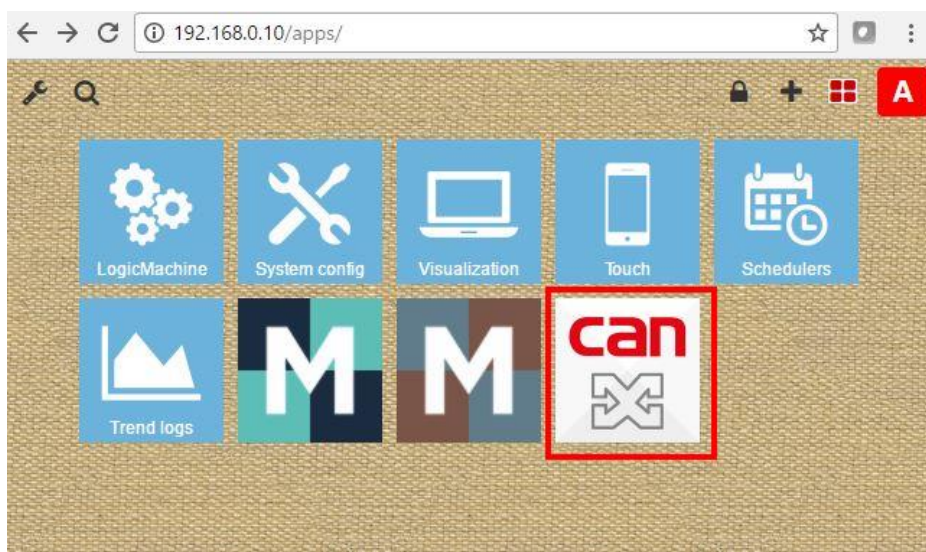



Embedded Systems SIA, VAT No LV40003411103  
47. Katolu str., Riga, LV 1003, LATVIA  
Phone: +371 67648888, fax: +371 67205036, e-mail: [sales@openrb.com](mailto:sales@openrb.com)

## canX app on LogicMachine


canX app on LogicMachine allows to configure, monitor canX protocol based devices. canX app is available in LogicMachine application store.


















## Scanning CAN FT line


If there are already devices connected in online mode, go to **Line Scan** tab, specify **Line range** (or leave default) and click on  magnifier icon

Groups Devices Locations Connection helper **Line scan** Device scan Reports Monitor Notes Tools

Line range: 0. 0. New line 0. 0. New line 

Filter devices: All Prog Error

Address	Name	Type	HW-SW ID	State	
0.12		UIO16	00 00 00 01 / 01	-	  
0.6	8R (8 Relay outputs)	8R (8 Relay outputs)	00 00 00 03 / 01	-	  
0.4	UIO16	UIO16	00 00 00 01 / 01	-	  
0.3	UIO16	UIO16	00 00 00 01 / 01	-	  
0.2	UIO16 Name 1	UIO16	00 00 00 01 / 01	Prog	  




Once devices are found, the scanning process can be cancelled with  icon.  
*Address* – physical address of the device

*Name* – name of the device. Can be changed in **Devices** tab.

*Type* – type of the device (profile)

*HW-SW ID* – hardware ID and software ID

*State* – Defines either the device is in Programming state or not (the programming button is pressed)

	add the device to <b>Devices</b> for further project configuration
	Scan and show device configuration ( <b>Device scan</b> )
	Flash device LED

## Device scan

Device scan shows all objects of the specific device.

Groups
Devices
Locations
Connection helper
Line scan
Device scan
Reports
Monitor
Notes
Tools

Line
0. 0. New line
Node
6

Filter state

Filter mode

ID	State	Type / Name	Data type	Value	Mode	Flags	Groups	Write config
1	✓	A Relay 1	1 bit (bool)	1	Normal - On after power-up	F T R W	0 / 20	
2	✓	A Relay 2	1 bit (bool)		Normal - Off after power-up	F T R W	0 / 20	
3	✓	A Relay 3	1 bit (bool)		Inverse - On after power-up	F T R W	0 / 20	
4	✓	A Relay 4	1 bit (bool)		Inverse - Off after power-up	F T R W	0 / 20	
5	✓	A Relay 5	1 bit (bool)		Disabled	F T R W	0 / 20	
6	✓	A Relay 6	1 bit (bool)		Disabled	F T R W	0 / 20	
7	✓	A Relay 7	1 bit (bool)		Disabled	F T R W	0 / 20	
8	✓	A Relay 8	1 bit (bool)		Disabled	F T R W	0 / 20	
9	✓	S Relay status 1	1 bit (bool)		Disabled	F T R W	0 / 20	
10	✓	S Relay status 2	1 bit (bool)		Disabled	F T R W	0 / 20	
11	✓	S Relay status 3	1 bit (bool)		Disabled	F T R W	0 / 20	
12	✓	S Relay status 4	1 bit (bool)		Disabled	F T R W	0 / 20	

*ID* – ID number of the object

*State* – internal state of the device, defines is there are not mistakes with the device

*Type / Name* – Name of the object

*Data type* – data type of the object

*Value* – current value of the object

*Mode* – mode of the object

*Relay / Output modes:*

Disabled

Normal, off after power-up

Inverse, off after power-up

Normal, on after power-up

Inverse, on after power-ups

*Relay status modes:*

Disabled

Normal

Inverse

*Input modes:*

Switch on/off

Switch off/on (inverse)

Switch Toggle

Button Toggle (optional long press)

Button On (optional long press)

Button Off (optional long press)

Button Start/Stop

Button Stop/Start (inverse)

*Input Long Press modes:*

Long press Toggle

Long press On

Long press Off

*Flags*



F – Filter. Defines if the device can filter telegrams (for CAN-CAN Routers)

T – Transmit. Defines if the device can initiate communication

R – Read. Defines if the read command is allowed to the device

W – Write. Defines if the write command is allowed to the device

*Groups* – number of groups associated with a particular object

	Configure device object mode, flags and group addresses
	Set device object value

Once some change is done  button gets active. By finishing all changes and clicking on this button, all updated configuration is written to a respective device.

## Group address configuration

Group addresses can be configured either in object configuration settings or directly in **Groups** tab.

Groups

Devices

Locations

Connection helper

Line scan

Device scan

Reports

Monitor

Notes

Tools

Name or address

Datatype

Tags

Location

Properties

- All datatypes -

- All locations -

E R P

Address	Name	Datatype	Tags	Value	Properties	
0/0/11	UIO16 (16 Universal IO ports) - Output 1	0.1. 1 bit (boolean)		0	E R P	
0/0/12	UIO16 (16 Universal IO ports) - Output 1	0.1. 1 bit (boolean)		0	E R P	
0/0/13	UIO16 (16 Universal IO ports) - Output 1	0.1. 1 bit (boolean)		0	E R P	
0/0/14	UIO16 (16 Universal IO ports) - Output 1	0.1. 1 bit (boolean)		0	E R P	
0/0/15	UIO16 (16 Universal IO ports) - Output 1	0.1. 1 bit (boolean)		0	E R P	
0/0/16	UIO16 (16 Universal IO ports) - Output 1	0.1. 1 bit (boolean)		0	E R P	
0/0/17	UIO16 (16 Universal IO ports) - Output 1	0.1. 1 bit (boolean)		0	E R P	
0/0/18	UIO16 (16 Universal IO ports) - Output 1	0.1. 1 bit (boolean)		0	E R P	
0/0/19	R8 (2 Relay outputs) - Relay status 1	0.1. 1 bit (boolean)		0	E R P	
0/0/20	R8 (2 Relay outputs) - Relay status 1	0.1. 1 bit (boolean)		0	E R P	

1

2

3

Showing group addresses 11-20 of 21

*Address* – group address

*Name* – group name

*Datatype* – group data type

*Tags* – tags for a specific group

*Properties* – properties of group



L – Log each group change

E – Export. Used to map this group automatically to KNX group (**Objects** tab in LogicMachine)

P – Post process. Execute a script on group address change

R – Read on initialization (device start)

*Location* – location of the object/group

	List devices that use this group address
	Read/write group address value

### Edit group address

Group address

0/0/1

Name

UIO16 Name 1 - Input 1

Datatype

0.1. 1 bit (boolean)

Tags

lightsGroup

Location

Office Demo / Floor 1 / Room 101

L Log

E Export




P Post-process

R Read on init

Comments

Save

Cancel

	Post-process script
	Edit group address
	Delete group address

### Importing group addresses from ETS project file

 Import KNX project

button allows to import KNX project file.

Import JSON
×

---

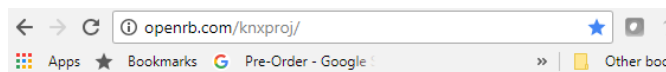
Choose JSON file  
or drag it here


KNX project file can be converted to JSON at  
<http://openrb.com/knxproj/>

---

Cancel

Prior to importing, ETS project file need to be converted to JSON file using this online utility




  
The easiest way to create complex installation

KNX project file converter for LogicMachine

☐ Include parent level names in object name

Select file

## Device configuration

Devices section is used to configure canX devices. The configuration can be done even if the device is not yet physically connected to the line (offline mode).

Groups

Devices

Locations

Connection helper

Line scan

Device scan

Reports

Monitor

Notes

Tools

?

✕

All

Name

Device type

Tags

All tags

Any tag

Location

Exact

Incl. sub

0

- All types -

- All locations -

1

Address

Name

Device type

Tags

Write project

Add

2

0.2

UIO16 Name 1

UIO16 (Binary inputs/outputs)

input

3

0.3

UIO16

UIO16 (Binary inputs/outputs)

4

0.4

UIO16

UIO16 (Binary inputs/outputs)

5

0.6

8R (8 Relay outputs)

8R (8 Relay outputs)

actuator

6

0.7

8R (8 Relay outputs)

8R (8 Relay outputs)

7

0.10

UIO16

UIO16 (Binary inputs/outputs)

8

0.11








UIO16

UIO16 (Binary inputs/outputs)

9

10

- Address – physical address of the device
- Name – name of the device
- Device type – device type / profile
- Tags – tags associated with the device

	Check if device is online on the bus
	Duplicate device configuration
	Scan and show device configuration
	Write configuration to the device
	Configure device parameters
	Edit device base configuration
	Delete device

When clicking on a specific entry, device settings appear.

Edit device

Line

0. 0. New line

Node

4

Name

UIO16

Device type

UIO16

More info

Profile mode

Binary inputs/outputs

Tags

input

Location

Office Demo

Block write (skip device during full project write)

Comments

Save

Cancel

*Line* [0..15] – Line ID

*Node* [1..255] – Node ID

*Name* – Name of the device


*Device type* – automatically detected device type

*Profile mode* – software profile mode for the device. There are several profiles possible for one type e.g. for UIO16 – Binary input/outputs, binary outputs, binary inputs. This makes easier and faster configuration by limiting count of objects

*Tags* – associated tags with this device

*Location* – location of the device

*Block write* – skip device during full project write

When clicking on Parameters icon , objects settings appear. Depending on the software profile chosen, you see respective control option for each channel.

8R 8 channel relay module:

## 8R (8 Relay outputs) (0.7)



All Enabled Disabled

Port 1  
Port 2  
Port 3  
Port 4  
Port 5  
Port 6  
Port 7  
Port 8

Relay 1 Relay status 1

Relay 1

Normal - On after power-up

Flags F T R W

Group addresses 1 bit (boolean)

0/0/4 Relay 1

0/0/5 8R (8 Relay outputs) - Relay 1

0/0/8 Relay 1

0/0/9 Relay 1

## UIO16 16 channel universal input/output module:

### UIO16 (0.11)



All Enabled Disabled

Port 1  
Port 2  
Port 3  
Port 4  
Port 5  
Port 6  
Port 7  
Port 8  
Port 9  
Port 10  
Port 11  
Port 12  
Port 13  
Port 14  
Port 15  
Port 16

Output 1 Output status 1 Input 1

Input 1

Switch - Toggle

Flags F T R W

Send inverse of the current value when switched On or Off

Group addresses 1 bit (boolean)

0/0/11 UIO16 - Input 1

When the changes are done for the device, it is marked in yellow.



Groups

Devices

Locations

Connection helper

Line scan

Device scan

Reports

Monitor

Notes

Tools

All

0

1

2

3

4

5

6

7

8

9

Name

Device type

Tags

Location

- All types -

- All locations -

All tags

Any tag

Exact

Incl. sub

Write project

Add

Address	Name	Device type	Tags	
0.2	UIO16 Name 1	UIO16 (Binary inputs/outputs)	input	
0.3	UIO16	UIO16 (Binary inputs/outputs)		
0.4	UIO16	UIO16 (Binary inputs/outputs)		
0.6	8R (8 Relay outputs)	8R (8 Relay outputs)	actuator	
0.7	8R (8 Relay outputs)	8R (8 Relay outputs)		
0.10	UIO16	UIO16 (Binary inputs)		
0.11	UIO16	UIO16 (Binary inputs/outputs)		

Press this button 

Write project

 to upload new settings to all modified / all devices.

Select write mode 

×

Device list

Whole project

Matching current filter

Device status

Only modified





































All devices

Write

Cancel

## Locations





Project structure can be defined in **Locations** tab.


Groups	Devices	Locations	Connection helper	Line scan	Device scan	Reports	Monitor	Notes	Tools			
Name	Comments	Devices	Groups	+ Add								
Office Demo		0 / 2	0 / 3	   								
└ Floor 1		0 / 2	0 / 3	   								
└└ Room 101		2 / 2	2 / 2	   								
└└ Room 102		0 / 0	1 / 1	   								
└└ Room 103		0 / 0	0 / 0	   								
└ Floor 2		0 / 0	0 / 0	   								
└└ Room 201		0 / 0	0 / 0	   								
└└ Room 202		0 / 0	0 / 0	   								
└└ Room 203		0 / 0	0 / 0	   								

*Name* – name of the location

*Devices* – devices associated with a particular location. First number indicates count of specific device in a particular room. Second - total count of devices.

*Groups* – group addresses associated with a particular location. First number indicates count of specific groups in a particular room. Second - total count of groups.

	Add new sub-location
	Duplicate location, including devices and group addresses
	Edit location
	Delete location and all sub-locations

By clicking on  button, you will add a new location

Add new location ×

**Parent location**  

Office Demo

**Name**  

Floor 2

**Comments**

Save

Cancel

*Parent location* – choose parent location

*Name* – name of the new location

## Reports

**Reports** section lists all devices in the project indicating which devices are located in which line, how many group addresses are associated with a particular device, how much space it is required in the cabinet.

GroupsDevicesLocationsConnection helperLine scanDevice scanReportsMonitorNotesTools

Report for project Office Demo

Print

Address	Name	Device type	Group associations	Width (units)
0. New line				
0.2	UIO16 Name 1	UIO16 (Binary inputs/outputs)	2	3
0.3	UIO16	UIO16 (Binary inputs/outputs)	6	3
0.4	UIO16	UIO16 (Binary inputs/outputs)	1	3
0.6	8R (8 Relay outputs)	8R (8 Relay outputs)	0	4
0.7	8R (8 Relay outputs)	8R (8 Relay outputs)	5	4
0.10	UIO16	UIO16 (Binary inputs)	0	3
0.11	UIO16	UIO16 (Binary inputs/outputs)	0	3
Line total (7 devices)			14	23
Project total (7 devices)			14	23

## Monitoring the fieldbus line

In **Monitor** tab you can see the activity on the bus line, send read/write requests to specific device / group /object.

GroupsDevicesLocationsConnection helperLine scanDevice scanReportsMonitorNotesTools

Command typeGroupCommandGroupValueWriteGroup address0/0/8Data (HEX, space-separated) 1-8B00

Filter messagesAllIndividualGroupFilter directionAllRXTXName or address

Stop monitorExport CSVClear

#	Time	Type	Address	Name	Command	Data	
50	14:24:02.868	RX Group	0/0/10	Relay status 1	GroupValueResponse	00	
49	14:24:02.866	TX Group	0/0/8	Relay 1	GroupValueWrite	00	
48	14:23:57.421	RX Group	0/0/10	Relay status 1	GroupValueResponse	01	
47	14:23:57.418	TX Group	0/0/8	Relay 1	GroupValueWrite	01	
46	14:23:24.270	RX Object	0.7.1	8R (8 Relay outputs) (Relay 1)	GAListResponse	09 00 08 00 04	
45	14:23:24.267	TX Object	0.7.1	8R (8 Relay outputs) (Relay 1)	GAListRead	-	
44	14:23:17.910	RX Device	0.7.0	8R (8 Relay outputs)	DeviceModeResponse	00	
43	14:23:17.907	TX Device	0.7.0	8R (8 Relay outputs)	DeviceModeSet	00	

*Time* – time of telegram received

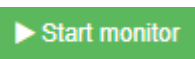
*Type (RX/TX; Object/Device/Group/Control)* – type of telegram

*Address* – group or physical device address the telegram was received to/from

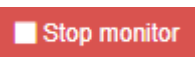
*Name* – name of the device

*Command* – command received

*Data* – data received



Start monitor to see real-time messages



Stop monitoring



Export monitoring results to CSV file



Clear the list



Fill the search fields with a respective data from a selected entry

### Send a command to a bus line

Choose command type and other parameters and click on green arrow to send request

Command type	Command	Group address	Data (HEX, space-separated) 1-8B	
<div>Group ▾</div>	<div>GroupValueWrite ▾</div>	<div>0/0/8</div>	<div>01</div>	<div>&gt;</div>

### Command types

#### Object

***ObjectValueRead*** - Request of Value for Object

***ObjectValueResponse (1..8 bytes)*** – Response for ObjectValueRead command, return Value for given Object

**ObjectValueWrite (1..8 bytes)** – Write Value to given Object

**ObjectInfoRead** - Read request for EEPROM address, with address as parameter for command

**ObjectInfoResponse (1 byte)** - Response for ReadEEProm, return Value for given EEPROM address

**ObjectConfigWrite (1 byte)** - Write EEPROM addr. = object id (response with ResponseObjectEpromConfig msg. if success)

**GAListAdd (4 bytes)** – Add the group address to list for given object

**GAListDel (4 bytes)** - Delete given GA from list for Object, GA as parameter for delete command

**GAListClear** - Delete full list of group addresses connected to the given object

**ObjectCmdResponse** - ..

**GAListRead (1 byte)** - Counter of batch GA as parameter (1 for first 2 addr., 2 for second 2 addr. ..)

**GAListResponse (8 bytes)** - Response for GetListGA command, return 2 group adr. connected to object ( 2 x 4bytes )

## Device

**DeviceInfoRead** – Read Hardware ID and Software ID for given device, device status and device Objects Quantity

**DeviceInfoResponse (8 bytes)** – Response for ..., return ObjQty (byte), status (byte), 5 bytes for HW id and 1 byte for SW id

**DeviceAddrRequest (8 bytes)** - Request for address for this device (current Line ID and Node ID ) with HW+SFT id as message DATA

**DeviceAddrWrite** - Return: **Line ID** ( d[1] ), **Node ID** ( d[0] ) and **Master Code** for device 6 bytes ( d[2] - d[7] )

**DeviceConfigRead (2 bytes)** – EEPROM read command for given Address ( 2 bytes DATA )

**DeviceConfigResponse (8 bytes)** – Return 8 byte of data from EEPROM for given start address

**DeviceConfigWrite (8 bytes)** – Write 6 bytes of data with 2 bytes address

**DeviceConfigWriteResponse (1 byte)** – 1 if write is successful of 0 if failed

**DeviceModeSet** – set device mode

***DeviceModeResponse*** – ...

***DeviceLedOn*** – Switch the device green LED to **ON** state

***DeviceLedOff*** – Switch the device green LED to **OFF** state

***DevicePing*** – Ping request to device with given Line ID and Node ID

***DevicePong*** – Response on Ping request with CAN FT bus TX and RX errors and maximum quantity of GAs per Object

## Group

***GroupValueRead*** – Request of Value for given Group address

***GroupValueResponse (1..8 bytes)*** – Response for GroupValueResponse command, return Value for given Group address

***GroupValueWrite (1..8 bytes)*** – Write Value to given Group address

## Control

***DirectMsgOn (8 bytes)*** – Switch on direct communication with code ( master\_code = Fx(code) )

***DirectMsgOff (8 bytes)*** – Switch off direct communication with code ( master\_code = Fx(code) )

***SecureGAOn (8 bytes)*** – Switch to Secure GA mode with code ( master\_code = Fx(code) )

***SecureGAOff (8 bytes)*** – Switch Off Secure GA mode with code ( master\_code = Fx(code) )

## Write device / physical address

To write device another physical address, go to **Tools** → **Write device address**. Choose Line and Node number and click **Write**. Then click and release programming button on the device. The Write window will close once the new address is written.

Write device address

Line

0. 0. New line

Node

1

Programming button on the device must be pressed **after Write** button is clicked

Write

Cancel

## See current physical address of the device

To see currently flashed physical address, go to **Monitor** tab, start monitor. Then shortly press programming button on the device, you will see current physical **Address**.

Groups

Devices

Locations

Connection helper

Line scan

Device scan

Reports

Monitor

Notes

Tools

Command type

Object

Command

ObjectValueRead

Line

0. 0. New line

Node

1

Object

1

Data (HEX, space-separated)

Filter messages

All

Individual

Group

Filter direction

All

RX

TX

Name or address

Stop monitor

Export CSV

Clear

#	Time	Type	Address	Name	Command	Data
1	09:59:00.870	RX Device	4.4.0		DeviceAddrRequest	02 03


## Project configuration


General project settings are available in **Tools → Project configuration**.

Project configuration ×

---

**Project name**

**Master code**  
 

*Up to 4 HEX characters, separated by space*  
**Recommended action:** click  to generate new random code

Master code is written to devices together with physical address. It must be set only once per project. Changing master code requires physical device reset via programming button.

**Semantic dictionary support**  
☒ Use tag dictionary from Project Haystack [More info](#)

---

Save Cancel

*Project name* – name of the project

*Master code* – master code is written to devices together with physical address. It must be set only once per project. Changing master code requires physical device reset via programming button. Master code is a protection for the installer from possibility to read or program canX devices by other parties. If it is necessary to block full access to the canX device, it is necessary to disable direct communication / messages in **Tools → Disable direct communication** or through **Monitor**.

*Semantic dictionary support* – define either tag dictionary is used from Project Haystack: <https://project-haystack.org/> Project Haystack is an open source initiative to streamline working with data from the Internet of Things. Haystack standardize semantic data models and web services with the goal of making it easier to unlock value from the vast quantity of data being generated by the smart devices that permeate our homes, buildings, factories, and cities.

## New device profiles

To add new canX device profiles to your canX app, go to **Tools → Upload device profile**



Upload profile ✕

Choose device profile  
or drag it here

Cancel

## Find device if you have several of them in the line

In Line Scan when scanning, you can click on programming button of the device and the respective field will be set to green color

Devices

Group addresses

Line scan

Device scan

Write address

Monitor

Notes

Tools ▾

?

✕

Line range

0

0

Q

Filter devices

All

Prog

Error

Address	Name	Type / HW-SW ID	State
0.3	UIO16 (Binary input/output)	UIO16 (Binary input/output)	Prog

## Connection helper

Connection helper allows easing making connection between unused input and output objects.

Groups

Devices

Locations

Connection helper

Line scan

Device scan

Reports

Monitor

Notes

Tools ▾

?

✕

Add new group address

Group address

0/0/19

Name

Kitchen light

Datatype

0.1. 1 bit (boolean)

Tags

Q No tags set

Location

Office Demo / Floor 1 / Room 101

Log

Export

Post-process

Read on init

Comments

Datatype

- All datatypes -

Tags

All tags Any tag

Location

- All locations -

Exact

Incl. sub

Show objects

Uncollected

All

Save connection

Actuators

8R (8 Relay outputs) - Relay 2

8R (8 Relay outputs) - Relay 3

8R (8 Relay outputs) - Relay 4

Sensors

UIO16 - Input 2

UIO16 - Input 3

Save

Cancel

Once the Save Connection button is clicked, a new group address window is opened

Address – group address  
Name – group name  
Datatype – group data type

*Tags* – tags for a specific group

*Properties* – properties of group

L – Log each group change

E – Export. Used to map this group automatically to KNX group (**Objects** tab in LogicMachine)

P – Post process. Execute a script on group address change

R – Read on initialization (device start)

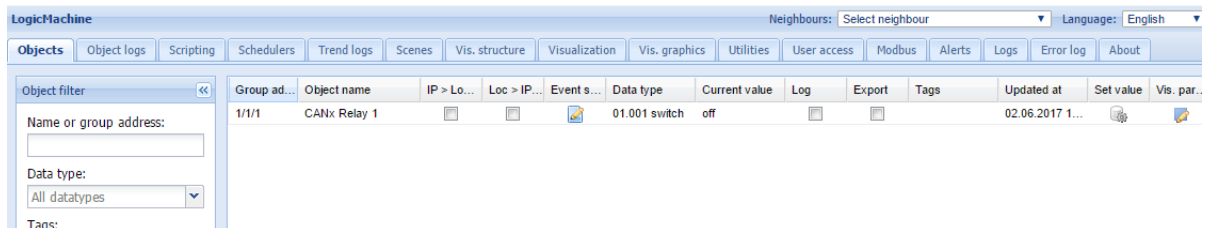
*Location* – location of the object/group

## Reset the device

Press programming button for ~3 sec and release when the device blinks red twice. Wait while green LED will blink once. Preferably is to restart the power as well after this is done.

## Accessing canX objects from LogicMachine scripts

To use canX configuration from LogicMachine scripts use scripting



Control via script via grp address:

```
can = require('canx')
can.sendrequest({
  data = event.getvalue(),
  command = can.cmds.group.valuwrite,
  groupaddress = 1,
})
```

Control via physical address:

Change lineid / nodeid to match your relay device settings.

```
can.sendrequest({
```

```
lineid = 0,  
nodeid = 1,  
objectid = 1,  
command = can.cmds.object.valuewrite,  
data = grp.getvalue(),  
})
```