



## Application note

---

Local Display" script

# Introduction

This application note describes how to implement the "**local display**" script.

This script provides general information about the photovoltaic power plant:

- Instantaneous active power
- Total energy produced since the plant was commissioned
- Daily energy
- Self-consumption rate (requires the use of a meter)
- Total CO2 savings ;
- Daily CO2 savings.

This information can be regularly transmitted to one or more Modbus devices such as Siebert displays. The script does not require a display.

This information can also be made available by querying WebdynSunPM via Modbus or HTTP requests.

This script is available free of charge and its source code is accessible, so it can be freely adapted or modified.

The script is available in the WebdynSunPM internal library from version 5.0.10.

It can be downloaded from the following link: <https://www.webdyn.com/download/LocalDisplay.zip>

# Operating principle

Most of the information is generated by aggregating the individual information from each inverter. They must therefore be available in the data made available by the inverters.

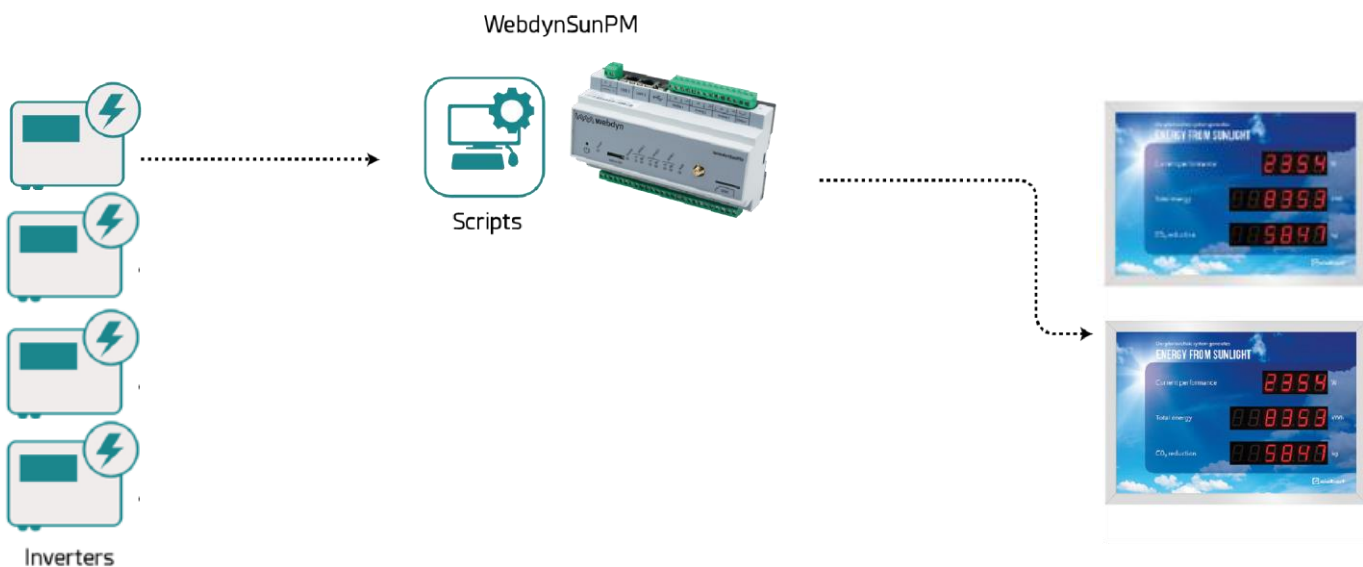
To calculate the self-consumption rate, a meter needs to be added in order to know the local consumption of the site.

Two options are available:

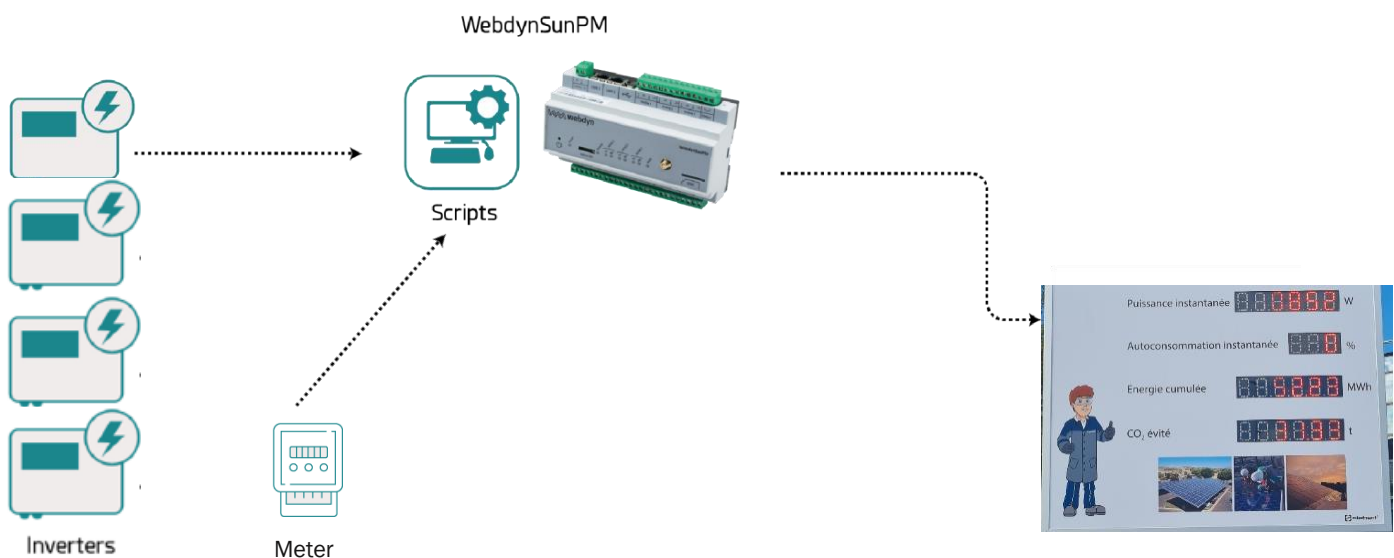
- A general meter at the point of injection into the network;
- In other words, a local consumption meter.

CO2 emission savings are calculated simply by applying a coefficient to the cumulative Energy data.

## Standard



## Own consumption rate





# Prerequisites

We recommend updating WebdynSunPM to the latest firmware version available.

The script is available in the WebdynSunPM script library from version 5.0.10.

However, it can be retrieved by following the link below and imported via the web interface or the server

: <https://www.webdyn.com/download/LocalDisplay.zip>

Knowledge of the basic principles of WebdynSunPM operation is strongly recommended.

Please refer to the WebdynSunPM user manual ([https://www.webdyn.com/wp-content/uploads/2024/10/WebdynSunPM-User-Manual\\_EN\\_V5.03.pdf](https://www.webdyn.com/wp-content/uploads/2024/10/WebdynSunPM-User-Manual_EN_V5.03.pdf)) for the following information:

- Chapter **§3.2.3.2.2.1 Adding a device** page 102
- Chapter **§3.1.2.2.2 Contents of the definition file** page 66
- Chapter **§3.2.4.1 Importing a service or licence** page 151
- Chapter **§3.1.2.1.4 "<UID>\_scl.ini" file** page 63
- Virtual devices and Modbus Slave

The settings described below for the inverter, counter and display definition files are already made in most of the files included in the WebdynSunPM internal library.

In such cases, using the script does not require any additional specific parameterisation of the definition files.

# Configuring inverters

The following elements are required in each definition file used by the inverters connected to the concentrator:

- **Category (equipment identification)**

In the definition file header, the category field (first line, 2nd column) must be defined with the name "Inverter". This name is used to identify all the inverters to be taken into account.

- **Tags (variable identification)**

All equipment identified by the "Inverter" category must have the following tags:

**RealPower**" tag Used to identify the variable containing the instantaneous active power.

**EnergyTotal**" tag: Used to identify the variable containing the energy produced by the inverter since it was commissioned.

**EnergyDay**" tag (depending on the inverter): Used to identify the variable containing the energy produced by the inverter since the start of the day.



Check that the units of the variables identified by the same tag are identical between inverters and consistent with the desired display.

**Tips:** You can use this script to aggregate variables other than those recommended above by cleverly placing the tags.

Tags" must be entered in column G (field 7) of the equipment definition file.

**Example:**

	A	B	C	D	E	F	G	H	I	J	K
1	modbusTCr	Inverter	HUAWEI	V4							
77	76	3	32078	I32		Active power peak of current day		0.001	0 kW		4
78	77	3	32080	I32		Active power	RealPower	1	0 W		4
79	78	3	32082	I32		Reactive_power		1	0 Var		4
80	79	3	32084	I16		Power factor		0.001	0		4
81	80	3	32085	U16		Frequency		0.01	0 Hz		4
82	81	3	32086	U16		Inverter efficiency		0.01	0 %		4
83	82	3	32087	I16		Cabinet temperature		0.1	0 °C		4
84	83	3	32088	U16		Insulation resistance		0.001	0 M/Ohm		4
85	84	3	32089	U16		Device status		1	0		9
86	85	3	32090	U16		Fault code		1	0		4
87	86	3	32091	U32		Startup time		1	0 s		4
88	87	3	32093	U32		Shutdown time		1	0 s		4
89	88	3	32106	U32		Energy_total	EnergyTotal	0.01	0 kWh		4
90	89	3	32114	U32		Energy yield of current day	EnergyDay	0.01	0 kWh		4
91	90	3	32116	U32		Energy yield of the month		0.01	0 kWh		4

# Setting the counter

Setting a meter is only necessary if you want to display a self-consumption rate.

To use the self-consumption rate calculation function, you need to edit the "local display" script in order to modify the **AutoconsoRatio** parameter on line 24.

```
23 local CarbCoef = 0.7
24 local AutoconsoRatio = 0
25
```

- **Naming (identification of meter type)**

To calculate the self-consumption rate, a meter needs to be added in order to know the local consumption of the site.

There are two possible cases:

- Or a general meter at the point of injection into the grid, to which we add the power from the inverters to obtain a local consumption value. In this case, the meter should be called **"MainMeter"**.
- Or a meter that directly supplies local consumption. In this case, the meter should be named **"ConsoMeter"**.

- **Tags (variable identification)**

In the meter definition file, you need to identify the power variable by assigning it the **"ActivePowSumkW"** tag, then check that it is expressed in the same unit as the power from the inverters, and adjust the A (gain) coefficient if necessary.

The "Tag" must be entered in column G (field 7) of the equipment definition file.

If necessary, refer to the equipment manufacturer's manual to identify the desired variable.



The power supplied by the meter must be the same as that

The sign convention used for meters should be as follows:

Positive values indicate withdrawal and negative values indicate injection.

Example:

meter" category

Tag of instantaneous power variables

	A	B	C	D	E	F	G	H	I	J	K
1	ModbusTCP	Meter	Janitza	Generic							
2	1	4	19000	F32		ULN[0]		1	0 V		4
3	2	4	19002	F32		ULN[1]		1	0 V		4
4	3	4	19004	F32		ULN[2]		1	0 V		4
5	4	4	19006	F32		ULL[0]		1	0 V		4
6	5	4	19008	F32		ULL[1]		1	0 V		4
7	6	4	19010	F32		ULL[2]		1	0 V		4
8	7	4	19012	F32		ILN[0]		1	0 A		4
9	8	4	19014	F32		ILN[1]		1	0 A		4
10	9	4	19016	F32		ILN[2]		1	0 A		4
11	10	4	19018	F32		I_SUM3		1	0 A		4
12	11	4	19020	F32		PLN[0]	ActivePow1kW	0.001	0 kW		4
13	12	4	19022	F32		PLN[1]	ActivePow2kW	0.001	0 kW		4
14	13	4	19024	F32		PLN[2]	ActivePow3kW	0.001	0 kW		4
15	14	4	19026	F32		P_SUM3	ActivePowSumkW	0.001	0 kW		4



# Display settings

In each definition file used by equipment that is to receive data calculated by the script, such as a Siebert display, the following elements are required:

- **Category (equipment identification)**

In the definition file header, the category field (first line, 2nd column) must be defined with the name "**Display**". This name is used to identify all the displays to be addressed.

- **Tags (variable identification)**

All equipment identified by the "**Display**" category must have the following tags:

**Pac-Tag**" tag Used to identify the variable that will receive the cumulative instantaneous active power of the inverters (identified by the **RealPower** tag).

**Eac-Tag**" tag: Used to identify the variable that will receive the total energy produced by the inverters since they were commissioned. (Identified by the **EnergyTotal** tag)

The following tags are optional:

**EacDay-Tag**" tag: Used to identify the variable that will receive the total energy produced by the inverters since the start of the day. (Identified by the **EnergyDay** tag). To use this tag, all the identified inverters must be able to supply their daily production and the **EnergyDay** tag must therefore be declared in each inverter definition file.

When **Auto-consumption** mode is used, it is the variable identified by this "**EacDay-Tag**" that receives the auto-consumption value.

**Carb-Tag**" tag: Used to identify the variable that will be used to calculate CO2 emission savings since the power plant was commissioned (value sent to **Eac-Tag** multiplied by the carbon coefficient, which is 0.7 by default). This value is often calculated directly by the inverter, in which case this tag should not be entered so that the value is not transmitted.

**CarbDay-Tag**" tag: Used to identify the variable that will receive the calculation of CO2 emission savings since the power plant was commissioned (value sent to **EacDay-Tag** multiplied by the carbon coefficient, which is 0.7 by default). This value is often calculated directly by the inverter, in which case this tag should not be entered so that the value is not transmitted.

Tags" must be entered in column G (field 7) of the equipment definition file.

It is not necessary for all the displays to have the same tags; each display can be configured with its own specific tags. However, it is not possible to have the Daily Energy on one display and the self-consumption rate on another.



## Example:

Display" category

	A	B	C	D	E	F	G	H	I	J	K
1	modbusR	U Display	3 EBERT	XC4XX	1						
2		1	3	1 U32_W		Puissance instantanée	Pac-Tag	1	0		4
3		2	3	3 U32_W		Energie journalière		1	0		4
4		3	3	5 U32_W		Energie cumulée	Eac-Tag	1	0		4
5		4	3	7 U32_W		Economie d'émission de CO2	Carb-Tag	1	0		4
6		5	3	9 U32_W		Economie d'émission de CO2 journalière	CarbDay-Tag	1	0		4

## Refresh rate and scaling coefficients.

The default refresh rate for equipment receiving data is **10s**.

This rate can be changed by editing the script and adjusting the value of the "DisplayRefreshFrequency" parameter on line 19.

```
19 local DisplayRefreshFrequency = 10
20 local Pac_coef = 1
21 local Eac_coef = 1
22 local EacDay_coef = 1
```

It is also possible to adjust the values of the data sent: Active Power, Cumulative Energy and Daily Energy by modifying their respective coefficients Pac\_coef (line 20), Eac\_coef (line 21) and EacDay\_coef (line 22).

It may be useful to modify these coefficients when using another script, such as the injection control script, which requires the use of a different unit to that required for the display.

This makes it impossible to use the coefficients in the inverter definition file.

This is an alternative solution to using the coefficients in the display definition file.

## Coefficient for calculating CO2 emissions

The coefficient applied to the Energy variables in order to calculate the CO2 savings can be entered as parameters in the script:

GenSet-V1_04	Generator	1.04	Missing/Invalid	Disabled	<input type="checkbox"/>	⋮
LocalDisplay	Local Display	8	Not required	Disabled	<input type="checkbox"/>	⋮
RelayControl	Relay Control	2.0	Not required	Disabled	<input type="checkbox"/>	⋮
SendCommand	Send Command	1.0	Not required	Disabled	<input type="checkbox"/>	⋮

Script arg

Script logs

View

It will then replace the default value of 0.7 which can also be changed directly in the script line 23 :

```
22 local EacDay_coef = 1
23 local CarbCoef = 0.7
24 local AutocosecRatio =
```

# Script

## Script loading

The script is available in the WebdynSunPM script library from version 5.0.10.

However, it can be retrieved by following the link below and imported via the web interface or the server

: <https://www.webdyn.com/download/LocalDisplay.zip>

It can be loaded via the embedded web interface by clicking on the Add script/licence file button.

LocalDisplay	Local Display	8	Not required	Disabled	<input type="checkbox"/>	⋮
RelayControl	Relay Control	2.0	Not required	Disabled	<input type="checkbox"/>	⋮
SendCommand	Send Command	1.0	Not required	Disabled	<input type="checkbox"/>	⋮

**Add script/licence file**

## Script settings

The carbon coefficient can be adjusted by changing the parameter (the default value is 0.7).

GenSet-V1_04	Generator	1.04	Missing/Invalid	Disabled	<input type="checkbox"/>	⋮
LocalDisplay	Local Display	8	Not required	Disabled	<input type="checkbox"/>	⋮
RelayControl	Relay Control	2.0	Not required	Disabled	<input type="checkbox"/>	⋮
SendCommand	Send Command	1.0	Not required	Disabled	<input type="checkbox"/>	⋮

**Script arg**  
Script logs  
View

**Add arguments**

Script arguments

Enter arguments here

## Starting the script

After activating the script using the button at the end of the line, you can access the script log:

LocalDisplay	Local Display	8	Not required	Enabled	<input checked="" type="checkbox"/>	⋮
--------------	---------------	---	--------------	---------	-------------------------------------	---

LocalDisplay	Local Display	8	Not required	Enabled	
RelayControl	Relay Control	2.0	Not required	D	Script arg
SendCommand	Send Command	1.0	Not required	D	Script logs
					View

```


2024-11-04 14:51:42 [LocalDisplay.lua 27] Local Display Script V8 Started
2024-11-04 14:51:42 [LocalDisplay.lua 177] No Carbon coefficient parameter, Using default value 0.7
2024-11-04 14:51:42 [LocalDisplay.lua 43] No change on template WPM0117B5_Script_LocalDisplay.csv
2024-11-04 14:51:42 [LocalDisplay.lua 68] 3 inverters found
2024-11-04 14:51:42 [LocalDisplay.lua 73] Inverter 1(INV2) has tag: RealPower
2024-11-04 14:51:42 [LocalDisplay.lua 80] Inverter 1(INV2) has tag: EnergyTotal
2024-11-04 14:51:42 [LocalDisplay.lua 87] Inverter 1(INV2) has tag: EnergyDay
2024-11-04 14:51:42 [LocalDisplay.lua 73] Inverter 2(INV3) has tag: RealPower
2024-11-04 14:51:42 [LocalDisplay.lua 80] Inverter 2(INV3) has tag: EnergyTotal
2024-11-04 14:51:42 [LocalDisplay.lua 87] Inverter 2(INV3) has tag: EnergyDay
2024-11-04 14:51:42 [LocalDisplay.lua 73] Inverter 0(INV1) has tag: RealPower
2024-11-04 14:51:42 [LocalDisplay.lua 80] Inverter 0(INV1) has tag: EnergyTotal
2024-11-04 14:51:42 [LocalDisplay.lua 87] Inverter 0(INV1) has tag: EnergyDay
2024-11-04 14:51:42 [LocalDisplay.lua 108] 1 Displays found
2024-11-04 14:51:42 [LocalDisplay.lua 112] Display0(Afficheur) has tag: Pac-Tag
2024-11-04 14:51:42 [LocalDisplay.lua 118] Display0(Afficheur) has tag: Eac-Tag
2024-11-04 14:51:42 [LocalDisplay.lua 127] Display0(Afficheur) no tag: EacDay-Tag
2024-11-04 14:51:42 [LocalDisplay.lua 134] Display0(Afficheur) no tag: Carb-Tag
2024-11-04 14:51:42 [LocalDisplay.lua 141] Display0(Afficheur) no tag: CarbDay-Tag

```

When the script starts up, it checks that the equipment and tags required for operation are present. If a tag is missing, the script will display an error message:

LocalDisplay	Local Display	8	Not required	Error		
--------------	---------------	---	--------------	-------	---	---

And the logs show the missing tag(s).

 Logs

```

2024-11-04 13:49:53 [LocalDisplay.lua 27] Local Display Script V8 Started
2024-11-04 13:49:53 [LocalDisplay.lua 177] No Carbon coefficient parameter, Using default value 0.7
2024-11-04 13:49:53 [LocalDisplay.lua 43] Creation or update of template WPM0117B5_Script_LocalDisplay.csv
2024-11-04 13:49:53 [LocalDisplay.lua 68] 3 inverters found
2024-11-04 13:49:53 [LocalDisplay.lua 75] Inverter 1(INV2) missing tag RealPower
2024-11-04 13:49:53 [LocalDisplay.lua 80] Inverter 1(INV2) has tag: EnergyTotal
2024-11-04 13:49:53 [LocalDisplay.lua 87] Inverter 1(INV2) has tag: EnergyDay
2024-11-04 13:49:53 [LocalDisplay.lua 75] Inverter 2(INV3) missing tag RealPower
2024-11-04 13:49:53 [LocalDisplay.lua 80] Inverter 2(INV3) has tag: EnergyTotal
2024-11-04 13:49:53 [LocalDisplay.lua 87] Inverter 2(INV3) has tag: EnergyDay
2024-11-04 13:49:53 [LocalDisplay.lua 75] Inverter 0(INV1) missing tag RealPower
2024-11-04 13:49:53 [LocalDisplay.lua 80] Inverter 0(INV1) has tag: EnergyTotal
2024-11-04 13:49:53 [LocalDisplay.lua 87] Inverter 0(INV1) has tag: EnergyDay
2024-11-04 13:49:53 [LocalDisplay.lua 104] No category with name Display
2024-11-04 13:49:53 [LocalDisplay.lua 106] No display screen declared
2024-11-04 13:49:53 [LocalDisplay.lua 167] Config error
2024-11-04 13:49:53 [LocalDisplay.lua 168] STOP

```

If the script runs correctly, the logs show the values calculated by the script and to whom they are sent.

```

2024-11-04 14:56:49 [LocalDisplay.lua 305] *****
2024-11-04 14:57:00 [LocalDisplay.lua 265] SendtoDisplay
2024-11-04 14:57:00 [LocalDisplay.lua 282] UpdateModbusSlave
2024-11-04 14:57:00 [LocalDisplay.lua 296] Instant Power: 20000 kW
2024-11-04 14:57:00 [LocalDisplay.lua 297] Total Energy: 199.99999552965 kWh
2024-11-04 14:57:00 [LocalDisplay.lua 301] Day Energy: 299.99999329448 kWh
2024-11-04 14:57:00 [LocalDisplay.lua 303] Co2 total savings: 139.99999687076 Co2
2024-11-04 14:57:00 [LocalDisplay.lua 304] Co2 Day savings: 209.99999530613 Co2
2024-11-04 14:57:00 [LocalDisplay.lua 305] *****
2024-11-04 14:57:12 [LocalDisplay.lua 265] SendtoDisplay
2024-11-04 14:57:12 [LocalDisplay.lua 282] UpdateModbusSlave
2024-11-04 14:57:12 [LocalDisplay.lua 296] Instant Power: 20000 kW
2024-11-04 14:57:12 [LocalDisplay.lua 297] Total Energy: 199.99999552965 kWh
2024-11-04 14:57:12 [LocalDisplay.lua 301] Day Energy: 299.99999329448 kWh
2024-11-04 14:57:12 [LocalDisplay.lua 303] Co2 total savings: 139.99999687076 Co2
2024-11-04 14:57:12 [LocalDisplay.lua 304] Co2 Day savings: 209.99999530613 Co2
2024-11-04 14:57:12 [LocalDisplay.lua 305] *****
2024-11-04 14:57:23 [LocalDisplay.lua 265] SendtoDisplay
2024-11-04 14:57:23 [LocalDisplay.lua 282] UpdateModbusSlave
2024-11-04 14:57:23 [LocalDisplay.lua 296] Instant Power: 20000 kW
2024-11-04 14:57:23 [LocalDisplay.lua 297] Total Energy: 199.99999552965 kWh
2024-11-04 14:57:23 [LocalDisplay.lua 301] Day Energy: 299.99999329448 kWh
2024-11-04 14:57:23 [LocalDisplay.lua 303] Co2 total savings: 139.99999687076 Co2
2024-11-04 14:57:23 [LocalDisplay.lua 304] Co2 Day savings: 209.99999530613 Co2
2024-11-04 14:57:23 [LocalDisplay.lua 305] *****

```

**UpdateModbusSlave** indicates that the values of the Modbus slave registers are updated

**SendToDisplay** indicates that the values are sent to the displays in the "Display" category. If no display is declared, this line does not appear.

## Virtual Device

It is possible to associate a definition file with a script, in which case the script behaves like a virtual device.

This is the case for the "LocalDisplay" script, for which the following variables have been defined:

```

6  --*****
7  -- Virtual Device declaration
8  --*****
9  virtualDevice = {
10     U32_Pac = 0,
11     U32_Eac = 0,
12     U32_EacDay = 0,
13     U32_Carb = 0,
14     U32_CarbDay = 0
15 }
16 --*****
17 local VirtualDevice_Enable = 1
18 local VirtualDevice_Frequency = 0

```

This declaration generates a definition file: **WPMxxxxx\_Script\_LocalDisplay.csv** containing these variables, which will be uploaded to the FTP server in the /DEF directory.

The contents of this file are as follows:

```

1 none;Script;Script;LocalDisplay
2 1;;;U32;;Carb;Carb;1.000000;0.000000;;4
3 2;;;U32;;CarbDay;CarbDay;1.000000;0.000000;;4
4 3;;;U32;;Eac;Eac;1.000000;0.000000;;4
5 4;;;U32;;EacDay;EacDay;1.000000;0.000000;;4
6 5;;;U32;;Pac;Pac;1.000000;0.000000;;4
7

```

These variables can be sent to WebdynSunPM's standard data files so that they can be used by the portal in the same way as supervised equipment.

To do this, you need to enter the recording frequency for these variables, which is independent of the data refresh rate of 10s. By default, a frequency of 0 does not write to the data file.

```

17 local VirtualDevice_Enable = 1
18 local VirtualDevice_Frequency = 0

```

This virtual device definition file is also used to enter the values calculated by the script as input to the Modbus slave (see Modbus slave file below).

### Modbus slave settings

See section 3.2.5.1.4 Modbus Slave in the WebdynSunPM user manual.

The Modbus slave functionality is activated from the daq.csv file in the /CONFIG directory on the remote server, as described in the example file below.

	A	B	C	D	E	F	G	H
1	type	pin	apn	login	password	authenticatic	server	dns
11	SERIAL1	9600		8 N		1	2 modbusRTU	0
12	SERIAL2	9600		8 N		1	2 modbusRTU	0
13	SERIAL3	9600		8 N		1	2 modbusRTU	0
14								
15	ModbusSlave	1		502 WPM01375A	ModbusSlave.csv			
16								

### Modbus slave file

The Modbus slave definition file must be placed on the FTP server in the /DEF directory and named as declared in the daq.csv file.

Below is an example of how to declare script data in the Modbus slave.

	A	B	C	D	E	F	G	H	I	J	K
1	ModbusSlave										
2	1	0_4		LocalDisplay		Pac		1	0		4
3	2	2_4		LocalDisplay		Eac		1	0		4
4	3	4_4		LocalDisplay		EacDay		1	0		4
5	4	6_4		LocalDisplay		Carb		1	0		4
6	5	8_4		LocalDisplay		CarbDay		1	0		4

It is perfectly possible to declare these variables with different register addresses and to mix them with other virtual or non-virtual devices.

# Other methods of recovering script data

## By request post

It is possible to retrieve the active power and total energy values of the inverters using a POST request via an application such as POSTMAN (<https://web.postman.co/>).

Note: All the functions accessible in the script are accessible via a post request.

- Open a session on the webservice enter the POST request: **shttp://<@IP\_WebdynSunPm>/auth** ;
- In the "Body" tab, enter: { "user": "userhigh", "password": "high" } ;
- Click on the Send tab.

Once the message has been sent, a cookie is retrieved and will only last for one minute if no command is sent to WebdynSunPM.

Opening a session with a POST request :

The screenshot displays the Postman interface for a POST request to `http://172.20.21.30/auth`. The request body is a JSON object: `{ "user": "userhigh", "password": "high" }`. The response is a `200 OK` status with a response time of `35 ms` and a body size of `223 B`. The response body is a single JSON object: `{ "KCOwQIDTHIKTRFC" }`.

To retrieve data from the display :

- Send the request post  
`http://<@IP_WebdynSunPm>/scripts?<Script_file_name>.<Function_name>` ;
- Click on the Send tab.

Post query to obtain cumulative energy :

The screenshot shows a web browser's developer tools interface. At the top, the URL bar displays `http://172.20.21.30/scripts?LocalDisplay.GetEnergy`. Below the URL bar, the request method is set to **POST** and the request body is empty. The **Send** button is visible. The **Body** tab is selected, showing a response of `1`. The status bar at the bottom indicates a **200 OK** response with a response time of **26 ms** and a size of **207 B**. The response is displayed in **JSON** format.



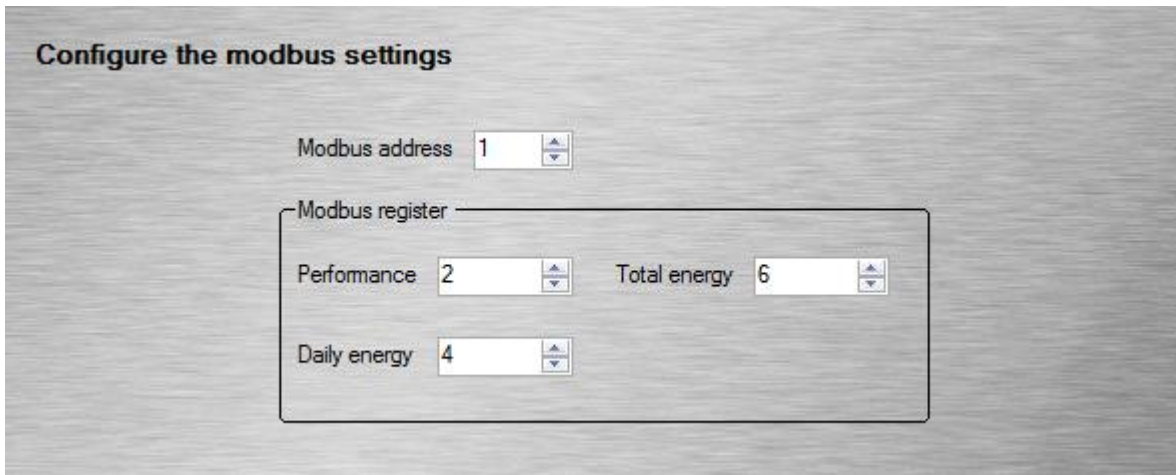
# Appendix 1: Siebert display

This script is often used with Siebert displays.

These displays are configured using Siebert's "SolarDisplayConfigurator" software. This requires the display to be connected to a computer running this utility via a USB/RS232 cable. The utility will then guide you step by step. It is advisable to remember the address allocated to the equipment and the RS485 link settings, so that you can transfer this information to WebdynSunPM when you declare the display.

Default :  
Speed: 19200 Bauds  
Parity: None  
Address: 1

The definition file for the Siebert display integrated into the WebdynSunPM library corresponds to the default declaration below (note that there is a one-register offset)  
Any modification will require adaptation of the definition file.



**Configure the modbus settings**

Modbus address: 1

Modbus register:

- Performance: 2
- Total energy: 6
- Daily energy: 4



These displays do not allow you to read their registers, which are write-only. They therefore appear in red in the WebdynSunPM Dashboard.

To test communication with these displays before starting the script, it may be useful to use the "**sendCommand**" script as described below.

## Display test :

The Send\_Command.lua script is very easy to use, you just need to enter :  
The name with which the equipment to be written to is declared, for example:  
Display.  
The tag of the variable to write to, for example: Pac\_tag.  
The value you wish to display, for example :526  
In the form: Display; Pac\_tag ;526