
domogik-plugin-script

Release 1.0

Aug 25, 2018

Contents

1	Plugin Script	1
1.1	Purpose	1
1.2	Dependencies	1
1.3	Plugin configuration	1
1.4	Create the domogik devices	1
1.5	Start the plugin	5
1.6	Set up your widgets on the user interface	5
1.7	Write a script	5
1.8	Troubleshooting	6
2	Development informations	9
2.1	xPL messages	9
3	Changelog	11
3.1	2.0	11
3.2	1.0	11
3.3	0.2	11
3.4	0.1	12

1.1 Purpose

The Script plugin is used to run some Linux scripts/command lines with Domogik.

The plugin can execute any **executable** script or program (shell, python, perl . . .) or compiled binary programs.

1.2 Dependencies

No dependencies needed

1.3 Plugin configuration

No configuration needed for the plugin

1.4 Create the domogik devices

1.4.1 Domogik device type : Script Action

This is a **command** device that will execute a script or command line. The usage is to run something.

This is a trigger command : there is no status (like on/off) and the command line to run will always be the same.

You need to configure this parameter :

Key	Type	Description
command	string	The command line to run when the command is triggered.

Example command line that can be run (for setting lights of the living room for example):

```
$ ~/bin/setLum.sh salon on
```

In this case, please notice that you will have to create another device to set the lights off. Example

```
$ ~/bin/setLum.sh salon off
```

1.4.2 Domogik device type : Script Switch

This is a **command** device that will execute a script or command line. The usage is to execute a switch.

This is a On/Off command.

You need to configure this parameters :

Key	Type	Description
command	string	The command line to run when the command is On.
command0	string	The command line to run when the command is Off.

One device is needed to set lights of the living room with this 2 commands in parameters:

```
$ ~/bin/setLum.sh salon on  
$ ~/bin/setLum.sh salon off
```

1.4.3 Domogik device type : Script String

This is a **command** device that will execute a script or command line. The usage is to run something.

This is a trigger command with one or two string parameter and the command line to run will always be the same.

You need to configure this parameter :

Key	Type	Description
command	string	The command line to run when the command is triggered.

Example command line that can be run with 2 string parameters:

```
$ ~/bin/sendmail.sh "title" "message"
```

In this case, one parameter is use

```
$ ~/bin/display.sh "message"
```

1.4.4 Domogik device type : Script Info State

This is a **sensor** device that will execute a script or command line each N seconds. The sensor value will be the value displayed by the script or command line.

The value is a **boolean**. The value displayed by the script or command line can be 0 or 1.

Key	Type	Description
interval	number	The time in seconds between two executions of the command line
command	string	The command line to run when the command is triggered.

Warning: The shell script of a **Script Info Binary device** should only return **'0'/'1'** with or without a ending **'\n'** otherwise you will get an error.

Example script that can be run (to get the status of a process for example):

```
$ ~/bin/check_process.sh lcdmeteo.py
1
```

1.4.5 Domogik device type : Script Info Opening

This is a **sensor** device that will execute a script or command line each N seconds. The sensor value will be the value displayed by the script or command line.

The value is a **OpenClose** type. The value displayed by the script or command line can be 0 or 1.

Key	Type	Description
interval	number	The time in seconds between two executions of the command line
command	string	The command line to run when the command is triggered.

Warning: The shell script of a **Script Info Binary device** should only return **'0'/'1'** with or without a ending **'\n'** otherwise you will get an error.

Example script that can be run (to get the status of a process for example):

```
$ ~/bin/check_windows.sh kitchen
1
```

1.4.6 Domogik device type : Script Info Value

This is a **sensor** device that will execute a script or command line each N seconds. The sensor value will be the value displayed by the script or command line.

The value is a **number** or **float**. The value displayed by the script or command line can be for example : 3, 43.2, ...

Key	Type	Description
interval	number	The time in seconds between two executions of the command line
command	string	The command line to run when the command is triggered.

Warning: The shell script of a **Script Info Number device** should only return a **number** with or without a ending **'\n'** otherwise you will get an error.

Example script that can be run (to get the power consumption of the day for example):

```
$ ~/bin/getElec.sh -jour
0.270
```

1.4.7 Domogik device type : Script Info Temperature

This is a **sensor** device that will execute a script or command line each N seconds. The sensor value will be the value displayed by the script or command line.

The value is a **Temperature** type. The value displayed by the script or command line can be for example : 3, 43.2, ...

Key	Type	Description
interval	number	The time in seconds between two executions of the command line
command	string	The command line to run when the command is triggered.

Warning: The shell script of a **Script Info Number** device should only return a **number** with or without a ending ‘n’ otherwise you will get an error.

Example script that can be run (to get the power consumption of the day for example):

```
$ ~/bin/getTempRPI
39.5
```

1.4.8 Domogik device type : Script Info String

This is a **sensor** device that will execute a script or command line each N seconds. The sensor value will be the value displayed by the script or command line.

The value is a **string**. The value displayed by the script or command line can be whatever you want, but it must use only one line!

Key	Type	Description
interval	number	The time in seconds between two executions of the command line
command	string	The command line to run when the command is triggered.

Example script that can be run (to get Weather warning for a location for example):

```
$ curl -s http://domogeek.entropialux.com/vigilance/78/color
vert
```

1.4.9 Additionnal informations about the devices creation

If you have some issues with a script, please check the plugin logs in `/var/log/domogik/plugin_script.log` (for a default Domogik configuration).

Note: You can disable a script by setting the interval to ‘-1’.

Warning: Special characters like '>', '<', '|', '&' are not authorized in **command** fields.

1.5 Start the plugin

You can now start the plugin (start button) and use the created domogik devices.

1.6 Set up your widgets on the user interface

You can now place the widgets of your devices features on the user interface.

1.7 Write a script

Note: In case of error, please check the **Troubleshooting** chapter at the end of this documentation page.

For all scripts, there are several rules to respect :

First, the script must begin with a **shebang**, for example :

```
#!/bin/bash
```

A shebang is an important information because it will tell the system which language interpreter must be used (bash, python, etc).

Here are some shebang examples :

- `#!/bin/bash : shell script`
- `#!/usr/bin/python : python script`

If you don't put a shebang in your script, it won't be executed and you will find an **Exec format error** in the plugin log file.

Then, the script must send a return code **0**.

1.7.1 Script/commands examples

There is a repository with scripts examples to give some ideas or use them. This repository is available here (thanks Tikismoke): <https://github.com/tikismoke/domogik-example-script>

Warning: CAUTION: The scripts provided on this website are given as such for example, no guarantee of proper operation in all cases and that the use is at your risk and your responsibility.

1.7.2 Special rules depending on the script usage

Sensor scripts

They must return a value corresponding to the type of the script (Number, Binary, String), Otherwise the execution will be **failed**.

Here are some script examples.

A **value** script that returns the value 32.4:

```
#!/bin/bash
echo 32.4
exit 0
```

A **string** script that returns the value 'hello':

```
#!/bin/bash
echo "hello"
exit 0
```

A **state** script that returns the status true:

```
#!/bin/bash
echo 1
exit 0
```

A **state** script that returns the status false:

```
#!/bin/bash
echo 0
exit 0
```

1.8 Troubleshooting

If a plugin script is not executed, please search for the keyword **ERROR** in the plugin log like this :

```
$ grep "ERROR" /var/log/domogik/xplplugin_script.log
```

Here are some ERROR examples:

- The sensor script must return the expected value type (Number, Boolean (binary), String), otherwise this ERROR will occur :

```
2015-12-19 00:05:55,231 domogik-script ERROR ### Script type Number '/home/USER/
↳bin/listdatawatermysql.sh -jour' not return a number: 'NULL'
```

- Don't use special characters like '>', '<', '|', '&', otherwise this ERROR will occur :

```
2015-12-19 00:30:07,310 domogik-script ERROR ### Script '/home/USER/bin/test.sh
↳"canape on" > /dev/null' is refused: specials characters like '>', '<', '|', '&'
↳are not authorized
```

- Script must be executable for the user who runs it, otherwise this ERROR will occur :

```
2015-12-19 00:35:20,835 domogik-script ERROR ### Script '/home/USER/bin/
↳notexecutable.sh' failed with OError : 13, (Permission denied)
```

- Yes, the script must exist in the right path, otherwise this ERROR will occur :

```
2015-12-19 00:35:37,887 domogik-script ERROR ### Script '/home/USER/bin/notexist.
↳sh' failed with OError : 2, (No such file or directory)
```

- This error might appear when the 'shebang' at top of the script is missing or the binary program is in bad arch (ARM for x86 or the other way around) :

```
2016-01-01 17:17:01,148 domogik-script ERROR ### Script '/home/dan/bin/noshebang.
↳sh' failed with OError : 8, (Exec format error)
2016-01-01 17:17:01,148 domogik-script ERROR ### Script '/home/dan/bin/noshebang.
↳sh': missing 'shebang' at top of the script or bad arch for binary program !
```


2.1 xPL messages

2.1.1 xpl-cmnd

The **exec.basic** message is used:

```
exec.basic { type='cmd_action'  program='/path/program [parameters]'  status='start' }
```

2.1.2 xpl-trig

The answer of the xpl-cmnd.

The **exec.basic** message is used:

```
exec.basic { type='script_info_number | script_info_binary | script_action'  program=  
↪ '/path/program [parameters]'  status='executed|value|failed' }
```

2.1.3 Examples of xPL messages of the plugin

The **script_action** xpl-cmnd message:

```
xpl-cmnd  exec.basic { program='/home/USER/bin/setchacon.sh salon on' type='script_  
↪ action' status='start' }
```

The **script_action** xpl-trig message:

```
xpl-trig  exec.basic { status='executed' program='/home/USER/bin/setchacon.sh salon on  
↪ ' type='script_action' }
```

The **script_info_binary** xpl-trig message:

```
xpl-trig exec.basic { status='1' program='/home/USER/bin/check_process.sh lcdmeteo.py  
↪ ' type='script_info_binary' }
```

The **script_info_number** xpl-trig message:

```
xpl-trig exec.basic { status='1999.8' program='/home/USER/bin/listdatawatermysql.sh -  
↪ jour' type='script_info_number' }
```

3.1 2.0

- No xPl Version
- Add more Datatype sensors (Script Info Temperature, Script Info Humidity, Script Info OpenClose, Script Info OnOff, ScriptOnOff, Script String)
- Display error log in Advanced page

3.2 1.0

- Add more control before executing script and more log to explain failed scripts
- Add possibility to use spaces in a parameter
- Stable version

3.3 0.2

- Adding **Script Info Binary** and **Script Info String**
- Sensor scripts (**Script Info Binary**, **Script Info Number** and **Script Info String**) will be executed every n secondes configured in the admin interface.

Don't need a scenario to do that anymore. * Changing xPL messages format * **WARNING:** This version is not compatible with the **stable** version, it's necessary to recreate the devices.

3.4 0.1

- Plugin creation