Track the changed values of a device over time in a google sheet to get some history in a free/low cost/zero server / zero DB fashion

2. Send an email, or a mobile device push notification whenever some device value changes.

This document is a step by step tutorial, with screen shots, that will walk you through the use case #1 here above.

Prerequisites

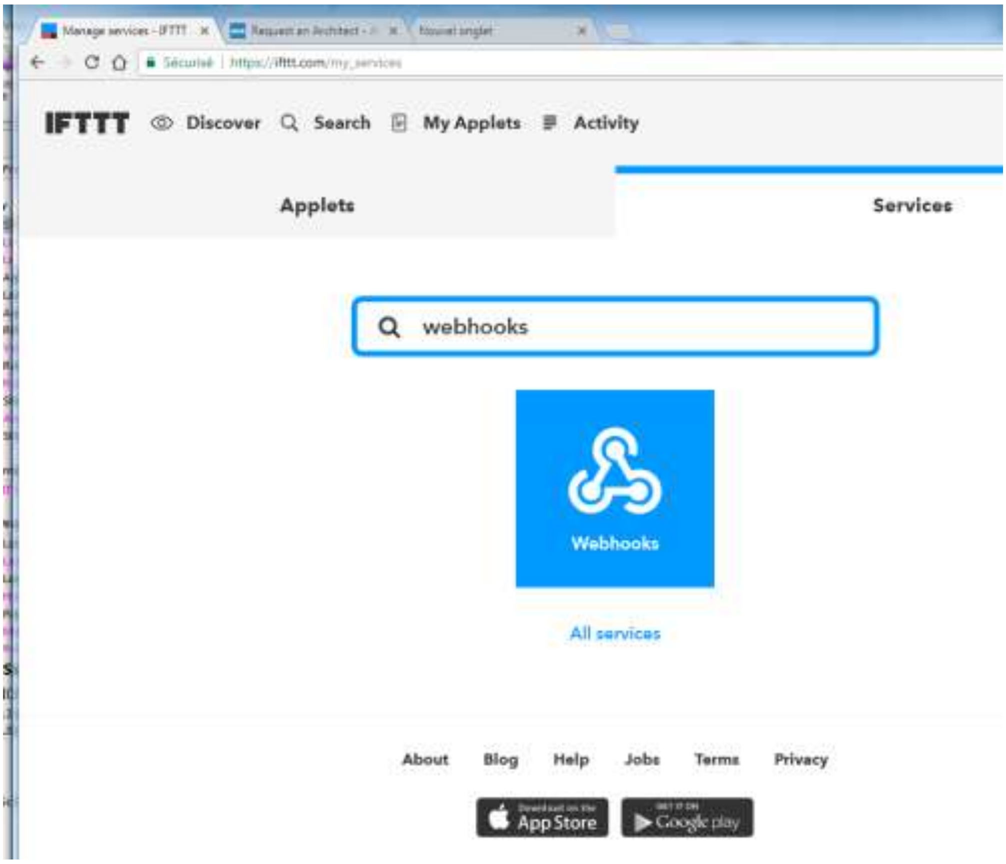
- Vera or openLuup
- Running ALTUI plugin http://apps.mios.com/plugin.php?id=8246
- An IFTTT account
- A Google account with a google drive folder where we will store a google sheet

3 fundamental steps

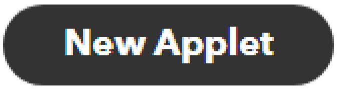
1. Configuration of IFTTT : where we configure IFTTT and prepare the necessary IFTTT applet
2. Configuration of ALTUI : where we configure ALTUI to send data notifications to IFTTT
3. Fine tuning : some improvements

Configuration of IFTTT

- First you need to have an IFTTT account https://ifttt.com/discover
- We will need to use the IFTTT service webhooks. For this you can go to My Applets and search for “WebHooks” and configure it

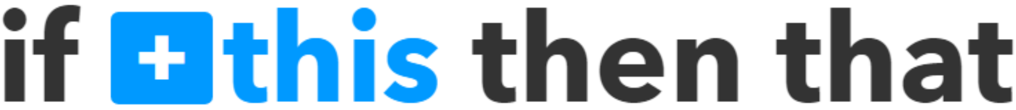


- Then go to “My Applets” and create a new applet clicking on the New Applet button



- You will be presented with a screen like this showing the basic structure of an applet with a **THIS** part and a **THAT** part.

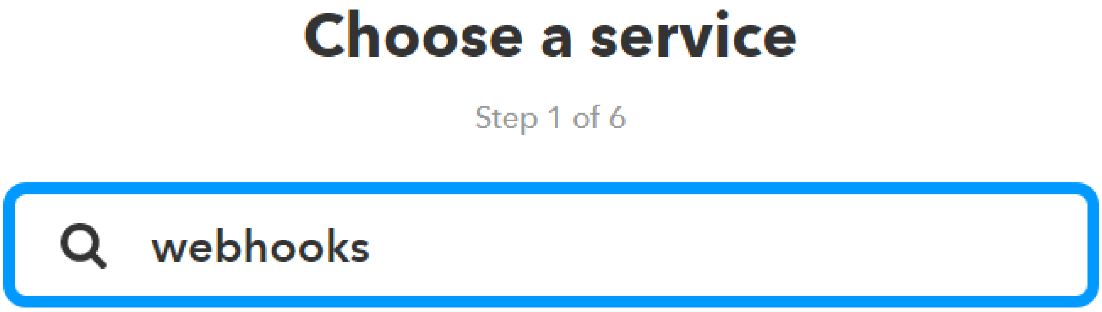
New Applet



if + this then that

Step1 – prepare the applet

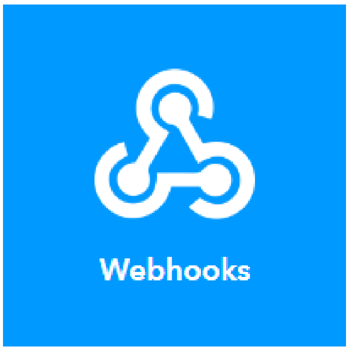
- In our case , we want to receive a web service from VERA so the **THIS** part will be a WebHook. Click on THIS, then type WebHook in the search field , then click on WebHooks bleu square



Choose a service

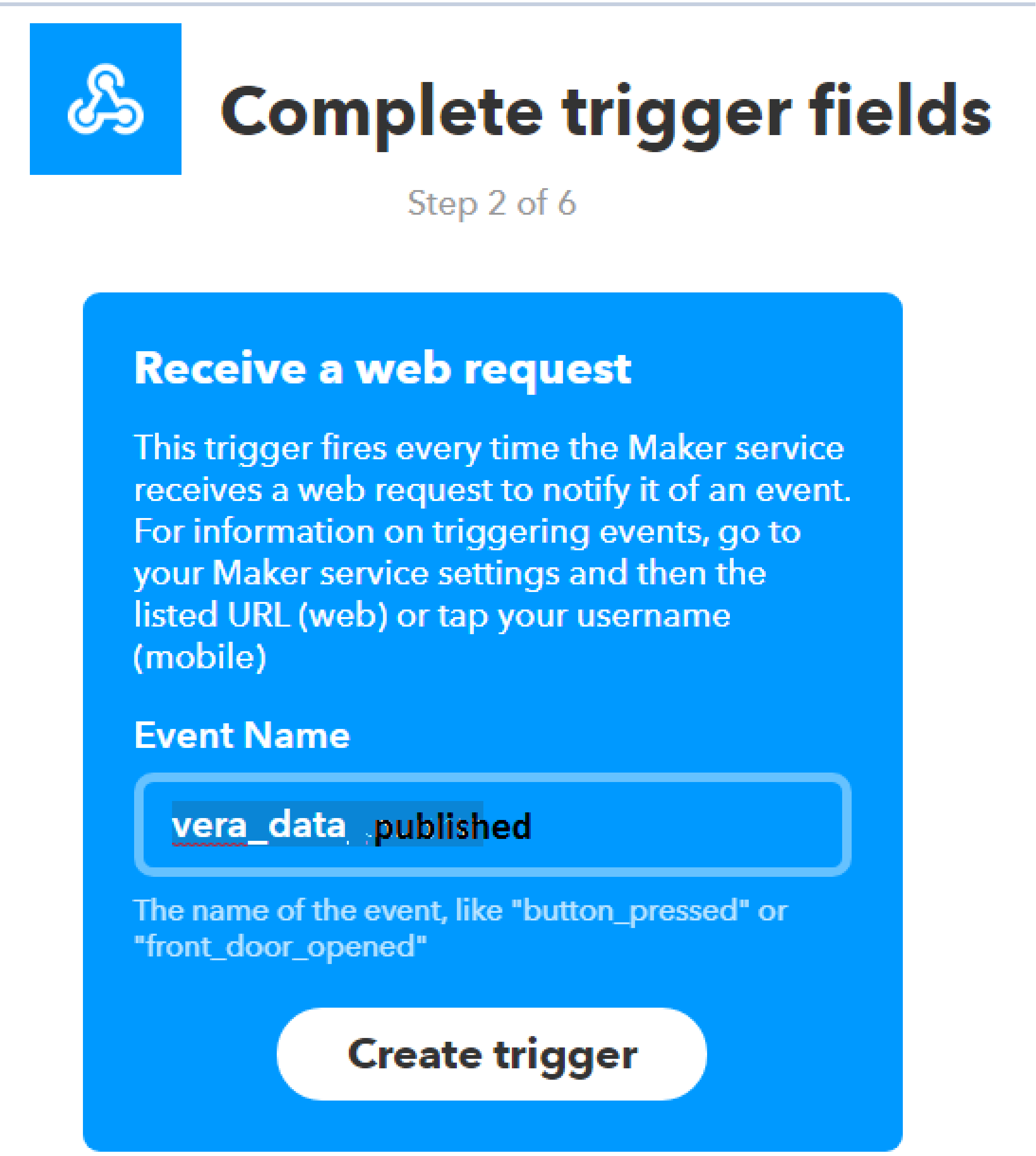
Step 1 of 6

webhooks



Step 2 – configure the webhook

- Click again on the “Receive a web request” bleu square and IFTTT will ask you for an event name. you can choose whatever you want here. You can have different event name per devices, or use the same if you just want all devices variable changes to go through the same action. Event name in the same google sheet can enable groupings differently, you can even have different files per event name. This is up to you. In my example I arbitrarily chose “vera_data_published” and all will be in the same file.
- Remember that event name it will be useful later



Complete trigger fields

Step 2 of 6

Receive a web request

This trigger fires every time the Maker service receives a web request to notify it of an event. For information on triggering events, go to your Maker service settings and then the listed URL (web) or tap your username (mobile)

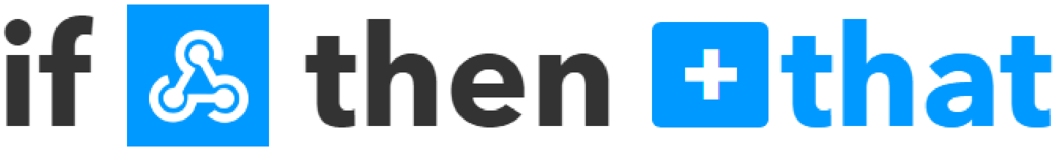
Event Name

vera_data_published

The name of the event, like "button_pressed" or "front_door_opened"

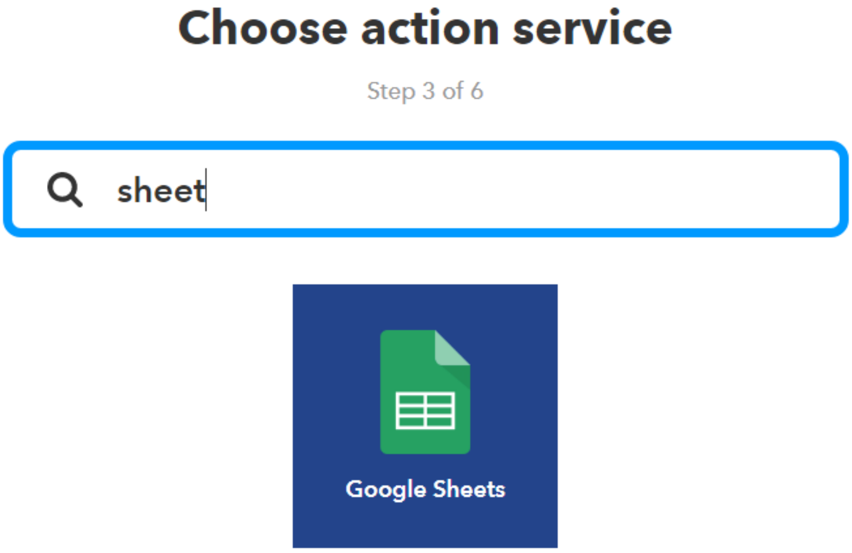
then click on “Create trigger”

- Now the Applet THIS part is complete and we still miss the THAT part. So click on the “That” word



Step 3 – configure google sheet action

- Choose a service (if you are part of VERA IFTTT beta program you may see the VERA action service but this **<u>is not what we are interested here</u>**)
- Search for google sheet and click on it



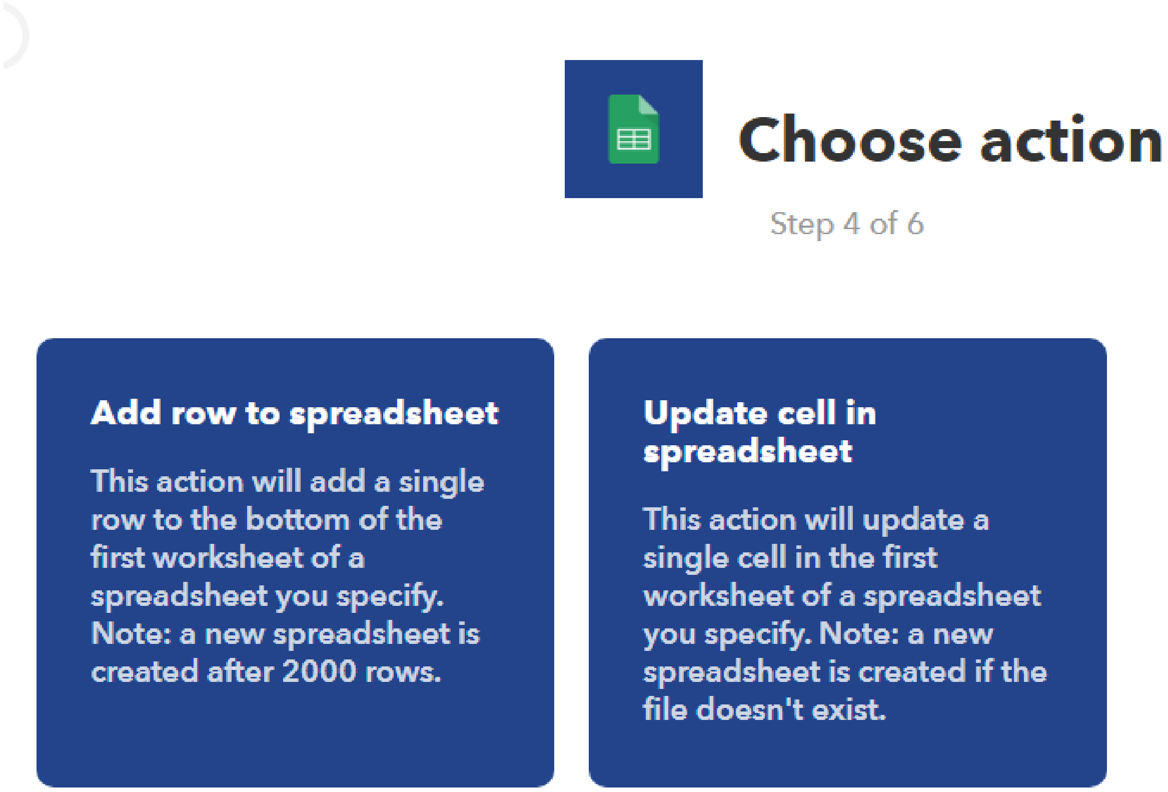
Choose action service

Step 3 of 6

sheet

Step4 - some simple formulas in the action

- We want to add a new row in a google sheet, every time we get a device variable change (for the chosen devices in ALTUI, we will see that part later).



Choose action

Step 4 of 6

Add row to spreadsheet

This action will add a single row to the bottom of the first worksheet of a spreadsheet you specify. Note: a new spreadsheet is created after 2000 rows.

Update cell in spreadsheet

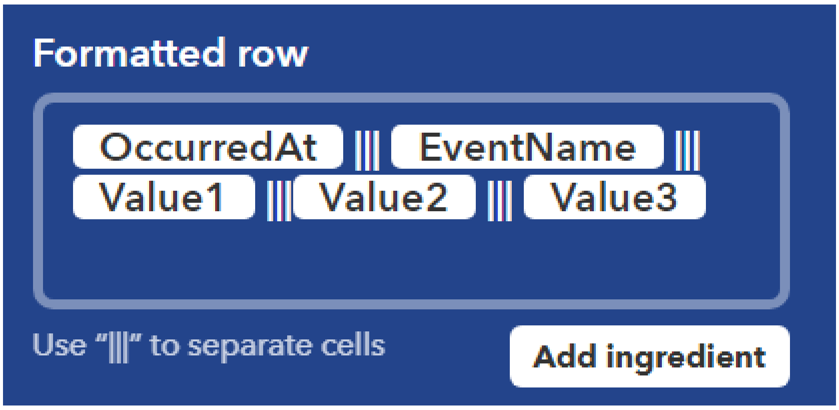
This action will update a single cell in the first worksheet of a spreadsheet you specify. Note: a new spreadsheet is created if the file doesn't exist.

- Click on Add row to spreadsheet

- By default the action is configured to use a google sheet named "IFTTT_Maker_Webhooks_Events", located in your google drive in a folder called IFTTT/MakerWebooks/{{EventName}} where {{EventName}} is the name you have chosen in step 2 so it will be IFTTT/MakerWebooks/vera_data_source

⇨ **Go ahead and create a google sheet of that name in that folder in your google drive. IFTTT would create the file alone but we are going to do some things in it later so we need to create it now by hand :**
File : /IFTTT/MakerWebooks/vera_data_source/TTT_Maker_Webhooks_Events

- The interesting part is the row that will be inserted in the sheet. It will allways be a new / last row of the first sheet tab . The row cells values will be what you put in the Formatted row field



Formatted row

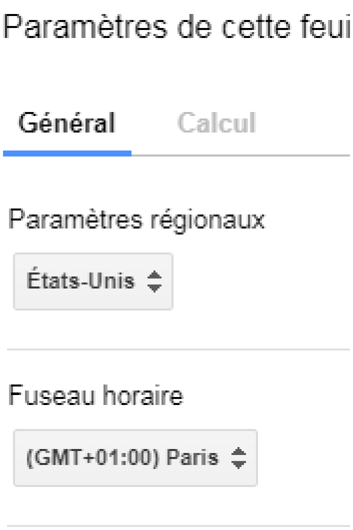
OccurredAt ||| EventName ||| Value1 ||| Value2 ||| Value3

Use "|||" to separate cells

- You can add as many cells as you want, providing you separate each cell definition by 3 vertical pipes : like "|||"

- This is the default but we can improve it quite a bit
  - **OccuredAt** is a textual field representing a date/time like "`December 4, 2017 at 09:33AM`" but that format is not very good in sheets for graphs or filtering. Fortunately you can use any google sheet formula. So I basically use 2 cells and formulas to translate into a valid date and a valid time:
    - Date : DATEVALUE(REGEXREPLACE("{{**OccurredAt**}}"," at .*",""))
    - Time: TIMEVALUE(REGEXREPLACE("{{**OccurredAt**}}",".* at ",""))

- NOTE: for non-English speakers, it is possible that this does not work because your google sheet locale is not set to English. We have no control over strings sent by IFTTT so to make it work, set your google sheet locale to English/USA. This can be done in the File / Parameters of the sheet menu command.



Paramètres de cette feui

Paramètres régionaux

États-Unis

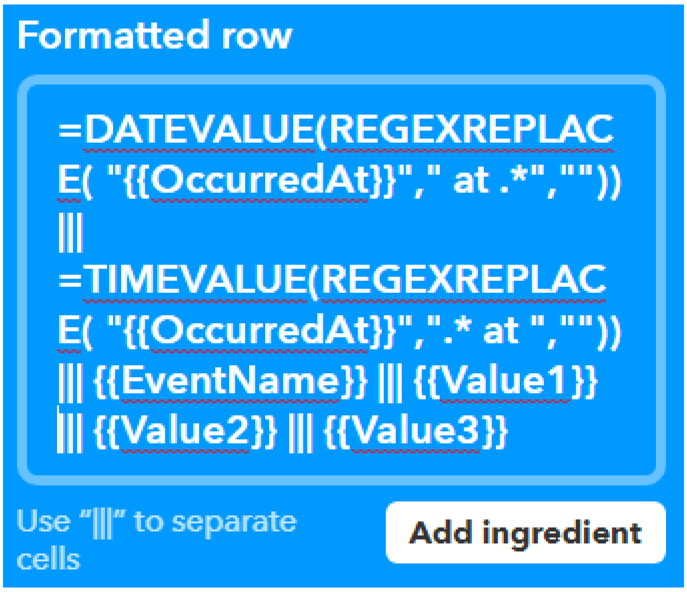
Fuseau horaire

(GMT+01:00) Paris

- For the Value1, Value2, Value3 , they are determined by ALTUI integration code. ALTUI will send :
  - **Value1** = the altuiid of the device (like 0-123)
  - **Value2** = the service":"variable that has changed like "urn:micasaverde-com:serviceId:DoorLock1:Status"
  - **Value3** = the new value taken by the variable

So the formula I put in Formatted Row field ends up being:

=DATEVALUE(REGEXREPLACE("{{**OccurredAt**}}"," at .*","")) ||| =TIMEVALUE(REGEXREPLACE("{{**OccurredAt**}}",".* at ","")) ||| {{**EventName**}} ||| {{**Value1**}} ||| {{**Value2**}} ||| {{**Value3**}}

- 

Formatted row

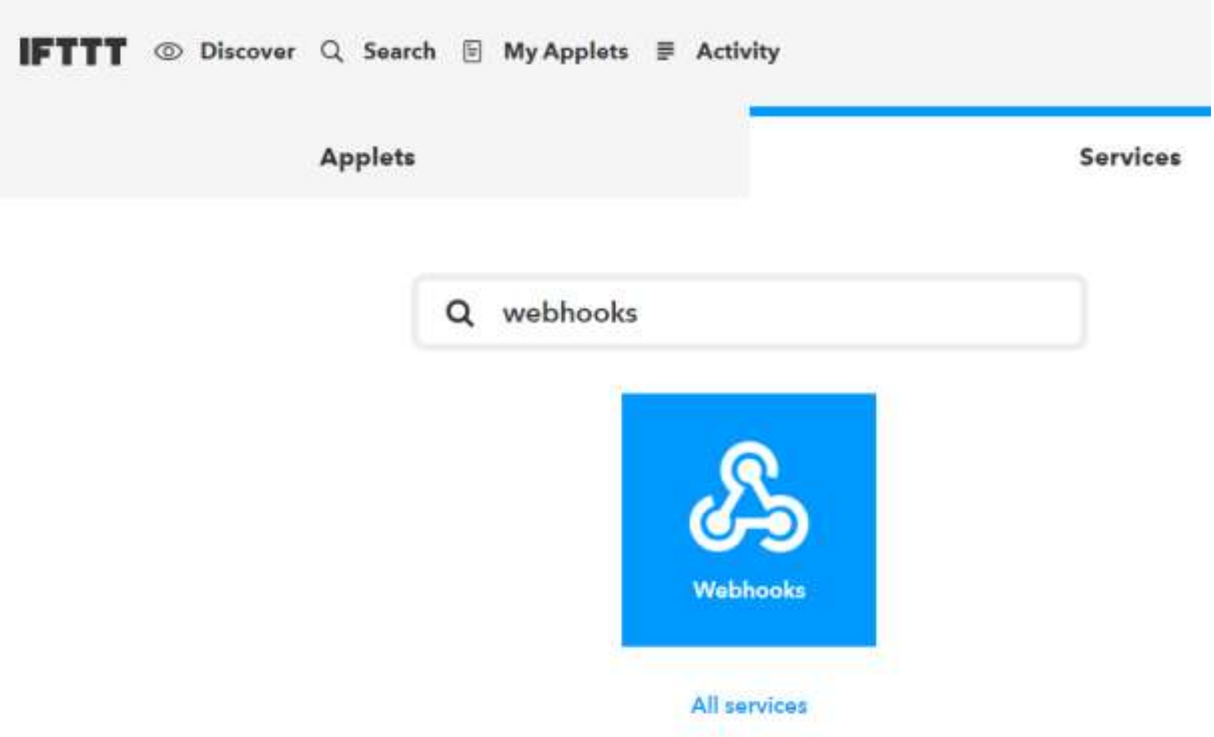
```
=DATEVALUE(REGEXREPLACE( "{{OccurredAt}}"," at .*","")) ||| =TIMEVALUE(REGEXREPLACE( "{{OccurredAt}}",".* at ","")) ||| {{EventName}} ||| {{Value1}} ||| {{Value2}} ||| {{Value3}}
```

Use "|||" to separate cells

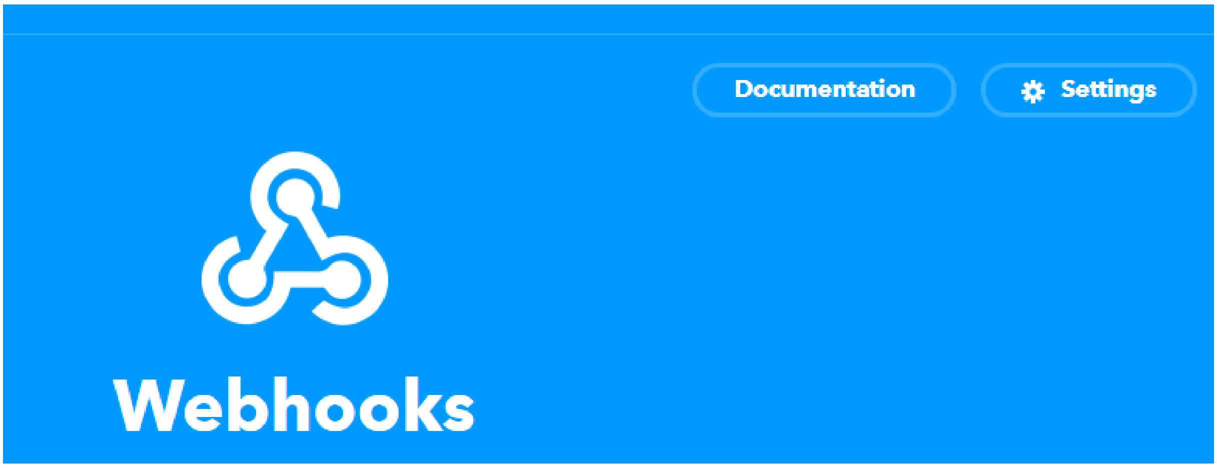
Then Save.

Step5 – remembering the WebHook API key

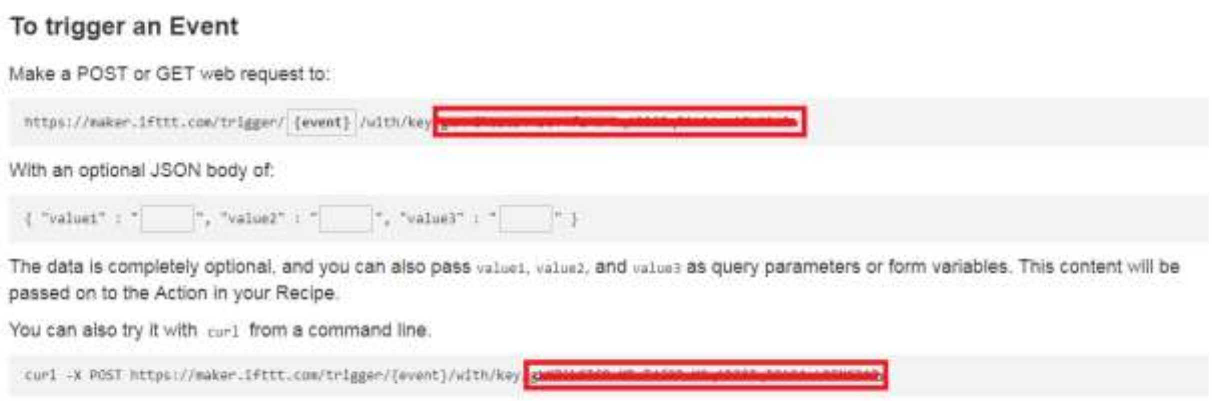
- Before going into ALTUI, we need to figure out what is the key that IFTTT allocated to use for the webhook. So Go again into the IFTTT Services screen and search for webhooks



Then click on it and click “**documentation**” on the following screen



Now you get into an important screen where you can see the webhook key that you need to use



To trigger an Event

Make a POST or GET web request to:

```
https://maker.ifttt.com/trigger/ {event} /with/key/[redacted]
```

With an optional JSON body of:

```
{ "value1" : "", "value2" : "", "value3" : "" }
```

The data is completely optional, and you can also pass `value1`, `value2`, and `value3` as query parameters or form variables. This content will be passed on to the Action in your Recipe.

You can also try it with `curl` from a command line.

```
curl -X POST https://maker.ifttt.com/trigger/{event}/with/key/[redacted]
```

Copy the key in the read area somewhere, you will need it later.

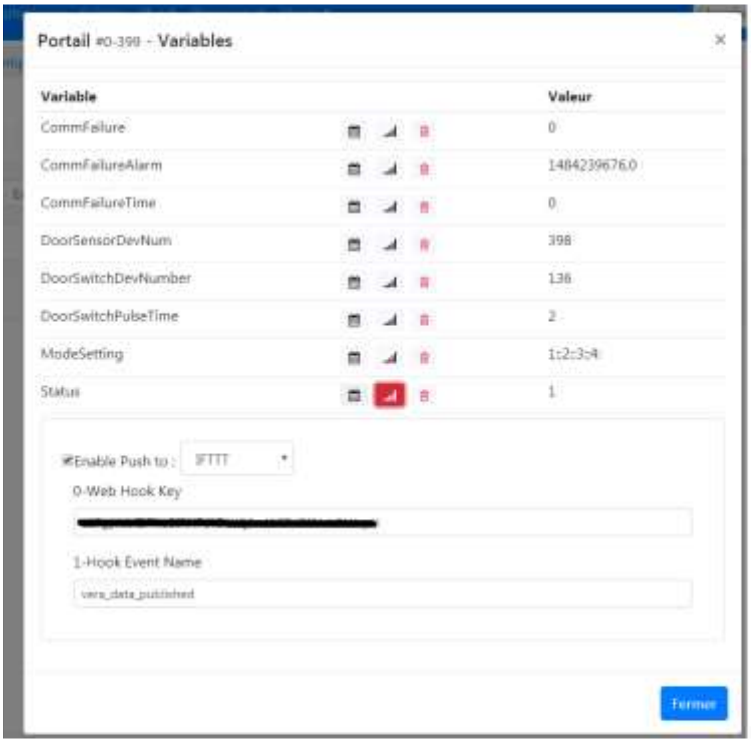
Configuration of ALTUI

Step1 – configuring the data push parameters

In ALTUI, the process is very simple. First you identify and go to the control panel of the device you want to use as the IFTTT trigger and click on the "histogram" icon to open the Data push properties.

- Here you select IFTTT
- You enter the Key that you have captured in the previous step
- You enter the event name that you have chosen when you configured your new webhook applet

NOTE. Do not forget to CLOSE THE HISTOGRAM icon to save the changes



Portail #0-398 - Variables

Variable	Valeur
CommFailure	0
CommFailureAlarm	1484239676.0
CommFailureTime	0
DoorSensorDevNum	398
DoorSwitchDevNumber	136
DoorSwitchPulseTime	2
ModeSetting	1:2:3:4
Status	1

☑Enable Push to : IFTTT

0-Web Hook Key

[redacted]

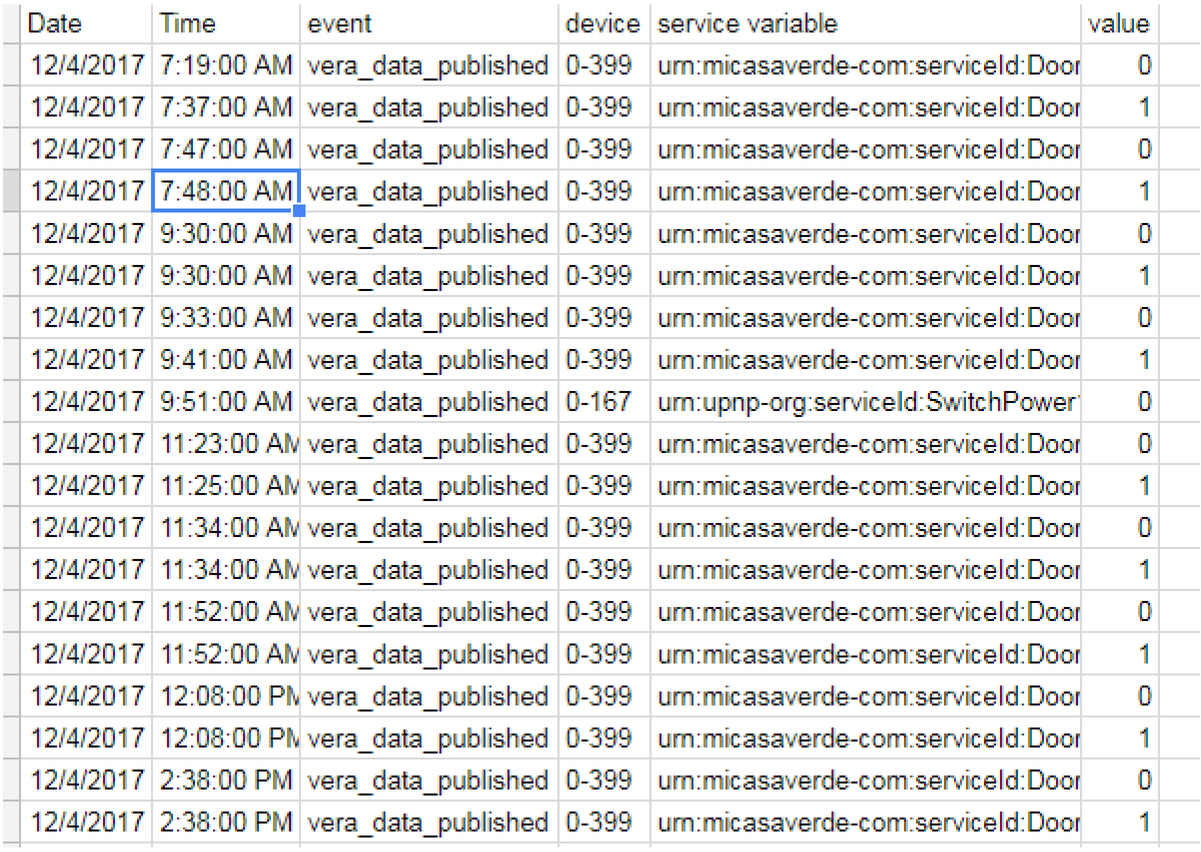
1-Hook Event Name

vera_data_published

Then click OK and **do a Luup Reload** to be sure all changes are taken into account.

From there on, any changes on this variable will trigger a web service to IFTTT which in turns will trigger your applet action, so for this example it will add a line in a google sheet

Here is the resulting sheet.



Date	Time	event	device	service variable	value
12/4/2017	7:19:00 AM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	0
12/4/2017	7:37:00 AM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	1
12/4/2017	7:47:00 AM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	0
12/4/2017	7:48:00 AM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	1
12/4/2017	9:30:00 AM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	0
12/4/2017	9:30:00 AM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	1
12/4/2017	9:33:00 AM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	0
12/4/2017	9:41:00 AM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	1
12/4/2017	9:51:00 AM	vera_data_published	0-167	urn:upnp-org:serviceId:SwitchPower	0
12/4/2017	11:23:00 AM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	0
12/4/2017	11:25:00 AM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	1
12/4/2017	11:34:00 AM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	0
12/4/2017	11:34:00 AM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	1
12/4/2017	11:52:00 AM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	0
12/4/2017	11:52:00 AM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	1
12/4/2017	12:08:00 PM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	0
12/4/2017	12:08:00 PM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	1
12/4/2017	2:38:00 PM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	0
12/4/2017	2:38:00 PM	vera_data_published	0-399	urn:micasaverde-com:serviceId:Door	1

Some additional Fine Tuning

Out of the sheet there is a nice enhancement needed so that we display the device name instead of just the device ID. We need a mapping table from device altuiid to device name.

Step1 – gather a table : device altuiid ⇔ device name

- For this we can go into ATLUI in the page More / Table Devices. Select ALTUIID & manufacturer, model and name fields in the field selector to show these columns. The “Name” is therefore the 4th field in each row

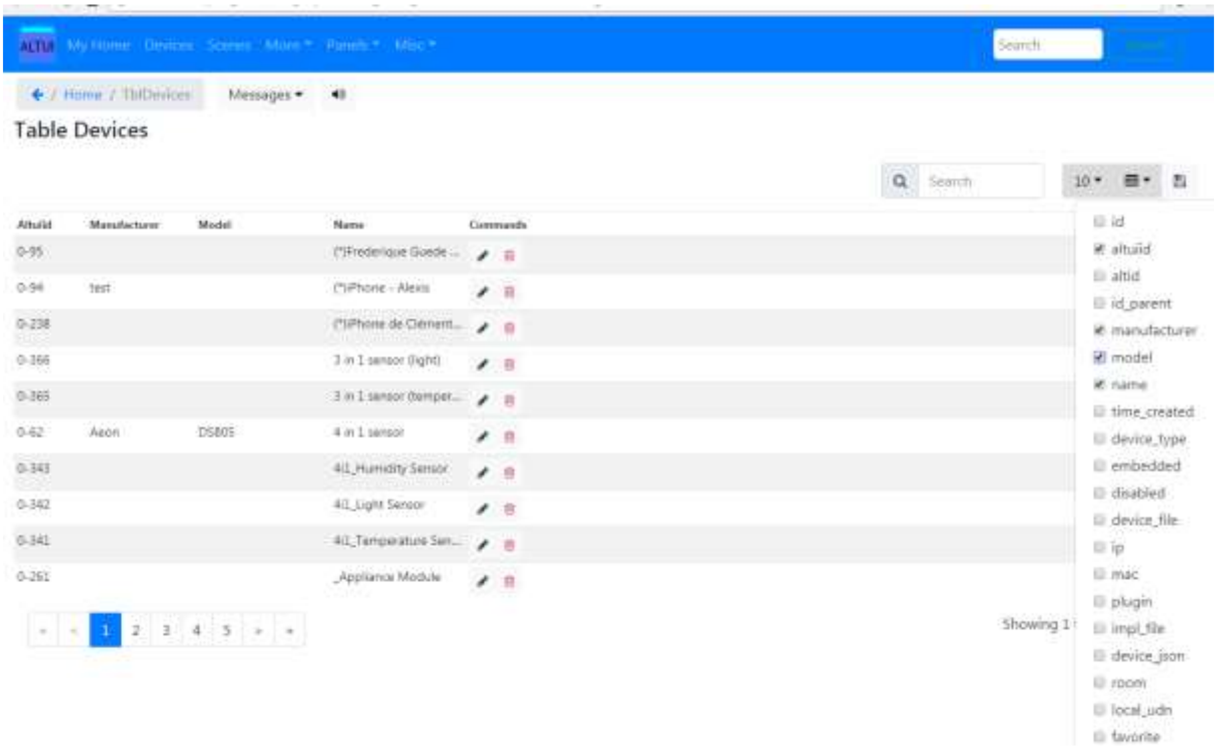


Table Devices

AltuiId	Manufacturer	Model	Name	Commands
0-95			(*)Frederique Guede ...	
0-94	test		(*)iPhone - Alexis	
0-238			(*)iPhone de Clement...	
0-366			3 in 1 sensor (light)	
0-365			3 in 1 sensor (temper...	
0-62	Aeon	DSB05	4 in 1 sensor	
0-343			4i1_Humidity Sensor	
0-342			4i1_Light Sensor	
0-341			4i1_Temperature Sen...	
0-261			_Appliance Module	

☐ id
☑ altuiid
☐ altid
☐ id_parent
☑ manufacturer
☑ model
☑ name
☐ time_created
☐ device_type
☐ embedded
☐ disabled
☐ device_file
☐ ip
☐ mac
☐ plugin
☐ impl_file
☐ device_json
☐ room
☐ local_udn
☐ favorite

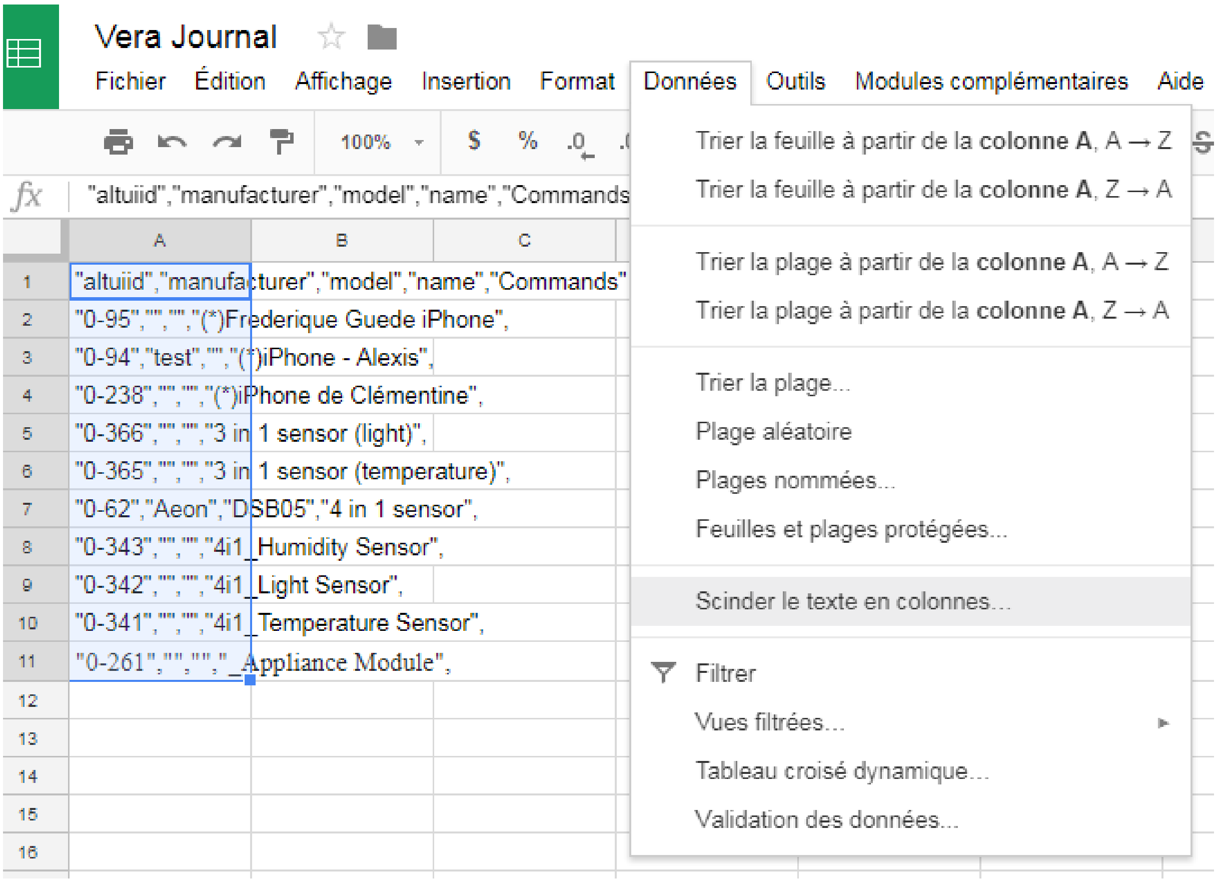
Then select "All Rows" in the batch size selector and click on the copy to clipboard button on the right.

Step2 - creating a mapping table in the sheet

- Then go into the google sheet file and add a second tab (must be second, IFTTT will fill in the first table always) , you can name it "Devices" for instance and select A1 cell then CTRL+V (paste). The following similar textual CSV data should be pasted in the cell.

```
"altuiid","manufacturer","model","name","Commands"
"0-94","test","","(*)iPhone - Alexis",
"0-366","","","3 in 1 sensor (light)",
"0-365","","","3 in 1 sensor (temperature)",
"0-62","Aeon","DSB05","4 in 1 sensor",
"0-343","","","4i1_Humidity Sensor",
"0-342","","","4i1_Light Sensor",
"0-341","","","4i1_Temperature Sensor",
"0-261","","","_Appliance Module",
```

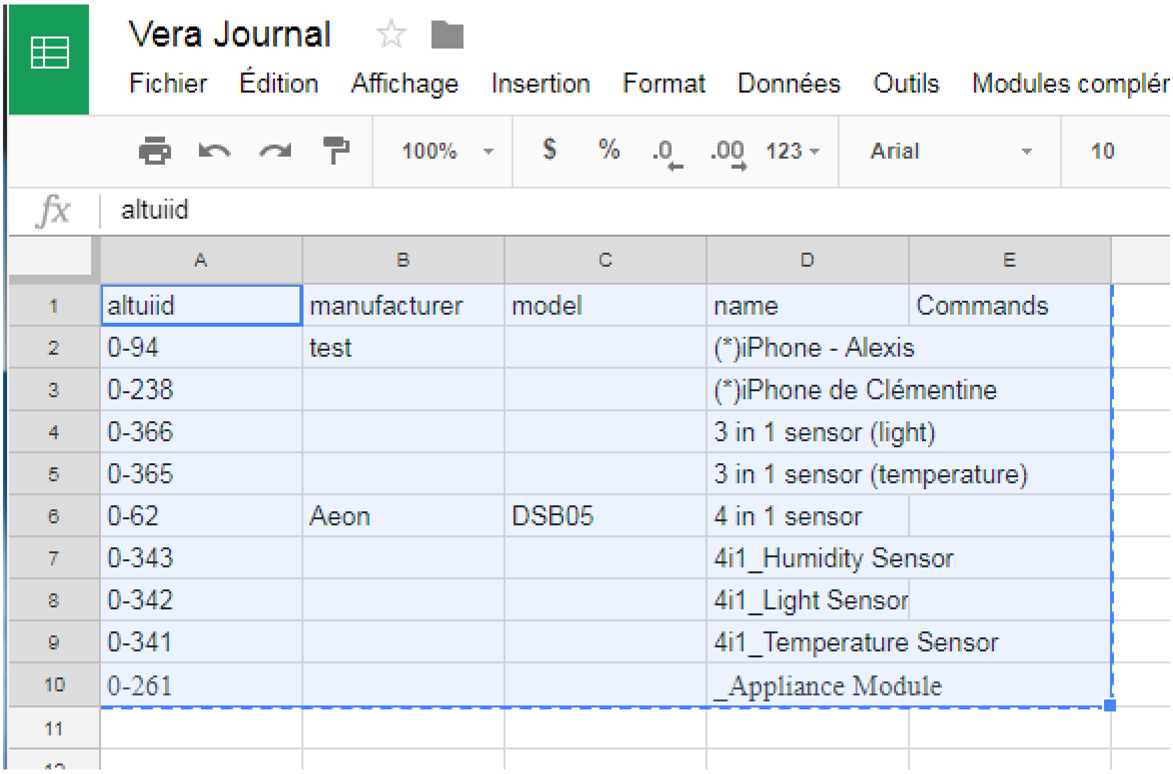
Then select the whole range of cells and go to **Data / Split text to columns command**


Vera Journal
Fichier
Édition
Affichage
Insertion
Format
Données
Outils
Modules complémentaires
Aide
100%
$
%
.0
Trier la feuille à partir de la **colonne A**, A → Z
Trier la feuille à partir de la **colonne A**, Z → A
Trier la plage à partir de la **colonne A**, A → Z
Trier la plage à partir de la **colonne A**, Z → A
Trier la plage...
Plage aléatoire
Plages nommées...
Feuilles et plages protégées...
Scinder le texte en colonnes...
Filtrer
Vues filtrées...
Tableau croisé dynamique...
Validation des données...

fx "altuiid","manufacturer","model","name","Commands

	A
1	"altuiid","manufacturer","model","name","Commands"
2	"0-95","","","(*)Frederique Guede iPhone",
3	"0-94","test","","(*)iPhone - Alexis",
4	"0-238","","","(*)iPhone de Clémentine",
5	"0-366","","","3 in 1 sensor (light)",
6	"0-365","","","3 in 1 sensor (temperature)",
7	"0-62","Aeon","DSB05","4 in 1 sensor",
8	"0-343","","","4i1_Humidity Sensor",
9	"0-342","","","4i1_Light Sensor",
10	"0-341","","","4i1_Temperature Sensor",
11	"0-261","","","_Appliance Module",
12	
13	
14	
15	
16	

That should give a nice table like this



fx altuiid

	A	B	C	D	E
1	altuiid	manufacturer	model	name	Commands
2	0-94	test		(*)iPhone - Alexis	
3	0-238			(*)iPhone de Clémentine	
4	0-366			3 in 1 sensor (light)	
5	0-365			3 in 1 sensor (temperature)	
6	0-62	Aeon	DSB05	4 in 1 sensor	
7	0-343			4i1_Humidity Sensor	
8	0-342			4i1_Light Sensor	
9	0-341			4i1_Temperature Sensor	
10	0-261			_Appliance Module	
11					

- Which means the table can now be used in a VLOOKUP() function so that , from a altuiid, we can find the device name. assuming this whole table is located in a tab called “devices” and it spans from A2 to D161, then the following function can give us the device name

```
=VLOOKUP(TRIM(<the altuiid given by the webhook>),
devices!A2:D161, 4, false)
```

Step3 – updating Google Sheet

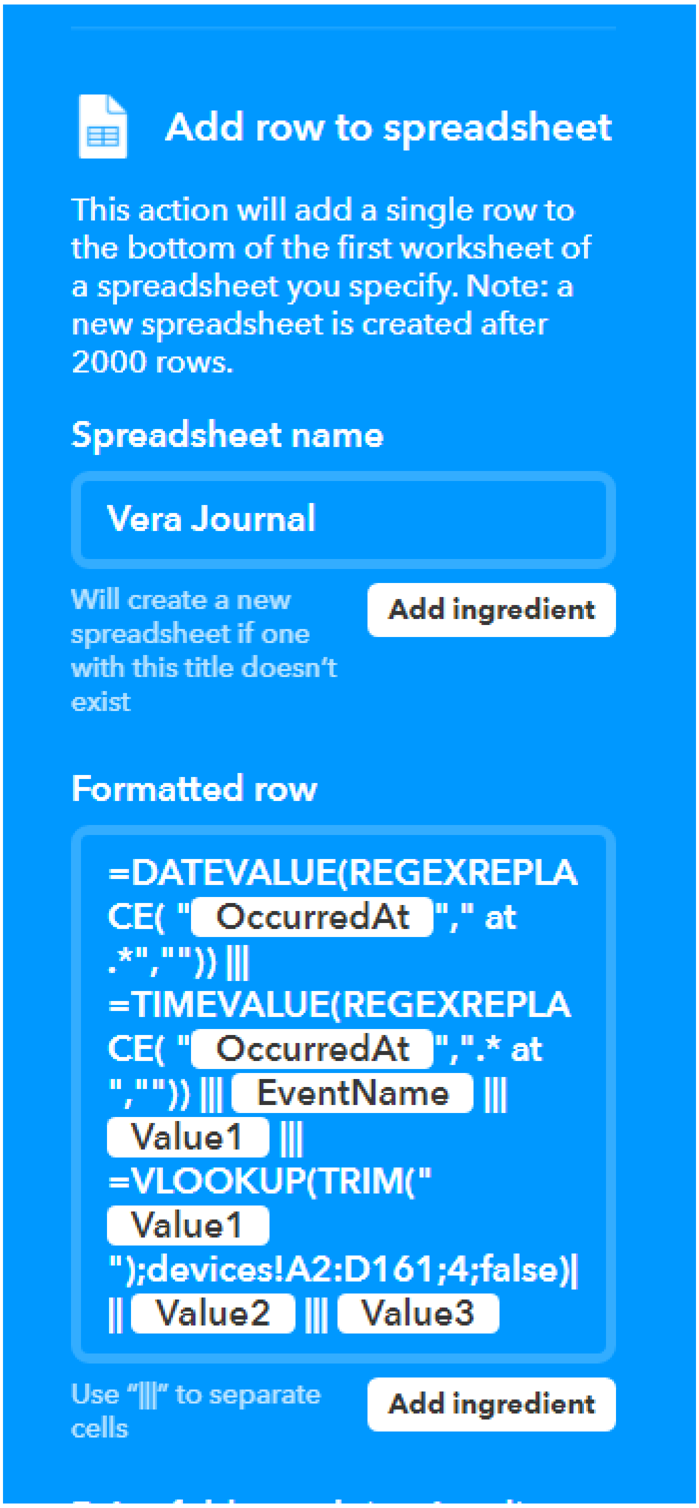
- Now we will have to make sure the IFTTT action is properly using formula, so we go back to IFTTT and edit our applet property this way. See the Formatted row value which includes a new field (separated by the ||| sign) and includes the formula

The whole formatted row is now:

- =DATEVALUE(REGEXREPLACE("{{OccurredAt}}"," at .*","")) ||| =TIMEVALUE(REGEXREPLACE("{{OccurredAt}}",".* at ","")) ||| {{EventName}} ||| {{Value1}} ||| =VLOOKUP(TRIM("{{Value1}}");devices!A2:D161;4;false)||| {{Value2}} ||| {{Value3}}

Which will produce these fields

1. A date
2. A time
3. The webhook event name
4. The altuiid of the device sending the event
5. The name of that device
6. The “service:variable” name
7. The new value taken by the variable



Add row to spreadsheet

This action will add a single row to the bottom of the first worksheet of a spreadsheet you specify. Note: a new spreadsheet is created after 2000 rows.

Spreadsheet name

Vera Journal

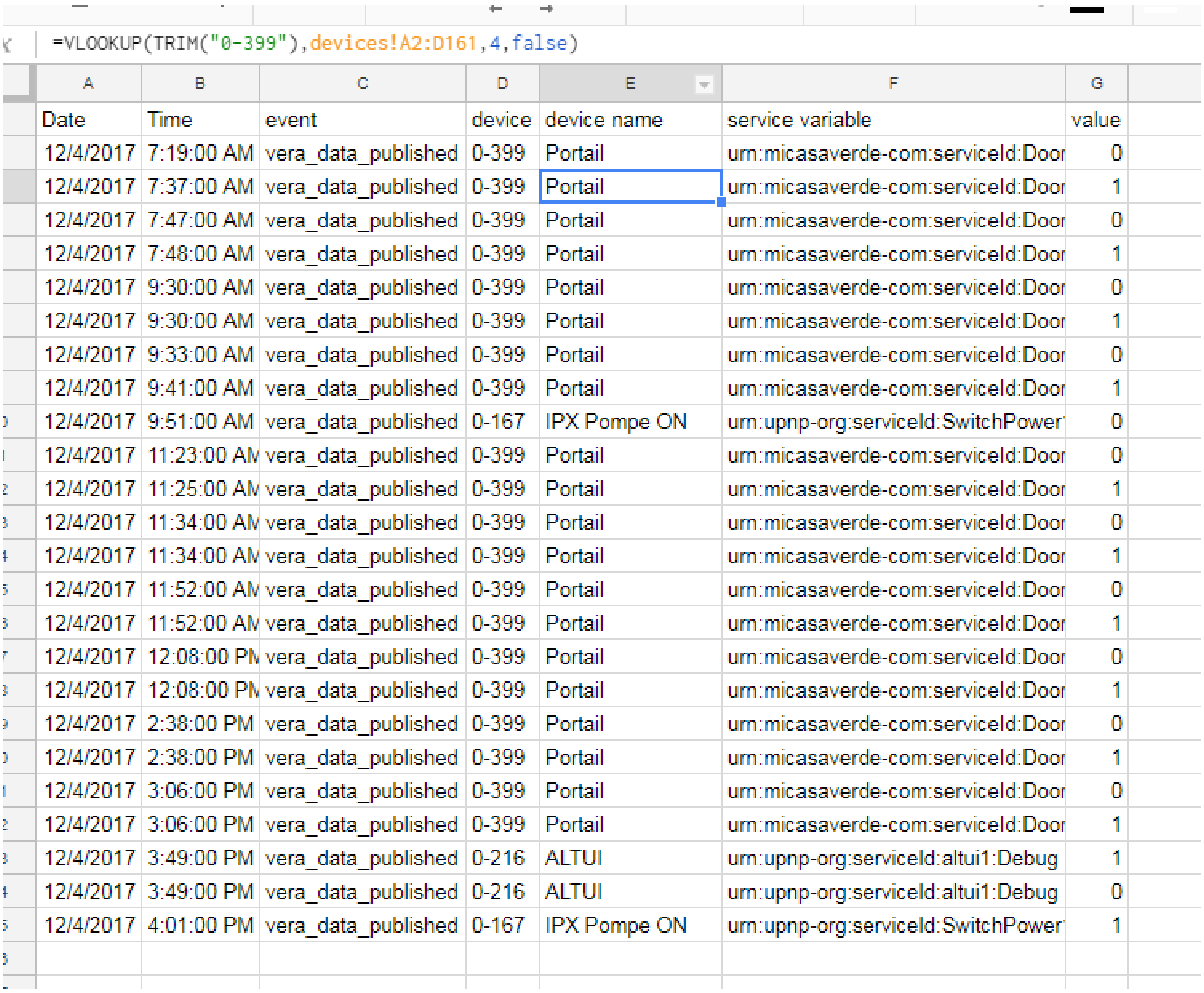
Will create a new spreadsheet if one with this title doesn't exist

Formatted row

```
=DATEVALUE(REGEXREPLACE( "OccurredAt"," at .*","")) ||| =TIMEVALUE(REGEXREPLACE( "OccurredAt",".* at ","")) ||| EventName ||| Value1 ||| =VLOOKUP(TRIM("Value1");devices!A2:D161;4;false)||| Value2 ||| Value3
```

Use “|||” to separate cells

For a result like this



```
=VLOOKUP(TRIM("0-399"),devices!A2:D161,4,false)
```

A	B	C	D	E	F	G
Date	Time	event	device	device name	service variable	value
12/4/2017	7:19:00 AM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	0
12/4/2017	7:37:00 AM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	1
12/4/2017	7:47:00 AM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	0
12/4/2017	7:48:00 AM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	1
12/4/2017	9:30:00 AM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	0
12/4/2017	9:30:00 AM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	1
12/4/2017	9:33:00 AM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	0
12/4/2017	9:41:00 AM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	1
12/4/2017	9:51:00 AM	vera_data_published	0-167	IPX Pompe ON	urn:upnp-org:serviceId:SwitchPower	0
12/4/2017	11:23:00 AM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	0
12/4/2017	11:25:00 AM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	1
12/4/2017	11:34:00 AM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	0
12/4/2017	11:34:00 AM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	1
12/4/2017	11:52:00 AM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	0
12/4/2017	11:52:00 AM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	1
12/4/2017	12:08:00 PM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	0
12/4/2017	12:08:00 PM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	1
12/4/2017	2:38:00 PM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	0
12/4/2017	2:38:00 PM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	1
12/4/2017	3:06:00 PM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	0
12/4/2017	3:06:00 PM	vera_data_published	0-399	Portail	urn:micasaverde-com:serviceId:Door	1
12/4/2017	3:49:00 PM	vera_data_published	0-216	ALTUI	urn:upnp-org:serviceId:altui1:Debug	1
12/4/2017	3:49:00 PM	vera_data_published	0-216	ALTUI	urn:upnp-org:serviceId:altui1:Debug	0
12/4/2017	4:01:00 PM	vera_data_published	0-167	IPX Pompe ON	urn:upnp-org:serviceId:SwitchPower	1

CONCLUSION

This is just one example of what can be accomplished with ALTUI IFTTT integration. The possibilities of IFTTT are enormous so the imagination is the limit. Have fun with it and feel free to report back interesting results