DOMIQ©

The fine art of building automation

DOMIQ/Base

Configuration Manual

Software version 1.8.0.3

March 2014

CM–BL–EN–140318

This manual describes the **DOMIQ** products and software in the latest version available at the time of publication of this document. Due to continuous product development are possible minor changes in appearance and functionality of each screen. In case of doubt, please contact us:

Skype: **domiq–support**
Email: **info@domiq.pl**

Table of Contents

Chapter 1
Introduction

This manual describes the configuration and operation of the **DOMIQ/Base** module. By reading this manual you will learn all the features offered by the **DOMIQ/Base** module.

This document includes the examples of use. Device configuration and implementation of standard functionality does not require programming knowledge. Advanced functionality, that is beyond the scope of this manual requires a knowledge of the programming in Lua scripting language. The basics of Lua with examples of its use are described in the "**Programming Manual**".

The **DOMIQ/Base** installation leaflet and tutorials, which describe useful examples of application of the **Base** module and other **DOMIQ** devices extends description of the **DOMIQ/Base** functionality.

Tutorials can be downloaded from our website **www.domiq.eu**, from the **Tutorials** section.

LEGEND

While reading this manual you will encounter some dedicated style text formatting and symbols:

• **DOMIQ/Base** – All product mentions are indicated in this way.

• **Local Network Only** – This formatting style is used for all interface elements, such as the names of tabs and controls, fields to fill, etc., and for all unique names contained on our website.

• *Admin*, *10*, *200* – In this way is indicated content that user has to enter manually.

• `C.LCN.output.0.10.1 = on` – in this way all the expressions related to identifiers (state, event, command) or Lua code are represented.

In this way the examples are indicated.

- Example 1
- Example 2

Chapter 2
DOMIQ Products

The **DOMIQ** system is a home automation system, which allows to integrate the most often used building automation subsystems into a coherent installation. Subsystems integration allows you to design better control algorithms and use of user interfaces on mobile devices, touch panels and computers. With the possibility to define an unlimited number of events and times, the **DOMIQ** system greatly facilitates the process of building automation. Remote access increases the flexibility of work of an installer and allows users to have an overview of status of a building from anywhere in the world.

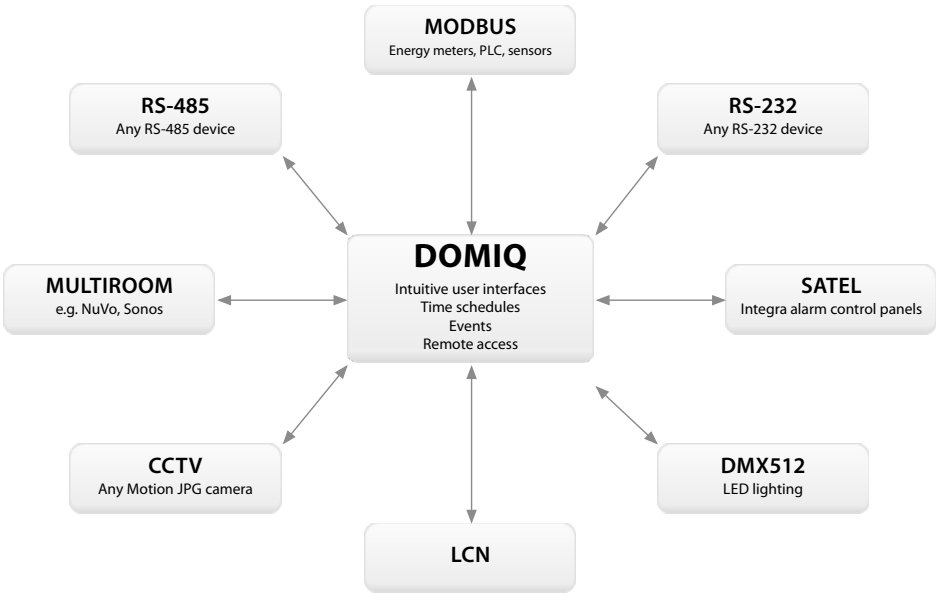


Fig 2.1: The integration diagram of building automation subsystems with use of the DOMIQ system.

The most important **DOMIQ** products are:

- **DOMIQ/Base**: main system module, allows to connect other system to the **LCN** installation and to the Internet.
- **DOMIQ/Remote**: native mobile application for remote control of the automation system, dedicated for **iPhone**, **iPad** and **iPod Touch**.
- **DOMIQ/Display**: dedicated TFT touch panel, available in desktop or wall–mounted version.

For integration with other subsystems following extension modules can be used:

- **DOMIQ/Serial–2SI**: integration with **SATEL Integra** alarm control panels.
- **DOMIQ/Serial–4DX**: LED lighting control using the DMX–512 protocol.
- **DOMIQ/Serial–4MB**: integration with devices using the MODBUS protocol.
- **DOMIQ/Serial–2SG**: integration with any devices using RS–232 serial interface.
- **DOMIQ/Serial–4SG**: integration with any devices using RS–485 serial interface.
- **DOMIQ/Meter**: counter module that can be connected to the water or energy meters

2.1. DOMIQ/Base

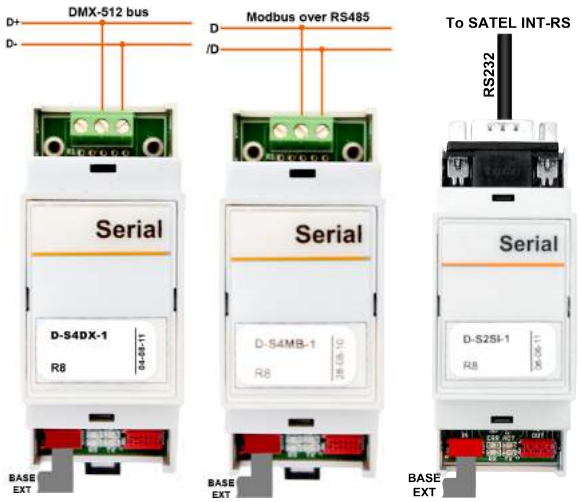
DOMIQ/Base is the main component of the **DOMIQ** system. Its functionality includes:

- Simply configurable user interfaces: built–in interactive visualization editor and editor for creating menus for mobile devices.
- Unlimited number of timers, which greatly simplifies the process of building automation.
- Unlimited number of events, which allows to create more sophisticated logical rules.
- User–friendly configurator.
- **Base–Base** type connection, which allows to create large building automation installations based on **DOMIQ** devices.
- Built–in **LCN** interface.
- Using the **Base** module as a segment coupler in a **LCN** installation.
- Remote access for **LCN** installers using **LCN–Pro** software (the **LCN–PCHK** protocol)
- Integrate real time clock with battery backup, automatic synchonization with time servers using Internet, automatic summer/winter time.
- Support for Internet protocols: IPv4, IPv6, DHCP, AutoIP, DNS, HTTP, SNTP, FTP, Telnet.

2.2. DOMIQ/Serial

Serial extension modules are available in two versions with different electrical interface:

- **RS–232** – to communicate for short distances between a pair of devices
- **RS–485** – to coummunicate for longer distances and between multiple devices.



Particular types of modules have also different software, which is always adapted to the specific communication protocol. Each type of **Serial** module is controlled using dedicated state identifiers, command and events.

Type	Description	Electric interface	Identifiers
D–S2SI–1	Interface for integration with **SATEL Integra®**	RS–232	IDS
D–S4MB–1	**MODBUS** master interface	RS–485	MODBUS
D–S4DX–1	**DMX–512** master interface	RS–485	DMX
D–S2SG–1	**RS–232** generic serial interface	RS–232	SER
D–S4SG–1	**RS–485** generic serial interface	RS–485	SER

2.3. DOMIQ/Light

DOMIQ/Light enables you to control the lightinh installationsbased on the DALI standard.

In accordance with the DALI standard you can connect up to 64 DALI devices to the ,**DOMIQ/Light** module. The driver provides full bi-directional communication (such as brightness update, information about the failure of the fluorescent lamps). The module has an implemented algorithm for automatic ballast addressing (64 lamps are addressed simultaneously and automatically).

2.4. DOMIQ/Remote

The **DOMIQ/Remote** is a dedicated application for Apple mobile devices (**iPhone/iPad/iPod Touch**), which through the **DOMIQ/Base** allows to control and visualize state of intelligent installation. The application is available for free in the **AppStore**.

Control and viewing of a building automation system can be implemented in two ways: through a convenient menu, whose contents are fully configurable (read more in Chapter 6. Remote) and interactive visualizations. The process of creating and editing visualization is described in Chapter 7. Display.

Fig 2.2: Controlling of building automation using the menu of DOMIQ/Remote application

Overview and building control on a single screen. Visualizations are displayed when you rotate a device into horizontal position. You can zoom in and out a visualization using multi touch gestures.

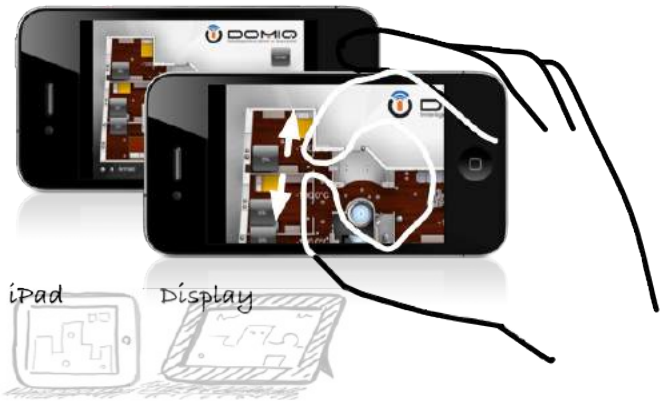


Fig 2.3: Controlling of building automation using visualizations

2.5. DOMIQ/Display

The **DOMIQ/Display** touch panel allows for quick overview of state of your house or apartment, thanks to fully configurable visualization. Exactly the same visualization can be displayed on the **Display** panels as well on the **DOMIQ/Remote** application.

The **Display** panel is available in three versions:

Type	Desciption
D–DSP10B–1	Touch panel with front made of anodized aluminium.
D–DSP10C–1	Touch panel with front made of anodized aluminium and box for flush mounting. Front is mounted to the box using neodymium magnets, which allows. It is possible to split purchase into two parts, described below.
D–DSP10D–1	Touch panel, included in D-DSP-10C-1, for separate purchase.
D–DSP10P–1	Flush mounted box, included in D-DSP10C-1, for separate purchase.

Fig 2.4: DOMIQ/Display panel with front made of anodized aluminium

Fig 2.5: DOMIQ/Display panel in wall–mount version

Communication with the module **DOMIQ/Base** takes place using WiFi network or Ethernet cable connection.

Chapter 3
First Steps

The **DOMIQ/Base** module is connected similarly as **LCN** modules: directly to the mains and to the **LCN** data wire (**D**). **EXT** connector is used to connect additional **DOMIQ** modules, such as **DOMIQ/Serial** in order to provide integration with other automation systems (SATEL, MODBUS, DMX-512) or others.

Fig 3.1: DOMIQ/Base wiring diagram

To use full functionality of the **Base** module is necessary to connect it to the Ethernet network. To do that, connect Fast Etherent RJ–45 wire to the **LAN** connector of a module. Connecting the **Base** module to network is necessary to control building automation installation via the **DOMIQ/Remote** application or the **Display** panel.

It is strongly recommended that **DOMIQ/Base** has an active Internet connection. The Internet connection is necessary for:

- Remote (outside local WiFi) control of the automation system.
- Updating of the **Base** module software.
- Synchronization of the real–time clock with internet time servers using the SNTP protocol.

After turning on, the **Base** module will try to obtain an IP address from a DHCP server. If this is not possible, because, for example, the local network has no DHCP server, then an address in the AutoIP standard is automatically assigned.

Assigned IP address can be found using the **DOMIQ/Discover** application or the **DOMIQ/Remote**.

3.1. Configuration

The **DOMIQ/Base** has build a comfortable and easy to use configuration interface implemented as a Flash application. Configurator is loaded directly from the embedded Web Server. Configurator can be opened in any web browser. Adobe Flash Player version 11.0 or later is required. We recommend use of the **Google Chrome** browser due to included Flash plugin.

Adobe Flash Player is available for most web browsers, in particular:

- **Chrome** (Flash Player included by default, Windows, Mac OS X, Linux)
- **Firefox** (Windows, Mac OS X, Linux)
- **Safari** (Windows, Mac OS X)
- **Opera** (Windows, Mac OS X)
- **Internet Explorer** (Windows). Due to internal browser bugs, configuration interface operation is slower than under other browsers, because of the need to block caching.

All configuration changes made using configuration interface are cached in web browser and do not affect the operation of the **DOMIQ** system until the **Save** button was pressed.

Each configurator's tab in addition to the **Save** button has the **Revert** button in the right–bottom corner. Pressing this button restores last saved configuraton.

Finding an IP address of the DOMIQ/Base

In order to start the configurator you need to enter Base's IP address in the web browser. IP address can be found in a few different ways:

- Using the **DOMIQ/Discover** application.
- Using the **DOMIQ/Remote** application.
- By checking the register of allocated IP addresses in your router.

The **DOMIQ/Discover** application is available for free on our webpage **www.domiq.eu**. Versions for **Windows** and **MacOS** are available. Download links are placed in the webpage with description of the **Base** module. The **Discover** application finds all **Base** modules that are present in a particular network. After running the program, following window appears:

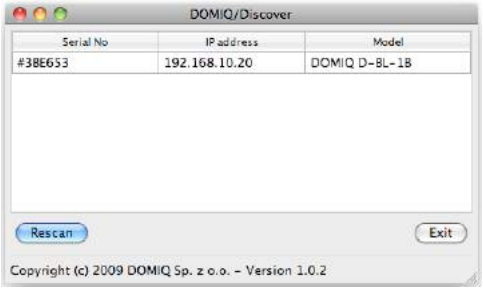


Fig 3.2: DOMIQ/Discover application

Double–clicking on the module description displays information from internal console of the module. This information can be useful in solving problems.

You can also read the IP address in the **DOMIQ/Remote** application menu. Clicking on the blue icon with an arrow displays information about detected modules.

Fig 3.3: FInding the Base module's IP address using the DOMIQ/Remote application.

You can also find the IP address by checking the list of the assigned DHCP router addresses.

Running the Configurator

After entering the **Base's** IP address in a web browser, the login screen will be loaded. The application will ask you to enter username and password. Default login and password is: *admin*.

Fig 3.4: Login screen

After clicking on the **Configuration** button, default configuration screen is loaded. The configurator has tabbed structure. Each tab is described in details in the remainder of this manual.

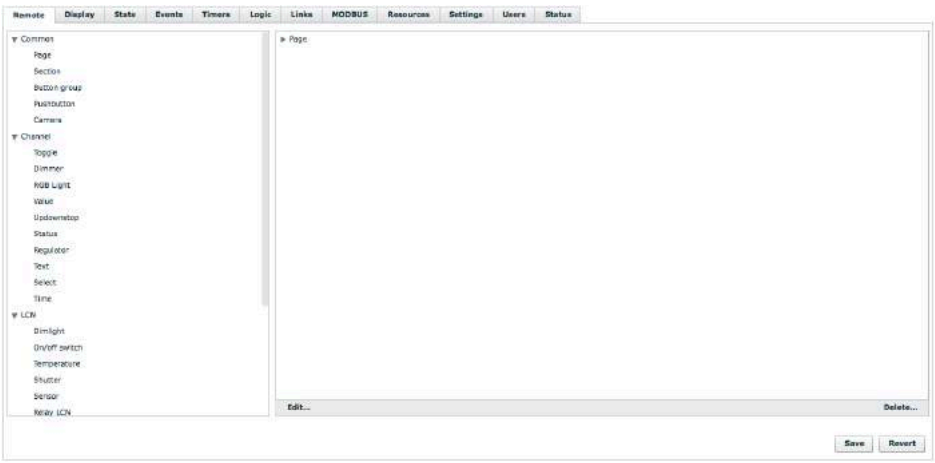


Fig 3.5: General view of the configuration interface

Whereas if on the login screen you click on the **Visualization** button, a default visualization will be displayed, from which you can control the automation system.

Marking the option "**Fullscreen Visualization**" shows the visualization by filling the whole window of the browser. In this mode the visualization can be smoothly scaled-if we make the browser window smaller, the visualization adjusts to the new size of the window. If the option "**Fullscreen Visualization**" is not marked, it will be shown in the resolution of 800x600 pixels. In this mode the visualization is not scaled when the size of the browser window is changed. The chosen mode is locally stored in the cache of the browser.

Fig 3.6: An examplary visualization

3.2. Access Levels

After the first log in, we strongly advise to change login and password for each access level. In order to do that select the **Users** tab. The **Users** tab is divided into three sections: **User Access**, **Admin Access**, **Remote LCN**.

Fig 3.7: Access settings window

User Access

In this section you can configure login and password for user access. User has access only to the visualization screen. With user access level it is not possible to log into the **DOMIQ/Base** configurator. The following properties can be modified:

- **Access**: Here you decide from where user can access visualization:
  - **Local network only** – the visualization screen is only reachable for computers within local network to which the **Base** module is connected.
  - **Whole Internet** – you can gain access to the visualization screen either from local network or the Internet.
- **Username**: The user login that must be entered to gain access to the visualization.
- **Password**: The user password that must be entered to gain access to the visualization.

Admin Access

In this section you can set login and password to gain access to the configuration interface. The following parameters can be modified:

Access: Here you decide from where user can access the configurator:

- **Local network only** – the configurator is only reachable for computers within local network to which the **Base** module is connected.
- **Whole Internet** – you can gain access to configuration interface either from local network or the Internet.

- **Username**: The user login that must be entered to gain access to the configurator.
- **Password**: The user password that must be entered to gain access to the configurator.

Remote LCN

The **DOMIQ/Base** module can operate as a **LCN** coupler. In this section, remote access to **LCN** installation for **LCN–Pro** is configured. The following parameters can be modified:

Access: Here you decide from where user can access **LCN** installation:

- **Disabled** – access to the **LCN** bus using **LCN–Pro** is disabled.
- **Local network only** – **LCN** bus is only reachable for computers within local network to which the **Base** module is connected.
- **Whole Internet** – access to **LCN** installation is possible without address limitations.

- **Username**: The user login that must be entered in the **LCN–Pro** to gain access to **LCN** bus.
- **Password**: The user password that must be entered in the **LCN–Pro** to gain access to **LCN** bus.

Chapter 4
Settings

The **Settings** tab lets you to configure the basic parameters of the module and execute administrative operations. All changes made in the **Settings** tab are stored locally in the browser and do not affect on the **DOMIQ** system. Only when you press the **Save** button, changes will be saved in **DOMIQ/Base** module.

The **Revert** button restores the the last saved configuration of the **Settings** tab.

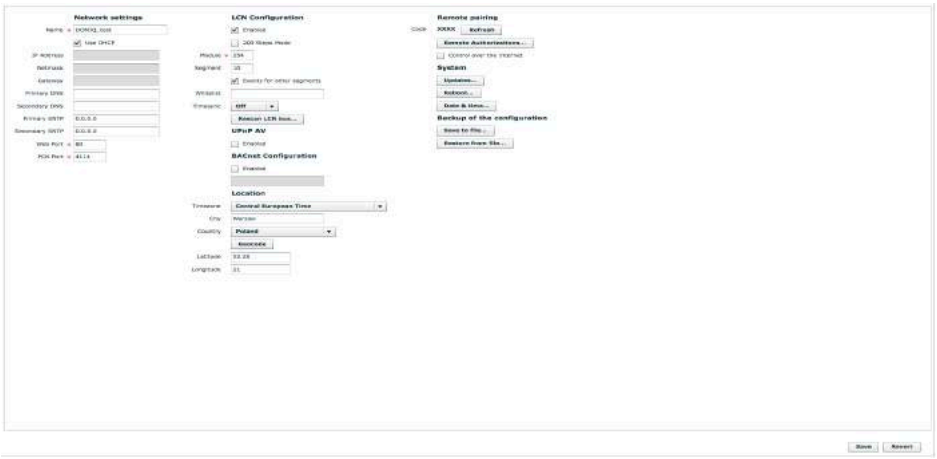


Fig 4.1: Base settings configuration window.

The **Settings** tab is composed of six parts:

1. Network settings
2. LCN Configuration
3. UPnP AV Configuration
4. BACnet Configuration
5. Location
6. Remote pairing
7. System
8. Backup of the configuration

It is necessary to fill the starred fields.

4.1. Network Settings

In this section are defined network settings of the **DOMIQ/Base** module. Moreover, here you assign a name (ID) to the **DOMIQ/Base** module. The name is displayed on the **Base** selection screen, in the **Remote** application and on the **Display** panel.

The following options are available:

- **Use DHCP**: If this option is selected, then **DOMIQ Base** tries to automatically obtain an IP address from a DHCP server (usually a router serves as DHCP server). The next five fields are editable only when the **Use DHCP** option is unchecked.
- **IP Address**: Static IP address assigned to the **DOMIQ/Base** module.
- **Subnet Mask**: The IP subnet mask.
- **Gateway**: Gateway in the IP subnet.
- **Primary DNS**: The IP address of the first domain name server.
- **Secondary DNS**: IP address of an additional server names.
- **Primary SNTP**: IP address of the Internet time server.
- **Secondary SNTP**: IP address of additional Internet time server.
- **Web Port**: The port number of a Web server running in the **Base** module that provides configuration interface.
- **PCK Port:** Port used for communicating the computer with **LCN–Pro** or **LCN–GVS** software.

If you leave SNTP settings blank, **Base** module, during the startup, will select random server from the global pool of time servers allocated for the use by DOMIQ products.

NOTICE! In order to apply any changes made in this section, you must restart the **Base** module.

In the case of problems with the network settings, you can reset the settings. The IP address will be assigned by the DHCP server again. In order to do this:

1. Disconnect the **Base** module from the power supply.
2. Put the jumper on the **ISP** connector as shown in the following picture:

Fig 4.2: The jumper mounted on the ISP connector

3. Connect the power supply (jumper put on the connector).
4. Wait approx. 2 minutes and disconnect the module from the power supply again.
5. Take off the jumper.
6. Start the module again. The module will receive the address assigned by the DHCP server.

4.2. LCN Configuration

Parameters of communications between the **DOMIQ/Base** module and **LCN** devices are configured in this section.

Following options are available:

- **Active**: Check box, which allows you to switch off the LCN interface in the **Base** module. In order to implement changes it is necessary to restart the module. When a field is selected (by default), then the LCN interface is on.
- **200-step-mode**: This option should be selected if the LCN modules were programmed in 200-step-mode. If this option is not selected in the case of LCN modules using the 200-step-mode, brightness status and control command will not work correctly. When in the installation there are older generation modules (without 200-step-mode), then you need to reprogram the entire installation to 50 step mode.
- **Module**: Identifier (Node ID) in the **LCN** address domain. Default ID of **Base** module is **254**.
- **Segment**: Required only when using **Base** as a **LCN** segment coupler. Enter here the segment number in which **Base** operates as a segment coupler.
- **Events from other segments**: Select this field to allow the **Base** module to react on events from other segments. It also enables **Base** module to keep state of **LCN** modules from other segments.
- **Whitelist**: This field allows you to reduce the pool of **LCN** modules with which the **DOMIQ/Base** communicates. Fill this field only if necessary. Incorrect filling this field will result in partial or total lack of communication between the **Base** module and **LCN** installation. To reduce the poll of visible modules, you can use two different forms:
  - Numbers of modules separated with commas.
  - Groups of modules, such as 1–10, 20–31.
- **Time synchronization**: Specifies how often **DOMIQ/Base** will be send a command to the **LCN** modules to synchronize the time. The options are:
  - Off
  - 5 min
  - 15 min
  - 1 hour
- **Rescan LCN bus** : After pressing this button **DOMIQ/Base** module will reread the state of all **LCN** modules connected to the bus. Rescan is recommended after each change in the configuration or in the **LCN** structure done using **LCN–Pro**, especially if the configuration of temperature regulators or shutters was changed.

NOTICE!

The installers report many problems which appear after filling the configuration field **Visible modules**. This field should be used only in exceptional cases, e.g. **LCN** installations in which there are several independent appartments in one **LCN** segment, which is not generally recommended.

4.3. UPnP AV Configuration

The field **Active** controls switching on of the UPnP AV interface in the **Base** module. When this field is active it is impossible to control the UPnP devices, e.g. SONOS players. Changes in the configuration should be confirmed with the **Save** button. The changes will be made after the reboot of the **Base** module. The field **Active** is unchecked by default.

4.4. BACnet Configuration

From the software version 1.8.0.0 the **Base** module allows integration with **BACnet** system. The field **Active** works identically as in the case of LCN configuration, so it controls switching on the BACnet interface in the **Base** module. BACnet protocol support in **Base** is offered as an additional license. In the text box paste the purchased license number. In order to approve the changes in the configuration click the **Save** button.

4.5. Location

Setting the location is necessary for corrrect operation of built–in astronomical clock which is calculating the sunrise and sunset times.

The following options are available:

- **Timezone**: It allows to select a timezone in which the **Base** module operates. The following timezones are available:
  - **CET** – Central European Time
  - **WET** – West European Time
  - **EEK** – East European Time
  - **MSK**– Moscow Standard Time
- **City**: Place of installation of the **Base** module.
- **Contry**: Country where the **Base** module is installed.
- **Geocode**: This button is used to automatically calculate the geographical coordinates based on city entered in the **City** field.
- **Longitude**: This field will be filled automatically after the **Geocode** button is pressed. You can also enter the longitude manually.
- **Latitude:** This field will be filled automatically after the **Geocode** button is pressed. You can also enter the latitude manually.
- NOTICE! In order to ensure the proper operation of the astronomical clock, it is necessary to fill the fields **Longitude** and **Latitude** and select the correct time zone.

4.6. Remote Pairing

In this section you can generate pairing codes and add new licences for devices with **DOMIQ/Remote** application.

DOMIQ/Remote is an application for the **iPhone/iPad/iPod Touch**, which, through **DOMIQ/Base** allows you to control and visualize the state of a building automation system. The application is available for free on the **Appstore**.

DOMIQ/Remote Pairing

In order to connect the **DOMIQ/Remote** with **DOMIQ/Base**, a pairing code is required. One–time activation code is generated by pressing the **Refresh** button. The code is valid for 5 minutes.

Fig 4.3: Pairing code generation

Next run the application, click on the name of the **Base** module. In result you will see the pairing screen:

Fig 4.4: Pairing screen in the DOMIQ/Remote

Enter the generated pairing code and then click **Activate**.

DOMIQ/Remote Licences

The **Remote Authorizations** button when pressed shows a pop–up window that allows you to manage licences for the **DOMIQ/Remote** application. The window contains:

- The list of currently paired devices.
- The number of available **DOMIQ/Remote** licenses.
- The serial number of the **Base** module.
- The **Cancel all pairings** button. This button removes all currently paired devices. Once the device is removed from the list of paired devices it has to be paired again with the **DOMIQ/Base**.
- A text field for entering an authorization key for additional access licenses.

DOMIQ/Base is sold by default with a single license. This means that only one device with the **DOMIQ/Remote** application may be paired with the **Base** module.

To purchase additional licenses, please contact us by e–mail: **info@domiq.pl**. In the email it is necessary to specify **Base's** serial number.

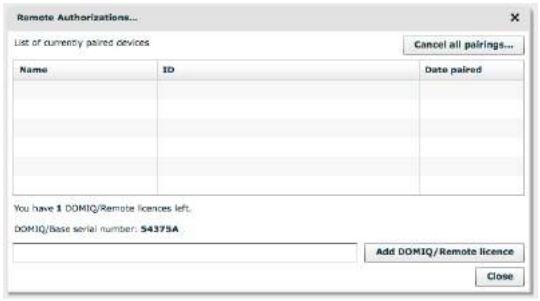


Fig 4.5: DOMIQ/Remote licences management window.

If the **Control over the Internet** box is checked, user can control the building automation system remotely, using the **DOMIQ/Remote** application. Otherwise, the building automation system can be controlled only in the local network to which the **Base** module is connected. In this scenario, **Base** notifications won't work.

If the **Control over the Internet** box isn't selected, it is possible to control the automation system remotely after opening TCP port 4554 in your router. In this case, you also need to configure the direct connection to the **Base** module in the **DOMIQ/Remote**.

4.7. System

Updates

Automatic update

Pressing the **Updates** button shows a pop–up window with a list of currently installed software and available software updates. The updates list is downloaded from the **DOMIQ** software updates server. If there is at least one software update available, it is listed in the **Available version** column. To initiate the update process, select the components that need to updated and then press the **Update selected** button.

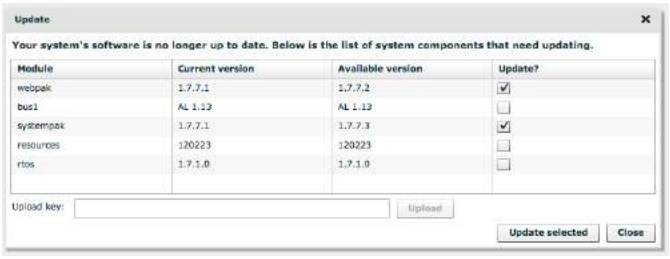


Fig 4.6: Updates window

Update process may take a few minutes. During update, the **DOMIQ/Base** module is unavailable. When update process is complete, the module reboots automatically and after that it is ready to use with updated software. Succesful update process is confirmed by following message:

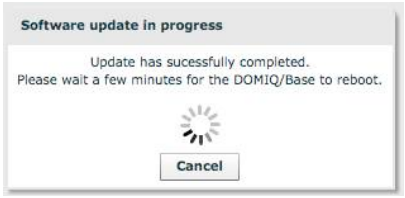


Fig 4.7: „Updates complete" message

Manual update

In case, when **DOMIQ/Base** module is not connected to the Internet, you can download the latest software to your computer (Internet connection required) from our webpage and then upload it to the module manually. To run manual update the upload keys are required. The upload keys are available on the updates web page, each update file has a separate upload key.

The manual update should be made in the following order:

- **rtos**
- **system**
- **web**

Update files are uploaded to the **Base** module using **Updates** dialog box. Paste the upload key in the **Upload key** field. Next click on the **Upload** button and browse for the udpate file from your computer. Repeat this procedure for the other update files.

When update process is finished please reboot the module by clicking on the **Restart** buton (**Settings** tab).

Restart

The **Restart** button is used to manually reboot the **Base** module. When pressed, the configurator ask you to confirm the restart command. After confirmation, all unsaved changes are discarded.

Date and Time

Pressing **Date and Time** button displays a pop–up window where you can set actual date and time. Setting correct date and time is essential for proper operation of timers.

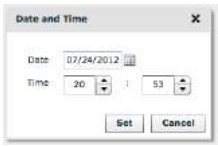


Fig 4.8: Date and time setting window.

Backup of the configuration

This section contains two buttons. The first is used to save a backup of your configuration on your computer. The **Restore from file** button allows to restore configuration from a previously created backup.

The backup file contains the following configuration data of the **DOMIQ** system:

- configuration of the interface in **DOMIQ**/**Remote** together with the pictures and icons used in the visualization,
- definitions of all events,
- definitions of all timers,
- logic script,
- **DOMIQ** configuration from **Settings.**

Chapter 5
State, Commands and Events

Knowledge the foundations of the operation of a **DOMIQ/Base** significantly simplifies understanding of the examples and descriptions presented in the remainder of this manual. This chapter describes only the most important information and two samples of using devices (identifiers). You can read more about identifiers in chapter 16. Identifiers

First of all, we will define some of the frequently used terms:

- **Subsystem** – one of the interfaces that you can integrate with the **DOMIQ** system, e.g. LCN, Satel.
- **Identifier** – a string of characters, that uniquely identifies a device or variable.
- **State** – actual state of an input, output or a variable with a given identifier.
- **Command** – a command that will be sent to a subsystem. The aim of a command is to change a devices's state. Commands are preceded by the `C.` prefix added to a **State** identifier.
- **Event** – data received from a subsystem, that informs about change of a device's state. Events are preceded by the `E.` prefix added to a **State** identifier.

The following diagram presents flow of information in a **DOMIQ/Base** module and illustrates the relationship of **State**, **Commands** and **Events**.

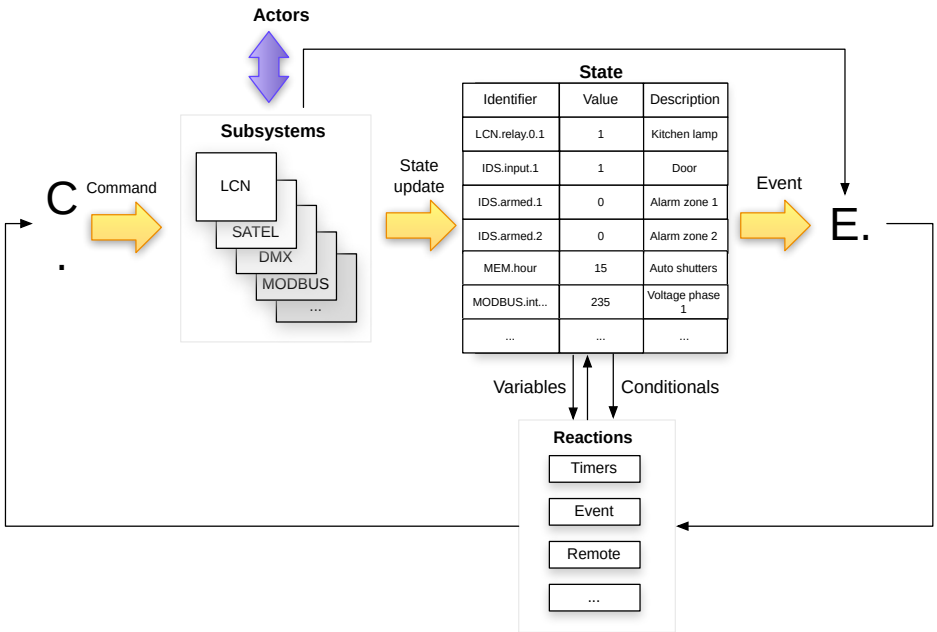


Fig 5.1: Information flow in the DOMIQ/Base

1. User clicks a button in the **DOMIQ/Remote** application. A button is responsible for controlling a kitchen lamp – it has assigned a proper **LCN** identifier.
2. A command is sent to the **LCN** subsystem, to turn the light.
3. An actor (in this case a **LCN** module) confirms turning on the light.
4. The **LCN** subsystem sends this information to the **State** (**State update**).
5. Information about turning on the lamp is passed through an event to the **Remote** application, which changes, for example the look of a light control.

State of all identifiers is available in the **State** tab. Using the **State** tab allows for quick overview of value of a desired device or variable, for example, value of a dimmable output, alarm sensor or a relay.

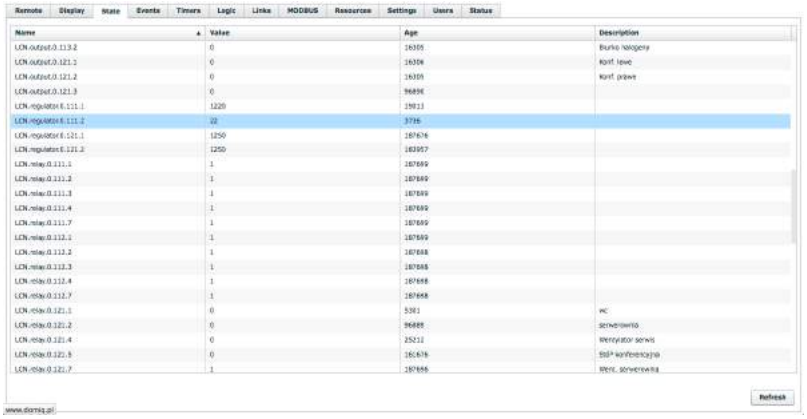


Fig 5.2: The State tab overview

Each device/variable is represented in the **State** tab by four attributes: **Name**, **Value**, **Age**, **Description**.

- The **Name** column contains names of all currently available identifiers.
- The **Value** column contains a current state of all devices and variables.
- The **Age** presents time since last change of a identifier's state.
- The **Description** column contains description of devices. This attribute is optional. In case of **LCN** modules, the **Description** column contains descriptions given in the **LCN–Pro.** In case of the **Satel** alarm system, the **Description** column contains descriptions created during configuration of the control panel.

The **State** tab is not updated automatically, in order to refresh the state, click the **Refresh** button.

5.1. Exemplary Identifiers

DOMIQ/Base module has a set of predefined identifiers. This section describes two of the most often used predefined identifiers, which is sufficient to understand further examples and how to use identifiers in general. A detailed description of all available identifiers, with use examples is included in the chapter 16.Identifiers.

LCN.output

This identifier is used to control and display state of a single **LCN** dimmer. It also informs about changes of value of an output. It's one of the most often used identifiers.

	Identifier	Value	Description
State	`LCN.output.`**`S`**`.`**`M`**`.`**`O`** **S** – Segment **M** – Module **O** – Output	`0–100`	Percentage value of a dimmable output.
Event	**`E`**`.LCN.output.`**`S`**`.`**`M`**`.`**`O`**	`0–100`	Percentage value of a dimmable output.
Command	**`C`**`.LCN.output.`**`S`**`.`**`M`**`.`**`O`**	`0–100`	Set value
		`on`	Turn on
		`off` `change:xx`	Turn off Relative percentage change of the brightness. Negative value means dimming.

- `C.LCN.output.0.10.1=on`
  Turn on the output No. 1, the module address: 10.
- `C.LCN.output.0.10.1=30;ramp:10`
  Set value of the output No.1 to 30%, ramp=10, the module address: 10.
- `E.LCN.output.0.10.1=100`
  The output No.1 in a module with the address 10 has turned on.
- `LCN.output.0.10.1`
  State of the output No.1 in a module with the address 10.

LCN.relay

The `LCN.relay` identifier is used to control and display state of a single relay. It also informs about changes of relay's state

	Identifier	Value	Description
State	`LCN.relay.`**`S`**`.`**`M`**`.`**`P`** **S** – Segment **M** – Module **R** – Relay	`0`	Relay off.
		`1`	Relay on.
Event	**`E`**`.LCN.relay.`**`S`**`.`**`M`**`.`**`P`**	`0`	Relay has been turned off.
		`1`	Relay has been turned on.
Command	**`C`**`.LCN.relay.`**`S`**`.`**`M`**`.`**`P`**	`0,1`	Set value.
		`toggle`	Toggle value.
		`on`	Turn on
		`off`	Turn off

- `C.LCN.relay.0.10.3=on`
  Turn on the relay 3, the module address: 10.
- `C.LCN.relay.0.10.5=toggle`
  Toggle the relay 5, the module address: 10.
- `E.LCN.relay.0.10.1=0`
  The relay 1 has been turned off, the module address: 10.
- `LCN.relay.0.10.2`
  State of the relay 2, the module address: 10.

5.2. Delays

Functions from the **Delays** group allows to define simplified time scenarios, involving execution of certain activities with specified delay.

Delays are complementary to timers (see „Timers" on page 83)

DELAY

DELAY allows to send any commands with specified delay. Commands will be executed as many times as the DELAY function is called.

The DELAY function is typically used for shutters positioning in installation without **LCN–BS4** module.

Function syntax

`DELAY.delay.action=V`, where

`delay` – A delay after which given action is executed. The delay can be expressed in hours, minutes and seconds and milliseconds, following abbreviations are valid: h, m, s, ms.

`action` – Action to be executed after the specified delay. In most cases, the actions are commands.

V – value which will be sent to a given identifier.

- `DELAY.30s.C.LCN.output.0.10.1=100`
  After 30 seconds turn on the first output in a LCN module with the address 10.
- `DELAY.3h.C.IDS.armed.1=1`
  After 3 hours arm the alarm zone No.1.

TIMER

DELAY allows to send any commands with specified delay. In contrast to the DELAY, each time the TIMER is called, **Base** module checks the moment of call and performs only the latest call of the TIMER function. It is possible to cancel a particular TIMER by setting a delay to 0.

A typical application is the implementation of stairway lighting control, shutters control, automatic arming of alarm zones, etc.

Function syntax

`TIMER.name.delay.action=data`, where

`name` – Any name of a timer (without spaces).

`delay` – A delay after which given action is executed. The delay can be expressed in hours, minutes and seconds and milliseconds, following abbreviations are valid: h, m, s, ms.

`identifier` - Action which is executed after a given delay. In most cases the actions are commands.

`data` - the value of the identifier to which the command is sent. In case of events it is the information regarding the actual value of the identifier.

Please bear in mind that the sequence `identifier=data` will always have the form of a command, so it will start from `C.`

- `TIMER.kitchen.30s.C.LCN.relay.0.10.1=on`
  After 30 seconds turn on the first relay in a LCN module with the address 10. If another call (within specified delay) to `TIMER` occurs then only the latest call will be executed.
- `TIMER.alarm1.1h.C.IDS.armed.1=1`
  Arm the zone No. 1 after an hour.
- `TIMER.alarm1.0=0`
  Cancels the timer from the previous example.

5.3. LOGIC

The `LOGIC` identifier allows to execute any fragment of Lua script. Operation of this identifier can be both: call to function created in the **Logic** tab or a modification of already existing code such as variable assignment.

	Identifier	Value	Description
Command	C.LOGIC	`Lua code`	Allows to execute any fragment of Lua code in context of a code defined in the **Logic** tab.

- `C.LOGIC=myFunction()`
  Invoke and execute `myFunction`.
- `C.LOGIC=a=1`
  Assign `1` to the global variable `a`. If a variable has not been yet declared, it will be created when the command is invoked. If the variable already exists then the command will change only its value.

5.4. Custom Identifiers

DOMIQ/Base module in addition to predefined identifiers, allows users to create their own custom identifiers. Custom identifiers are used to create more sophisticated logical rules: complex events, commands sequences (macros) or timers. Names of custom identifiers cannot contain spaces or national characters. For more complex names use the underscore ("_") or camel case. Own identifiers have the same properties as the predefined identifiers: they can have any value, value be changed using commands. State changes are reported by events.

Keep in mind that your own identifiers do not automatically have a state – using them as your own commands or events do not require programming in **Logic**, however defining a state requires to program proper procedures.

A good example of using own identifiers is the tutorial: „Shutters control". It includes a lot of examples of use of user–defined identifiers for shutters control using the **DOMIQ/Base** module..

- `C.Test=10`
  Send own command. Assign 10, to the `Test` identifier
- `C.shutters.groundfloor.kitchen=up`
  Send `up` command to the `shutters.groundfloor.kitchen` identifier. In the present form this command will not execute any action unless you assign commands to it, for example controlling the shutters in the kitchen.
- Send `on` command to the `shutters.groundfloor.kitchen` identifier. In the present form this command will not execute any action unless you assign commands to it, for example controlling the shutters in the kitchen.

Chapter 6
Remote

The **Remote** tab allows you to configure user interface and define a structure of the **DOMIQ/Remote** application. Each iPhone/iPad/iPod connected to the **DOMIQ/Base** module uses the same configuration to create a menu structure.This structure is defined on this page.

The pairing process of the **DOMIQ/Remote** with the **DOMIQ/Base** was described in the chapter „Remote Pairing" on page 21.

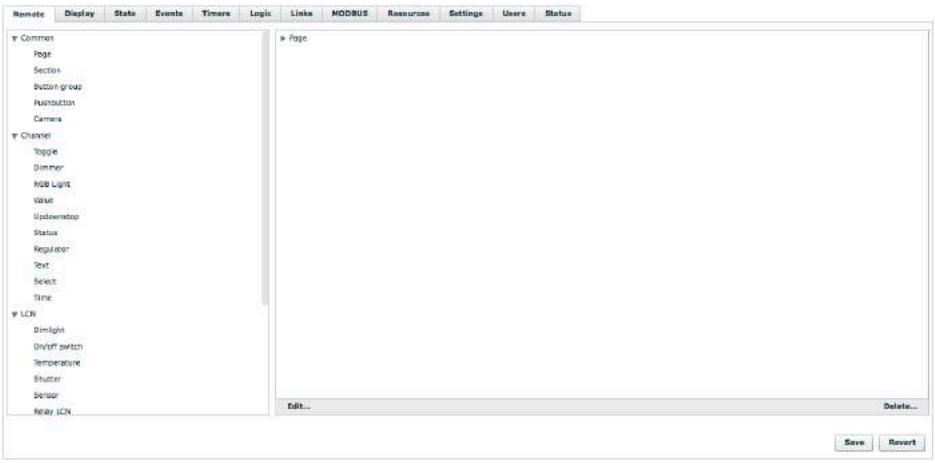


Fig 6.1: The configuration window of the DOMIQ/Remote application.

On the left side, there is a list of all available elements that can be used to build user interface. Elements are divided into five major groups:

1. **Common**
2. **Channel**
3. **LCN**
4. **IDS**
5. **Deprecated**

On the right side there is a tree view representation of the currently defined user interface elements.

Following are a few important guidelines about usage of the **Remote** editor:

- New elements in the tree view are added by selecting desired interface element from the list on the left side of the page and dragging and dropping it on the chosen location in the tree view on the right side of the page.
- Properties of every element in the tree view can be modified by either double clicking it or selecting it and clicking on the **Change...** button.
- Deleting an element in the tree view requires selecting it and then pressing the **Delete...** button. If the deleted element has hierarchy of elements underneath these elements will be deleted too.
- Elements in the tree view can be rearranged by dragging and dropping elements into the desired position.

- In order to copy instead of move, the SHIFT button should be held pressed during the drag and drop operation.

The following chapter provide detailed description of all user interface elements available in the **DOMIQ/Remote** application.

6.1. Common

Page

Page is the **DOMIQ/Remote** basic user interface element that corresponds to the entire page displayed by the **DOMIQ/Remote** running on the iPad/iPhone/iPod. Each page must have a label assigned. Each page can have a list of commands defined that will be executed when the user shakes the iPhone/iPad/iPod while the **DOMIQ/Remote** application is running and the corresponding page is shown.

Fig 6.2: Page edition view

The top level pages in the user interface hierarchy are selected by pressing buttons in the lower button bar in the **DOMIQ/Remote** application. It is recommended to define an icon for these pages.

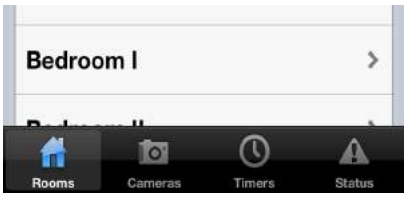


Fig 6.3: Mapping of pages on the icon bar

Section is the only user interface element that is allowed as direct child node of the **Page** element. Multiple section can be added to a single page.

Fig 6.4: Mapping of the configuration structure in the DOMIQ/Remote application

Section

Section allows arranging several user interface element into one group, for example lighting group, ventilation group etc. Section may, but does not have to have a label assigned.

Fig 6.5: Section edition view.

Fig 6.6: Examplary section for lighting control.

Sections can have nested pages. This allows to create more complex user interface structures. Nested page is indicated by an angle bracket.

Fig 6.7: An example of nested pages.

Button group

The Button group element allows to arrange several pushbuttons into one group. The **Button group** can be added only as a immediate child of a section. After double–clicking on it, you can choose the number of columns in which the buttons will be grouped. Values from 1 to 8 are available. For example if you choose *1*, buttons will be placed one below another, as shown in the first picture. If you choose *2*, then buttons will be arranged in two columns (see the second picture) and so on.

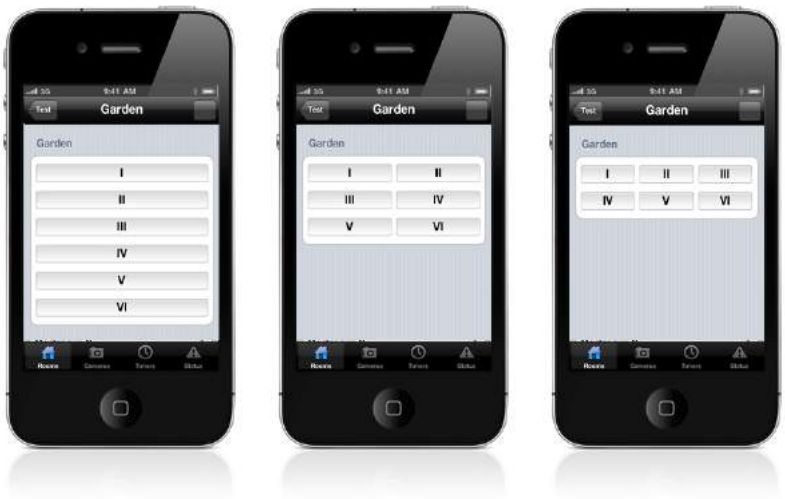


Fig 6.8: Button group examples

Pushbutton

Pushbutton executes a sequence of commands, depending on state of a button. There are three states: **Hit**, **Hold**, **Release** (analogically to LCN buttons). You can define a separate sequences for each state.

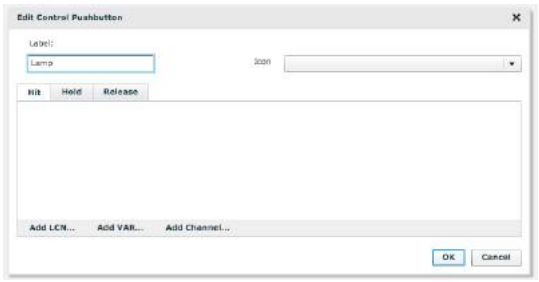


Fig 6.9: Pushbutton edition view

Pushbutton can be added as a subelement of both a **Button group** or a **Section.**

The definition of this user interface element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Icon**: You can choose an icon to be displayed on a button.
- **Sequence of commands**: A list of command to be executed depending on state of a button.

Camera

The **Camera** user interface element shows the real–time video preview from an IP camera. The definition of this user interface element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **URL**: The URL pointing directly to camera's Motion JPEG video stream. The exact notation depends on the specific IP camera manufacturer and model, and usually can be found in the camera's user manual.
- **Username**: The username used to login to the camera.

- **Password**: The password used to login to the camera.
- **Type**: Possible values are: Generic MJPEG, Vivotek, Axis, Mobotix, Generic JPEG.

Username and **Password** are optional. They should be provided, if the camera requires authorization to access video stream.

- **Pan–tilt control**: This option should be checked if camera has pan–tilt head and it should be possible to control it from the **DOMIQ/Remote**.
- **Zoom**: This option should be checked if camera has adjustable zoom lens and it should be possible to control it from the **DOMIQ/Remote**.

PTZ URI commands are available only for cameras with moveable heads. Moving a finger accross the screen of a device with the **DOMIQ/Remote** will cause movement of a camera's head. Refer to the camera manual for specific commands.

- **Left**: The command to move camera to the left.
- **Right**: The command to move camera to the right.
- **Up**: The command to move camera up.
- **Down**: The command to move camera down.
- **Narrow**: Feature unavailable.
- **Wide**: Feature unavailable.

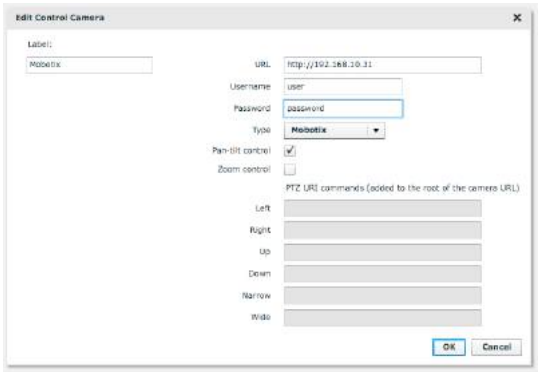


Fig 6.10: Camera edition view

6.2. Channel

Interface elements from the **Channel** group are used to control building automation using identifiers.

Toggle

Toggle is one of the most often used interface element, which allows to set state of a single identifier to **on** or **off**. The **Toggle** element can be used to control, for example a relay, dimmable output or to set values of MEM or VAR–type variables etc.

The definition of this element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Channel**: The device identifier that you want to control.

- `LCN.relay.0.36.1`
  Controlling the relay 1 in a LCN modulue with the address 36.

Fig 6.11: Toggle edition view

- **Reverse value**: Check this box to reverse the logic of a switch.

Fig 6.12: Toggle in the DOMIQ/Remote

Dimmer

Dimmer is one of the most often used user interface elements. It allows controlling a single identifier.

The definition of this user interface element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Channel**: The device identifier, whose proportional output you want to control.

- `LCN.output.0.112.1`
  Controlling the dimmable output 1 in a LCN module with the address 112.

This element is usually added as a subelement of the **Section**.

Fig 6.13: Dimmer edition view.

Depending on device, the look of the **Dimmer** element differs. On iPad, the **Dimmer** is represented as 2 controls: slider allows quick setting of the brightness. The second control allows to control the brightness more precisely and it has also the function of a traditional switch.

Fig 6.14: The look of the Dimmer element on iPhone and iPad.

RGB Light

RGB Light lets you control RGB lighting, such as RGB LED lighting using **DOMIQ/Serial-4DX** module in combination with DMX512 dimmers, for example **DAGON Lighting SPL1 or SPL3**. It's also possible to control RGB lights connected to dimmable outputs of **LCN** modules.

The definition of this user interface element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Red channel**: The device identifier, which controls the red component.

- **Green channel**: The device identifier, which controls the green component.
- **Blue channel**: The device identifier, which controls the blue component.

Fig 6.15: RGB Light edition view.

The **RGB Light** control consist of:

1. **Brightness slider** – using this slider you can adjust LED's brightness. To turn LED lighting off, move the slider to the leftmost position. When lighting is off **Color selection ring** is inactive.
2. **Saturation slider** – using this slider you can adjust LED's saturation.
3. **Color selection ring** – using this ring you can select any RGB color. The **color selection ring** is inactive, when brightness is set to 0.
4. **Circle of the actual color** – this circle shows the current color including brightness and saturation.

Fig 6.16: RGB Light on iPhone.

Fig 6.17: RGB Light on iPad.

For more practical examples about configuration and controlling LED lighting refer to the tutorial „**RGB LED with DMX**. Tutorial is available on our webpage **www.domiq.eu**, **Tutorials** section.

Value

The **Value** user interface element is used to set and display current value of any identifier. Additionally when pressed a new page is presented with a selector for changing the value of given identifier .

The definition of this element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Channel**: The device identifier, whose value you want to change and display.
- **Min value**: Minimum value that the identifier can be set to
- **Max value**: Maximum value that the identifier can be set to
- **Base value**: Mesured value will be displayed relative to this value.
- **Units**: Unit of measured value.
- **Scaling factor**: Measured value will be multiplied by value entered in this field.
- **Decimal places**: Here you can determine precision of displayed value.
- **Step**: Minimum change value of the controlled variable.

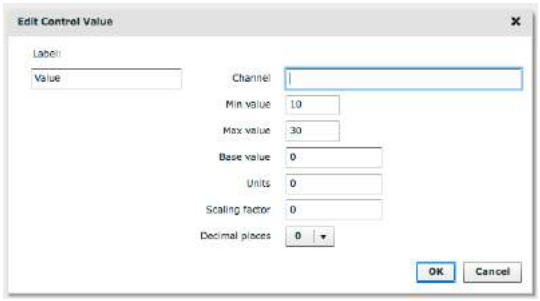


Fig 6.18: Value edition view

Updownstop

The **Updownstop** element is used for controlling shutters using identifiers. This approach allows to implement more advanced algorithms for shutters control, such as controlling groups of shutters, adding events or timers to control shutters etc.

You can read more about controlling shutters in the tutorial entitled„**Shutters control**". Tutorial is available on **www.domiq.eu/tutorials**.

The definition of this element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.

- **Channel**: The device identifier that you want to control.

Fig 6.19: Updownstop edition view

- `LCN.motor.0.36.1`
  Controlling shutters using the `LCN.motor` identifier.
- `shutters.groundfloor`
  Controlling a group of shutters using own identifier named: `shutters.groundfloor`.

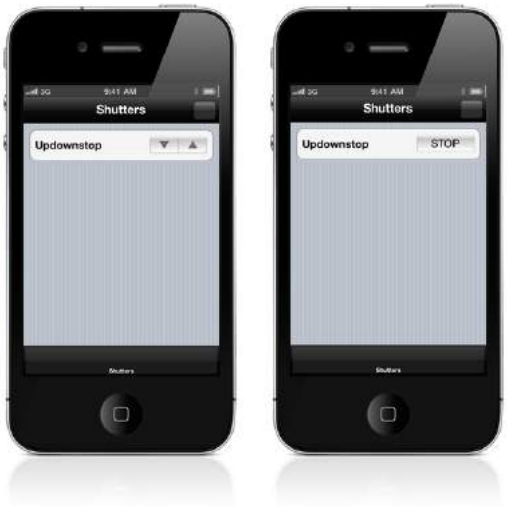


Fig 6.20: Updownstop element in the DOMIQ/Remote application

Status

The **Status** element is used for graphical representation of state of any identifier, for example dimmable output, relay, IDS sensor, logical variable etc. The look the **Status** element is defined during configuration of the control.

The definition of this element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Channel**: The device identifier, whose state you want to display.
- **Image off**: Graphical representation of an identifier whose state is off.
- **Image on**: Graphical representation of an identifier whose state is on.

Fig 6.21: Status edition view

Fig 6.22: Example of usage of the Status element to visualize state of an alarm zone

Regulator

The **Regulator** element can be used to control value of any variable, such as temperature (heating, air conditioning). The **Regulator** element can operate in combination with **LCN** regulator or in combination with Lua scripts created in the **Logic** tab.

The definition of this element consist of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Channel**: The identifier of a device keeping set value of the regulator. It can be an **LCN** regulator, e.g. `LCN.regulator.0.113.2` or a variable declared in the **Logic** tab, e.g. `MEM.temperature`.
- **Channel with actual value**: The identifier of device storing current value of the regulator.

Fig 6.23: Regulator edition view.

Fig 6.24: The look of the Regulator in the DOMIQ/Remote.

Text

The **Text** element was designed to display any text content, such as captions, measured values, identifiers state, logical variables etc.

The definition of this element consists of two fields:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Channel**: The device identifier, whose state you want to display.

Fig 6.25: Text edition view

Select

Select is dedicated to create interface elements, where user can choose one or more desired options.

The definition of this element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Channel**: The identifier, where information about selected options will be stored.
- **Mandatory**: When this box is checked, user has to choose at least one option from a given list.
- **Multiple**: This option allows to select multiple elements from a given list.

The **Select** element is commonly used in timers definition. Good example of using the **Select** element is the weekday selection. After you define a table with names of weekdays, user can select the days of a week, when a given action or actions will be performed, e.g. raising or lowering shutters.

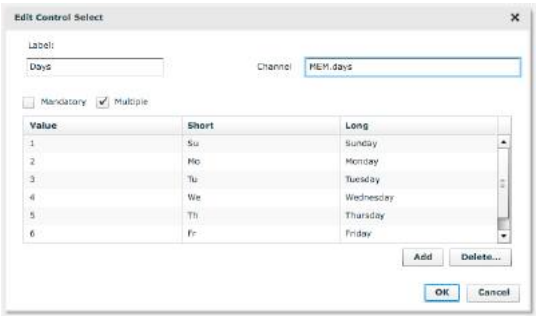


Fig 6.26: An example of usage of the Select element to choose weekdays

Fig 6.27: Result in the Remote application.

Time

The **Time** element is dedicated to create timers using **DOMIQ/Remote** application. Using this control, user can choose hour and minute, when given action or sequence of actions will be triggered. The **Time** element in combination with the **Select** allows to define more complex timers, such as timers triggered at a specified time and on specified days. Complete example of using the **Time** element is described in chapter „Timers" on page 83.

The definition of this element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Hour channel**: The identifier where selected hour will be stored, typically: `MEM.<name>`, e.g. `MEM.hour`. MEM–type variable is a non–volatile variable, so its value is stored even after power failure or system restart.
- **Minute channel**: The identifier where selected minute will be stored, typically: `MEM.<name>`, e.g. `MEM.hour`.

Fig 6.28: Time element edition window

Fig 6.29: The look of the Time element in the DOMIQ/Remote.

6.3. LCN

The **LCN** group contains elements dedicated to use in combination with the **LCN** system.

Dimlight

Dimlight is one of the most often used user interface elements. It allows controlling a single proportional output of a **LCN** module.

The definition of this user interface element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Destination**: Segment number and ID of a LCN module. By clicking on the **Select destination** button, you can choose destination module from a list of all modules available in installation.
- **Output**: The number of a proportional output, that you want to control. Available values: **1**, **2**, **3** and **4**.

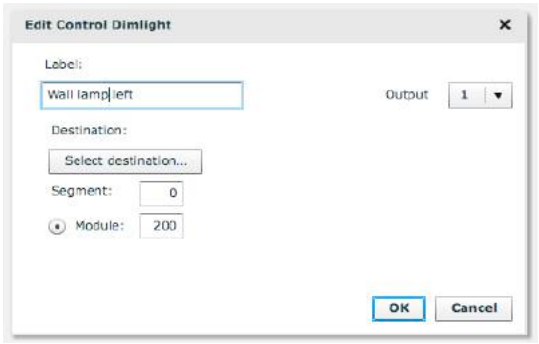


Fig 6.30: Dimlight edition view.

The look of this control is identical to the **Dimmer** element.

On/off switch

On/off switch allows to define multifunctional user interface element. Depending on option selected in the **Type** field, the **On/off switch** operates in a different way. The declaration of the element also differs for each type. The following types are available: **Relay**, **Output**, **Bit**, **IDS Output**.

Relay

If you choose **Relay**, then **On/off switch** can control a single LCN relay.

The definition of this user interface element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Destination**: Segment number and ID of a **LCN** module. By clicking on the **Select destination** button, you can choose destination module from a list of all modules available in installation.
- **Relay**: Relay number you want to control. Available values: from **1** to **8**.
- **Reverse value**: Check this box to reverse the logic of a switch.

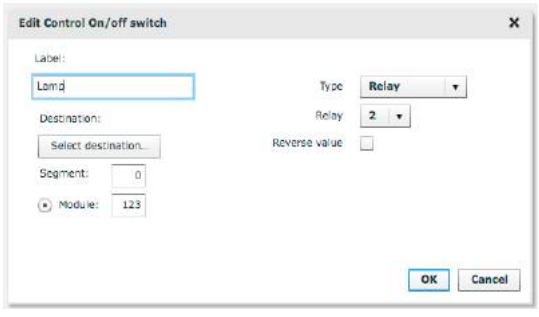


Fig 6.31: On/off switch (relay–type) edition window.

Output

If you choose **Output**, then **On/off switch** can control a single **LCN** proportional output, but without dimming.

The definition of this control consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Destination**: Segment number and ID of a LCN module. By clicking on the **Select destination** button, you can choose destination module from a list of all modules available in installation.
- **Output**: The number of a proportional output, that you want to control. Available values: **1**, **2**, **3**.
- **Ramp**: The time at which an output reaches a set value. Range: from **0** to **20**.
- **Threshold**: Output value at which an output is considered turned on. Available range: from 1 to 100.
- **Reverse value**: Check this box to reverse the logic of a switch.

Fig 6.32: On/off switch (output–type) edition window.

Bit field

If you choose **Bit field**, then **On/off switch** can control a single bit field.

The definition of this user interface element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Bit field number**: Available values: **1** to **256**.
- **Reverse value**: Check this box to reverse the logic of a switch.

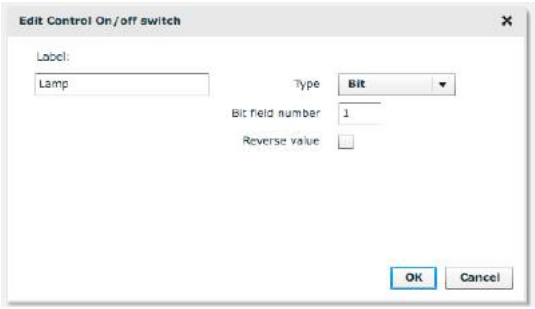


Fig 6.33: On/off switch (output–type) edition window.

IDS Output

If you choose **IDS Output**, then **On/off switch** can control a single output of the **Satel** alarm panel.

The definition of this user interface element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Reverse value**: Check this box to reverse the logic of a switch.
- **IDS output #**: The number of an alarm output you want to control.
- **Default PIN**: The PIN code of the alarm panel. **Satel** alarm panel requires authentication each time you want to change state of an output.

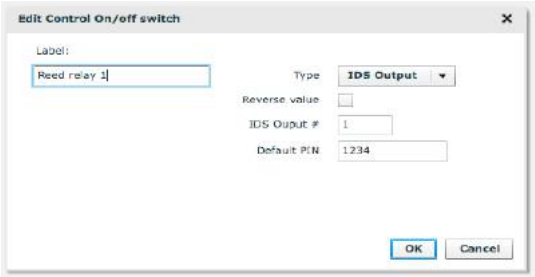


Fig 6.34: On/off switch (IDS output–type) edition window.

Fig 6.35: On/off switch in the DOMIQ/Remote application. The look of the control is the same for each type.

Temperature

Temperature user interface element displays the current and requested temperature. Additionally when pressed a new page is presented with a chart showing actual temperature changes during the last 24 hours and a selector for changing the requested temperature (only if **Control type** is the **Regulator**).

The definition of this user interface element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Destination**: Segment number and ID of a **LCN** module. By clicking on the **Select destination** button, you can choose destination module from a list of all modules available in installation.
- **Temp. sensor**: Variable in the **LCN** module that holds the current temperature. Possible values: **TVar**, **R1Var** or **R2Var**.
- **Control type**: Available options: **Regulator**, **Thresholds**. If you choose **Regulator**, then the temperature will be controlled using a proportional LCN regulator. Whereas, if you choose **Thresholds**, the temperature will be controlled using threshold values set in the **LCN–Pro**.

- Set treshold 1 to value of 22°C and treshold 2 to 25°C. When the temperature drops below the first threshold the heating will be activated. However, when the temperature rises to 25°C, the heating will be deactivated.

- **Regulator** (only when **Control type** is set to **Regulator**): Regulator used to set the requested value. Possible values: **Regulator 1** or **Regulator 2**
- **Threshold value #** (only when **Control type** is set to **Thresholds**): The number of a threshold set in the **LCN–Pro**.
- **Min value**: Minimum value that the requested temperature can be set to
- **Max value**: Maximum value that the requested temperature can be set to

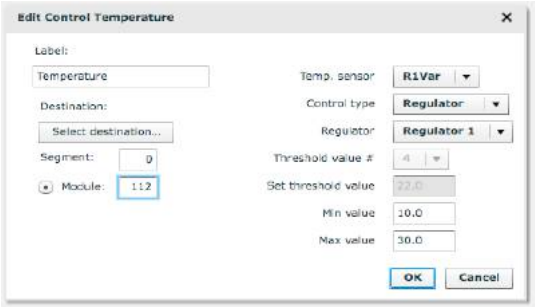


Fig 6.36: Temperature edition window

Fig 6.37: The look of the Temperature element in the DOMIQ/Remote.

Shutter

Shutter user interface element allows controlling shutter motors.

The definition of this user interface element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Destination**: Segment number and ID of a **LCN** module. By clicking on the **Select destination** button, you can choose destination module from a list of all modules available in installation.
- **Type**: The method of controlling the shutter motors. The following options are supported:
  - Relays (without positioning)
  - Relays (with positioning)
  - Analog output
- **Motor**: Motor number (in an **LCN** module) that positions the shutter (present only when relays are used to control shutter motors). The following options are available:
  - **1**,**2**,**3**,**4** – controlling a single shutter;
  - **1+2**, **3+4** – controlling pairs of shutters
  - **1–4** – controlling 4 shutters simultaneously (8 relays)

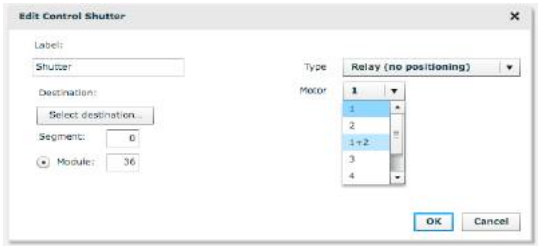


Fig 6.38: Shutter edition view

The look of this control is the same as the look of the **Shutter** element included in the **Common** group.

LCN binary input

LCN binary input shows current status of a single **LCN** binary input.

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Destination**: Segment number and ID of a **LCN** module. By clicking on the **Select destination** button, you can choose destination module from a list of all modules available in installation.
- **Sensor number**: The number of a **LCN** binary input that's state is presented. Possible value is from **1** to **8**
- **Image on**: Representation of the **on** value.
- **Image off**: Representation of the **off** value.

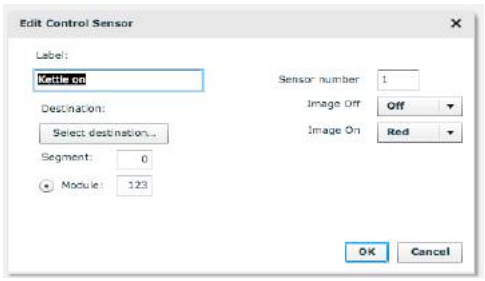


Fig 6.39: Sensor edition window.

LCN Relay

LCN Relay is used to show current state of a single LCN relay.

The definition of this control consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Destination**: Segment number and ID of a **LCN** module. By clicking on the **Select destination** button, you can choose destination module from a list of all modules available in installation.
- **Sensor number**: The number of a **LCN** relay that's state is presented. Possible value is from **1** to **8**
- **Image on**: Representation of the **on** value.
- **Image off**: Representation of the **off** value.

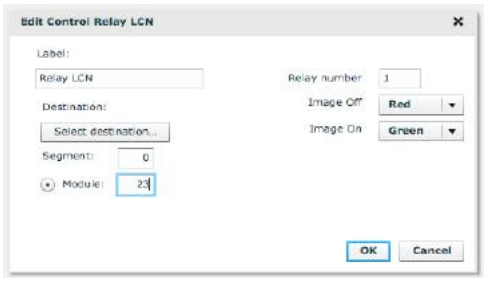


Fig 6.40: Relay LCN edition window.

6.4. IDS

IDS (Intrussion Detection System) elements are dedicated to use in conjuntion with the **SATEL Integra** intrusion detection system.

IDS input

IDS input user interface element shows current status of a single alarm input (PIR sensor, reed relay etc.).

The definition of this user interface element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Input number**: The number of a IDS input that's state is presented. The number of available inputs may differ depending on alarm panel type and used additional extension modules.
- **Image on**: Representation of the **on** value.
- **Image off**: Representation of the **off** value.

Fig 6.41: IDS input edition window.

IDS output

IDS input user interface element shows current status of a single alarm output.

The definition of this user interface element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Output number**: The number of a IDS outpus that's state is presented. The number of available outputs may differ depending on alarm panel type and used additional extension modules.
- **Image on**: Representation of the **on** value.
- **Image off**: Representation of the **off** value.

Fig 6.42: IDS outpus edition view.

IDS zone

IDS zone shows current state of a single alarm zone and allows arming or disarming of the selected zone. Additionally when pressed a new page is presented with a keypad to enter your PIN code (arming/disarming alarm zone).

Fig 6.43: The look of the IDS zone in the DOMIQ/Remote application.

Depending on status of an alarm zone, the look of the **IDS zone** control is different: There are five different states, signaled as follows:

1. **Gray indicator light** – zone disarmed.
2. **Green indicator light** – zone armed.
3. **Blinking red** – alarm.
4. **Flashing gray/green** – exit time.
5. **Flashing gray/red** – entry time.

The definition of this user interface element consists of:

- **Label**: A brief description of the element displayed in the **Remote** application.
- **Zone number**

Fig 6.44: IDS zone edition window.

Chapter 7
Display

The **Display** editor is used to create interactive visualizations, so you can easily control home automation system. Creating a visualization is very simple and does not require any programming knowledge. Controlling house with a visualization can be done in three ways:

- Using the **iPhone/iPad/iPod Touch** with **DOMIQ/Remote** application. The visualization is shown after turning the device to the horizontal position.
- Using the **DOMIQ/Display** touch panels. It is possible to use multiple panels, each of them can display different default visualization.
- With any internet browser. To do this, enter the IP address of the **Base** module in the browser. In the login window enter the name and password and then press the **Visualization** button.

All configuration changes made using the **Display** tab are cached in web browser and do not affect the operation of the **DOMIQ** system until the **Save** button was pressed. The **Revert** button, restores last saved configuration of the **Modbus** tab.

To display changes made in the **Display** tab on the **Display** panel, click on the **Save** button and then on the **Restart**.

7.1. Display Editor window

Display editor window is divided in two main parts: the **Editing part** and the **Visualization window**.

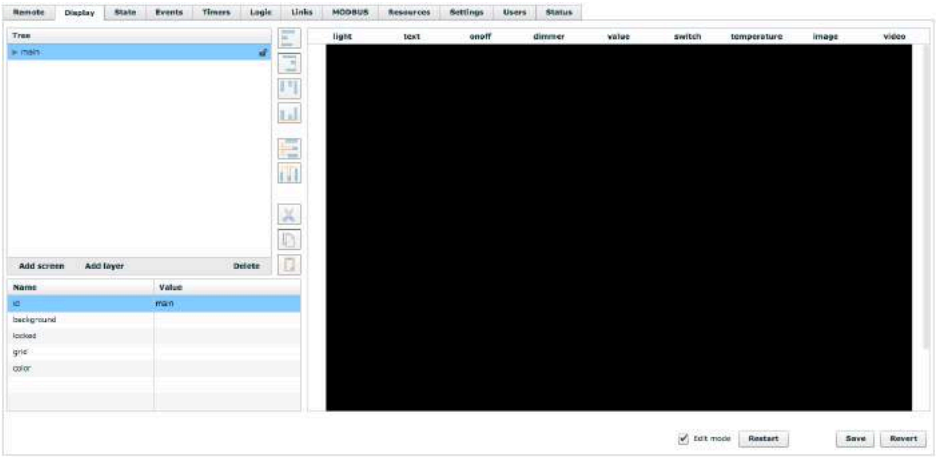


Fig 7.1: Display editor window

Editing Part

In the editing part you can add new elements to the visualization structure. You can also edit the properties of already used items.

The **Structure** view shows tree representation of the currently defined user visualization structure. Basic elements of visualization are screens, layers and elements.

Screen

Screen is a basic element of the visualization structure that corresponds to the entire screen displayed on a visualization. To add a new **Screen,** click the **Add screen** button. By clicking on the name of a screen, you can edit its settings:

- Assign an ID.
- Choose a visualization background.
- Set the lock
- Define the grid

Setting the lock for the screen means that the visualization elements added directly to the screen can not be moved, but you can still edit their parameters. You can also enable (disable) the lock by clicking the padlock icon on the right side of the screen ID. Switching on a screen lock does not lock the layers added to the screen.

The **grid** parameter creates an invisible grid, on which the visualization elements are arranged. Typing 5 in the net field means that the minimum movement step of the visualization elements is 5 pixels.

The screens structure can be freely changed by draging a screen and droping it in the desired location on the list.

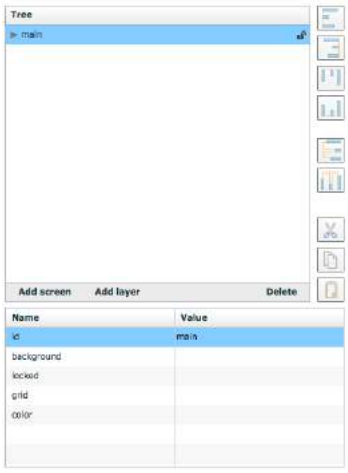


Fig 7.2: Display Editing Part

To add your image, that will be used as a visualization background, you have to upload it to the **Base** module (the **Resources** tab). Max resolution is 800x600 pixels. A file must be either JPEG or PNG file and may not be be larger than 120KiB. The name of a file cannot contain spaces.

Layer

Layer is an element of the visualization structure that can be only added as a child element of a **Screen**. Layer allows to group elements of the same type e.g. switches, light etc. We recommend you to use layers as often as it is possible. This approach allows to keep order in the structure and facilitates the management of elements added to the visualization.

To add a new layer, click on the screen ID and then on the **Add layer** button. After selecting the new layer, you have to enter a unique identifier. You can also, as for the screens, turn on the lock and the grid. The **visible** parameter controls the visibility of the layer. Visibility can also be turned on/off by clicking the eye icon on the right side of the layer ID.

Layers structure can be freely changed, the same as for screens. However, the layers structure has an impact on displaying layers on a visualization. The layers are displayed in reverse order than they are placed in the structure. In other words, the deeper a layer is in the structure, the higher it is displayed on the visualization. For example, if the layer is the deepest in structure, it will be displayed on top of a visualization.

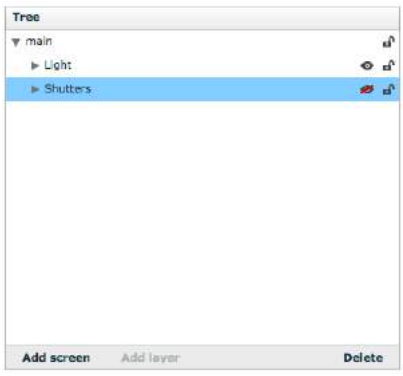


Fig 7.3: Example of layers structure

Visualization Window

Visualization window is the graphic part of the **Display** editor. Here you create the graphic look of a visualization by adding active elements and arranging them on a screen. All the active elements of visualization are grouped in the bar above the visualization window. For the remainder of this manual we will call this bar the **active elements bar**. New element is added to a visualization by draging it from the **active elements bar** and dropping it on the **visualization window**.

Fig 7.4: The Active elements bar

On the left side of the visualization, there is a set of buttons for positioning the active elements of visualization:

Align left	L
Align right	R
Align top	T
Align bottom	B
Align the vertical spacing	V
Align the horizontal spacing	H
Cut	X
Copy	C or SPACE
Paste	P or ENTER

Copied/cut element is always pasted above element that you copied/cut.

To select multiple elements of visualization you have to click on them with the Shift button pressed. Visualization elements can be moved using the keyboard arrow buttons. Each pressing moves the control by one pixel.
The **Edit mode** check box toggles the visualization modes between editing and testing. In test mode, the **Editing part** is deactivated, while the **visualization window** simulates the operation of visualization.
Visualization elements can be removed in two ways: by selecting the control in the **visualization window** and then pressing the Delete or Backspace button on your keyboard or by selecting an item in the visualization structure and pressing the **Delete**.

7.2. Active Elements of Visualization

In this section we describe each element of the **Active elements bar**. As a quick reminder, the **Active elements bar** is placed just above the **Visualization window**.

Light

The **Light** element is one of the most often used elements in a visualization, and also the most universal control. Typically a **Light** is used to control lightning. But the **Light** Element can also display current state of any identifier such as zone alarm, relay, sensor status (reed relay or PIR), etc.

To add a new **Light** element, drag it to the **visualization window**. By clicking on it you can set its properties in the **editing part**.

The following attributes are available:

- **X** i **Y**: Coordinates of the element. The values of **X** and **Y** can be changed by entering these values from the keyboard or simply moving the element in the visualization window by holding down the left mouse button.
- **Height** i **Width** (read only): The height and width of a visualization element
- **Layer:** The layer on which you placed the element.
- **Channel:** The name of the indentifier that you want to control and whose status will be displayed on the visualization screen.

- `LCN.relay.0.121.1`
  Control the relay 1 in a LCN module with the address 121 and display its status.
- `LCN.output.0.113.1`
  Control the dimmer 1 in a module with the address 113 and display its status.

If the **On** and **Off** fields are filled, then the **Channel** field is used only to display state of a given identifier.

- **Theme**: A set of icons used for the graphical representation of the control. Themes can be uploaded to the Base module in the **Resources** tab. Creating your own themes is described later in this chapter.
- **Control**: Operating modes of a Light element. There are four options:
  - **None**: Selected item will be inactive on visualization screen. Only its status will be displayed, but it will be impossible to control it.
  - **Dimmer**: This option may be used only with dimmable outputs. Depending on the hardware platform, this element works and looks different. If you use **DOMIQ/Display** panel, press the light element and hold it pressed at least for 1 second. Then release the button. In efect you will see the scale on which you can set the brightness level. In the **Remote** application, the **Light** element operates similarly to LCN buttons. Press the control and hold it to brighten/dim a light. Brief pressing the **Light** element executes on/off commands.
  - **On/Off**: The traditional two–state switch. This method of control can be used both to relays and dimmers.
  - **Code**: Execution of the assigned action will be protected by a PIN code. This function is currently supported only on the **Display** panel. This method of control is designed to disarm/arm alarm zones from the visualization.
- **Parameter**: In the case of lighting control in this field you can specify the value of `ramp`, which is responsible for turning on and off of a light with a certain speed. Parameter range from 0 to 200. The notation is as follows: `ramp:5`.
- **On**: Command that will be executed when you click on a Light element. This command is executed only when the controlled element is currently off.

- `LCN.output.0.113.2=80`
  This commands sets the dimmer output 2 in a LCN module with the address 113 to the value of 80%.

When this field is left blank, then pressing the **Light** element sends the `on` command to the identifier entered in the **Channel** field.

- **Off**: Command that will be executed when you click on a **Light** element. Content of **Off** field is executed only when the controlled element is currently on.

- `LCN.output.0.113.2=0`
  This commands sets the dimmer output 2 in a LCN module with the address 113 to the value of 0%.

When this field is left blank, then pressing the **Light** element sends the `off` command to the identifier entered in the **Channel** field.

NOTICE! When you set the **On** and **Off** then **Param** attribute is ignored. To use ramp parameter together with the **On** and **Off** parameters use the following syntax: `<identifier_name>;<ramp:value>`

- `LCN.output.0.113.2=100;ramp:5`
- `LCN.output.0.113.2=0;ramp:5`

Temperature

This element is used to display the temperature value on visualization screen. Once the **Temperature** is added to a visualization you can set its properties:

When you click on the item, you can set its properties:

- **X** i **Y**: Coordinates of the element. The values of **X** and **Y** can be changed by entering these values from the keyboard or simply moving the element in the visualization window by holding down the left mouse button.
- **Height** i **Width** (read only): The height and width of a visualization element
- **Layer:** The layer on which you placed the element.
- **Channel**: The name of the identifier from where value of temperature is read.

- `LCN.value.0.111.r1`
  Displays the current temperature measured by the LCN R1Var sensor in a LCN module with the address 111.
- `LCN.regulator.0.111.1`
  Displays the set temperature of the LCN regulator 1 in a module with the address 111.

- **Prefix**: Here you can enter text, that will be displayed as a prefix
- **Suffix**: Here you can enter the temperature unit (default °C)
- **Color**: Here you can choose color, that will be used to represent the value of temperature
- **Format**: This option allows to represent the value of temperature in many different formats, described on next page.

Formating examples

Assume that the measured temperature value is 25.43 ° C.	
FORMAT	**RESULT**
.000	25.430°C
.0##	25.43°C
0.##	25.43°C
000.000	025.430°C

Summary: 0 in formatting means that a digit in a concrete position in a number is always displayed. If there is no value, it will be completed with zeros.

The # means that digit in this position will be displayed only if its value is nonzero. The # sign can be placed only after the decimal point.

If the **Format** field is blank, then the temperature will be displayed without formatting.

- **Size**: Font size.
- **Min** i **Max** (only the **DOMIQ/Display** panel): If these fields are blank, then the **Temperature** is used only to display temperature values. But, if the **Min** and **Max** fields are filled, then on the visualization is possible to enter any values by using a virtual keyboard, where **Min** and **Max** determine the range of acceptable values. The value entered will be saved in the state of a identifier entered in the **Channel** field.

On/off

This is a bistable button. It can be used to turn on and off lighting, ventilation, etc. Button label shows the action that will be executed, when button is pressed. For example, if you use **Onoff** to control a single light, then when the light is on, the label of the button displays: **off**. When light is **off** then it shows **on**.

The definition of this element consists of:

- **X** i **Y**: Coordinates of the element. The values of **X** and **Y** can be changed by entering these values from the keyboard or simply moving the element in the visualization window by holding down the left mouse button.

- **Height** i **Width** (read only): The height and width of a visualization element
- **Layer:** The layer on which you placed the element.
- **Channel:** The name of the indentifier that you want to control and whose status will be displayed on the visualization screen.

- `LCN.relay.0.121.1`
  Controls the relay 1 in a LCN module with the address 121 and displays its status.
- `LCN.output.0.113.1`
  Controls the dimmer 1 in a LCN module with the address 113 and displays its status

- **Parameter**: In the case of lighting control in this field you can specify the value of `ramp`, which is responsible for turning on and off of a light with a certain speed. Parameter range from 0 to 200. The notation is as follows: `ramp:5`.
- **On**: Command that will be executed when you click on a Light element. This command is executed only when the controlled element is currently off.

- `LCN.output.0.113.2=80`
  This commands sets the dimmer output 2 in a LCN module with the address 113 to the value of 80%.

When this field is left blank, then pressing the **Light** element sends the `on` command to the identifier entered in the **Channel** field.

- **Off**: Command that will be executed when you click on a **Light** element. Content of **Off** field is executed only when the controlled element is currently on.

- `LCN.output.0.113.2=0`
  This commands sets the dimmer output 2 in a LCN module with the address 113 to the value of 0%.

When this field is left blank, then pressing the **Light** element sends the `off` command to the identifier entered in the **Channel** field.

NOTICE! When you set the **On** and **Off** then **Parameter** attribute is ignored. To use ramp parameter together with the **On** and **Off** parameters use the following syntax: `<identifier_name>;<ramp:value>`

- `LCN.output.0.113.2=100;ramp:5`
- `LCN.output.0.113.2=0;ramp:5`

- **Theme**: A set of icons used for the graphical representation of the control. Themes can be uploaded to the Base module in the **Resources** tab. Creating your own themes is described later in this chapter.
- **Color**: Color of a text on the button label
- **Size**: Size of a text on the button label

Dimmer

Dimmer is one of the most common used visualization element. It allows controlling the proportional output of the **LCN** module, but it can be also used to send **on** and **off** commands to any identifier. Depending on the hardware platform, this element works and looks different. If you use **DOMIQ/Display** panel, press the **dimmer** element and hold it pressed at least for 1 second. Then release the button. In efect you will see the scale on which you can set the brightness level. In the **Remote** application, the **Dimmer** element operates similarly to **LCN** buttons. Press the control and hold it to brighten/dim a light. Brief pressing the **Dimmer** element executes `on`/`off` commands.

Current value of the proportional output is displayed on the **Dimmer** button. **Dimmer** has the same set of features as the **Onoff** button.

Value

Value element is used to display any numeric value on a visualization screen. For example: measurement value from a MOBDUS sensor, relay state, current value of the proportional output etc. Value element has the same set of features as the **Onoff** button and two new options:

- **Offset**: The measured value will be displayed relatively to the number entered in this field.
- **Gain**: Here you can enter any positive number (including floating point numbers) by which, the measured value will be multiplied. In case of floating–point multipliers use a dot as a decimal separator.
- **Min** i **Max**: See the **Temperature** element.

Switch

Switch is an universal visualization element and one of the most often used. It can be used to send any channel command, e.g. setting the value on the LCN regulator, invoking Lua function etc. It is also used to navigate in the visualization structure.

The definition of this element consist of:

- **X** i **Y**: Coordinates of the element. The values of **X** and **Y** can be changed by entering these values from the keyboard or simply moving the element in the visualization window by holding down the left mouse button.
- **Height** i **Width** (read only): The height and width of a visualization element
- **Layer**:The layer on which you placed the element.
- **Label**: Brief description of a switch.
- **Toscreen**: This is one of the navigational functions of the **Switch** button. This feature allows to switch between vizualization screens. This gives a lot of convenience while creating more complex visualizations.

An example of navigation between screens is presented in the pictures below:

Fig 7.5: Screen No. 1

Pressing the **Screen 2** button will navigate to the screen assigned to the **toscreen** parameter of the **Screen 2** button.

Fig 7.6: Screen No. 2

Pressing the **Screen 1** button will navigate to the screen assigned to the **toscreen** parameter of the **Screen 1** button.

- **To layer**: This is another navigational function of the **Switch** button. This feature allows to assing the **Switch** button to a layer. Pressing on the **Switch** button will result in displaying and hiding the layer. It's a very convenient when large number of buttons, light controls etc. are displayed on the visualization screen. Displaying all these elements simultaneously reduces clarity of the visualization. To avoid this, assign the elements of the same type to a specific layer and control its appearance. To assing an element to a layer, select a layer name from the **tolayer** drop-down list.

You want to add a few buttons to control the lightning. The best solution is to create layer named, for example *Light buttons* and assign to it all the buttons. Then add a **Switch** button and from **tolayer** list select *Light buttons* layer. Pressing on the Switch button will result in displaying and hiding the *Light buttons* layer.

Fig 7.7: Visualization before pressing the Lights button

Clicking on the **Light** button displays *Light buttons* layer. Pressing the button again hides the *Light buttons* layer.

Fig 7.8: Screen after pressing the Lights button

- **Command**: Here you can enter any command that will be executed when user hits the **Switch** button.
- **Theme**: A set of icons used for the graphical representation of the control. ater in this chapter.
- **Color**: Color of a text on the button label.
- **Size**: Size of a text on the button label.
- **PIN**: (only on **DOMIQ/Display** panel): 8–digit security code assigned to the button. Command will be executed only after entering the PIN code using the virtual keyboard.

Text

This element is used to display any text on the visualization. The **Text** element can be also used to display state of any identifier.

The definition of the **Text** element consists of:

- **X** i **Y**: Coordinates of the element. The values of **X** and **Y** can be changed by entering these values from the keyboard or simply moving the element in the visualization window by holding down the left mouse button.
- **Height** i **Width** (read only): The height and width of a visualization element
- **Layer:** The layer on which you placed the element.
- **Text**: Any text content that you want to display in the **Text** element.
- **Channel:** The name of the indentifier whose status you want to display in the **Text** element
- **Size**: Font size.

- **Color**: Text color.

Image

This feature allows to add images to the visualization screen. Resolution of the image should be smaller than 800x600.

Do not use this feature to add visualization background – use background attribute on sthe screen!

The definition of this element consists of:

- **X** i **Y**: Coordinates of the element. The values of **X** and **Y** can be changed by entering these values from the keyboard or simply moving the element in the visualization window by holding down the left mouse button.
- **Height** i **Width** (read only): The height and width of a visualization element.
- **Layer:** The layer on which you placed the element.
- **Src**: from this drop–down list you can select an image that you want to add to a visualization screen. You can upload your own images to the **Base** module in the **Resources** tab.

Video

The **Video** element shows the real–time video preview from an IP camera.

Available options are:

- **X** i **Y**: Coordinates of the element. The values of **X** and **Y** can be changed by entering these values from the keyboard or simply moving the element in the visualization window by holding down the left mouse button.
- **Height** i **Width**: Here you define size of the video preview window.
- **Layer:** The layer on which you placed the element.
- **URL**: The URL pointing directly to camera's Motion JPEG video stream. The exact notation depends on the specific IP camera manufacturer and model and usually can be found in the camera's user manual. Username and password are optional. They have to be given if the IP camera is configured to authorize access to the video stream.

```
http://username:password@192.168.10.20/video.mjpeg
```

For detailed information about creating of the visualization refer to the tutorial **Visualizations.** Tutorial is available on our webpage **www.domiq.eu**, **Tutorials** section.

Moreover we can design the graphics for your visualization. For detailed information please write us on **info@domiq.pl**.

7.3. Custom Themes

Creating your own themes allow you to customize your own controls in the visualization. In this section we will present a procedure of creating custom themes. As examples we show how to make your own themes for the dimmer and on/off switch. Graphical aspect is not in the scope of this chapter. We start from the moment when the icons have already been created in any graphics editor.

Dimmer Theme

This particular theme for a dimmer is represented by a set of five icons. Each icon represents a concrete level of dimmer value. In this example, we have five icons for five ranges: 0%, 1–25%, 26–50%, 51–75% and 76–100%.

Some notes about graphic files:

- They must have `.png` extension and be saved as not interlaced.
- Set the transparency of the background in cases where it is the free space around the shape. Otherwise, this area will be filled with white.
- Each file has to be named according to the following scheme: `name_from_to.png`. The `name` used in the file has to be exact the same as the name of the theme. In this example files are named: `light_0_0.png`, `light_1_25.png`, `light_26_50.png` and so on.
- DO NOT use underscore character in the names except as indicated above.

With prepared image files, we can begin creating a theme file. To do that you need any archives creator for example WinRAR, WinZip, 7zip, etc. While creating an archive keep in mind three important options:

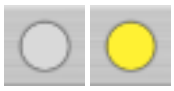
- Select a *ZIP* archive.
- Select the compression method as: *no compression*.
- Check that the files are compressed without any folder.

The archive should be named according to the following scheme: `name.theme`, in this example `light.theme`.

Upload created theme to the **Base** module (max file size is 100KB).

On/Off Theme

Theme to the on/off element is defined similarly to the dimmer theme. In this case you need only two icons: light on and light off.



The files should be named according to the following scheme: `name_0_0.png` for light off and `name_1_1.png` for light on. In this example files are: `switch_0_0.png` and `switch_1_1.png`. and the theme is named `switch.theme`.

Other steps need to be done exactly as for the dimmer theme.

Uploading of created themes to the **Base** module is presented in chapter „Resources" on page 114.

Chapter 8
Events

In the **Events** tab you can define any number of conditional events. Using this feature you can define any logical rules, which makes the **DOMIQ** system very flexible.

For each event you have to specify a condition that must fulfilled to trigger an event and a list of commands which will be executed everytime an event is triggered.

All configuration changes made using the **Event** tab are cached in web browser and do not affect the operation of the **DOMIQ** system until the **Save** button was pressed. The **Revert** button, restores last saved configuration of the **Events** tab.

8.1. Inteface

The **Events** tab is divided into three parts:

1. Tree
2. Details
3. Actions

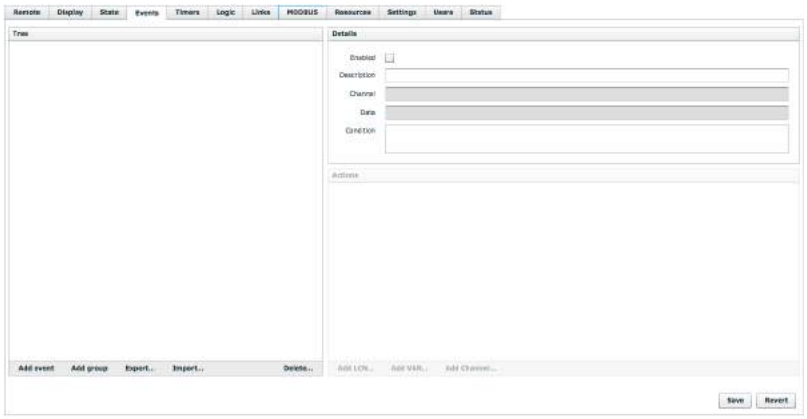


Fig 8.1: Events configuration window

All configuration changes made using the **Events** tab are cached in web browser and do not affect the operation of the **DOMIQ** system until the **Save** button was pressed. The **Revert** button, restores last saved configuration of the **Events** tab.

Tree

In the **Tree** section, the structure of events is displayed. To add a new event click on the **Add event** button. Groups of events facilitates controlling large amounts of events related to a particular area of building automation. A new group is created by pressing the **Add Group** button. Event can be added to the group in two ways: by dragging and dropping it on the name of the group (existing events) or by selecting group, and then pressing the **Add event** button (new events).

Events can be moved within a group by clicking on it and then dragging and dropping in the desired place.

To delete an event, select it and then press the **Delete** button. Deleting a group, deletes all events that belongs to a particular group.

The **Export** button allows to export a single event or a group of events and later import (the **Import** button) in any **Base** module. Using this functionality you can easily create prototypes of events for further use.

In order to export an event or a group of events click on them and then click on **Export**.

In order to import an event or a group of events click on **Import** and then indicate the `.xml` file that you have exported.

Details

In this section you define properties of events and groups. Fields where you typically use the `C.` or `E.` prefixs are highlighted with orange.

Groups have three properties:

- **Enabled**: This check box allows to activate/deactivate a group of events. If it is not selected, then any event that belongs to this group will not be triggered.
- **Label**: A brief description of the group displayed in the **Tree**.
- **Condition**: Condition added to the group creates a a bitwise AND with events within a group. Thus, if the group condition is not fulfilled, then none event that belongs to a particular group will be triggered.

Each event has a set of five attributes:

- **Enabled**: This check box allows to activate/deactivate an event. If it this checkbox is not selected, then an event will not be triggered
- **Label**: A brief description of the event, displayed in the **Tree**.
- **Channel**: The name of an identifier, whose change of state triggers the event.
- **Data**: Expected value of identifier entered in the **Channel** field.
- **Condition**: Additional condition that have to be fulfilled to trigger an event. We will study the examples of creating of conditions in section 8.2. Conditions.

Fig 8.2: The Details section

Actions

In this section you can define one or more commands, that will be executed, each time the event is triggered. Three types of commands are available:

1. Setting value of a variable (the **Add VAR...** button). By checking the **Remember** box, a VAR–type variable is converted to a non–volatile MEM variable.

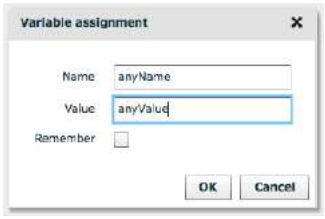


Fig 8.3: VAR/MEM configuration window.

2. Sending any command to any identifier in the **DOMIQ** system (the **Add channel...** button).

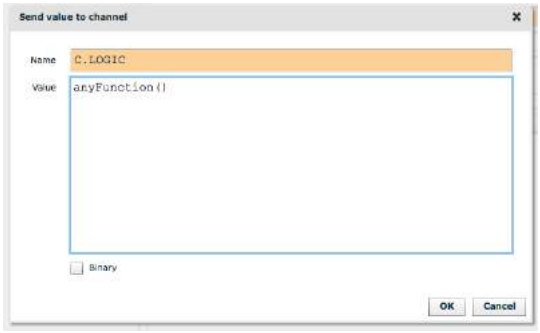


Fig 8.4: An example of invoking Lua function using the Add channel option

In the **Name** field enter the name of an identifier, whereas in the **Value** enter the value that will be assigned to a given identifier.

3. Sending a command to the **LCN** bus, to a selected module or group of modules (the **Add LCN...** button).

On the left side you can enter (the **Module** or **Group** field) or choose from the list (the **Select destination** button) the recipient of a LCN command. The **Test command** button sends a test command in order to test and validate the configuration. On the right, you define the command itself. The following commands are available.

- **Output**

Setting value of a LCN proportional output. Outputs 1, 2 3 and 4 are available. Using sliders you can set value and ramp.

Fig 8.5: The look of the configuration window

- **Relays**

Controlling of selected relays. Relays in range from 1 to 8 are available.

Fig 8.6: The look of the configuration window

- **Motors**

Controlling shutters motors using relays. It is possible to control up to 4 shutters with positioning (**LCN–BS4** required) or without positioning. The following options are available:

- **1**,**2**,**3**,**4** – controlling a single shutter;
- **1+2**, **3+4** – controlling pairs of shutters
- **1–4** – controlling 4 shutters simultaneously (8 relays)

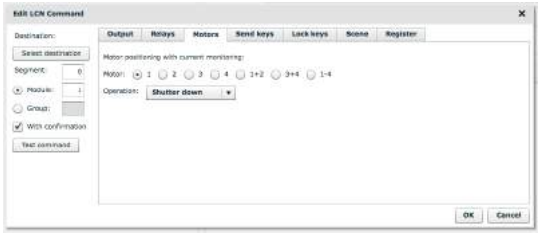


Fig 8.7: Configuration window of the LCN motors command.

- **Send keys**

This command simulates pressing any button(s) from any table(s) with any action(s).

Fig 8.8: Send key command configuration window.

- **Lock keys**

This command allows to lock selected keys from selected table. The **turn on** option means setting the lock. The **turn off** removes the lock. The **toggle** command sets value of the lock to opposite.

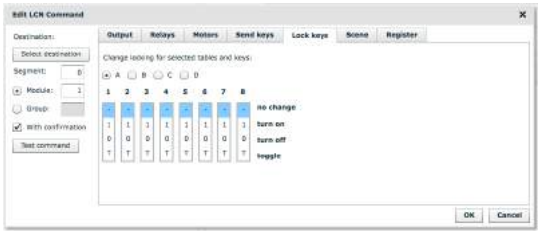


Fig 8.9: The lock keys command configuration window.

- **Scene**

This command is used to load or save a LCN scene.

Fig 8.10: LCN scene configuration

The following parameters are available:

Operation:

- **Load**: Loads selected scene.
- **Save**: Saves selected scene. This command saves the numbers of active outputs and their values at the time of sending the command.

Scene number: The number of scene to be saved or loaded. Available range: **1** to **10**.

Active output: The number of an output to be saved/loaded in a scene. The following options are available: **Output1**, **Output2**, **Output3, Output4**.

Time: If the **Use the time from the scene** box is checked, then outputs will use the ramp saved in the scene definition. Otherwise, you can set a ramp using slider.

- **Register**: This command switches a register of LCN scenes. Available options: **1** to **10**.

Fig 8.11: LCN register configuration

8.2. Conditions

The **Condition** field is used to define additional condition that must be met in order to trigger an event. As a condition you can enter the expected value of any identifier, using the following syntax: `<identifier>==<value>`. If the expected value is a string, for example *"dark"*, then the value should include quotes. Additional conditions may be several, so you can create fairly complex logical rules that determine the triggering of an event. However, note that the commands followed by the condition are executed if and only if the conditional expression is true.

The **Condition** field allows the use of mathematical and logical operators that are present in Lua language:

Operator	Description
<	Less than
>	Greater than
<=	Less than or equal
>=	Greater than or equal
==	Equal
~=	Not equal
and	Logical AND
or	Logical OR
not	Logical NOT

If using multiple operators in one conditional expression, keep in mind the precedence of operators. The precedence of operators in Lua language is as follows (from higher to lower priority):

```
not
< > <= >= ~= ==
and
or
```

However in more complex expressions, we recomend the use of parentheses to avoid precedence mistakes.

Usage examples

- `IDS.armed.1==1`
  Only if the alarm zone 1 is armed.
- `IDS.entry.1==1 and VAR.season=='summer'`
  Only if the alarm zone 1 has set the entry time and the VAR.season variable is equal to 'summer' (own identifier) .
- `MEM.hour>20 or VAR.brightness=='dark'`
  Only if value of the MEM.hour identifier is greater than 20 or the variable VAR.brightness has value set to *dark*.
- `IDS.armed.1~=1`
  Only if the alarm zone 1 is not armed.

Operator precedence and complex expressions.

- `variable3==1 or variable2==0 and variable1==1`
  Logical AND will be examinded first, then the logical OR.
- `(variable3==1 or variable2==0) and variable 1==1`
  Adding parentheses to the preceding example, changes the priority. The expression in parentheses is examined first, then the logical AND.
- `(variable 1==1 and variable 2==2) or (variable 3==3 and variable 4==4)`
  This is an example of sum of logical multiplication. As the first, the expressions in parentheses are examined, next the logical sum.

8.3. Patterns

Patterns allow you to filter events, thus increasing their flexibility and range of usage. Using patterns makes events universal, for example you can define an event triggered by changes of state of any alarm sensor instead of defining multiple events for each sensor. Patterns can be freely combined in a single definition of an event.

Pattern	Description
%d	Any digit in range from 0 to 9
%a	Any letter.
%w	Any digit or letter.
.	Any character. This pattern can by combined with „+" and „*" patterns.
[0–9aZ]	Any digit or letter from a given range. Ranges can be freely modified. For example [1–4aD] means, digit in range 1–4 and letter in range from a to d. This pattern allows to filter occurence of digits and letters at the same time.
*	Zero or more occurences. This parameter is an addition to the first four patterns.
+	One or more occurences. This parameter is an addition to the first four patterns.

- `E.IDS.input.(%d)=1`
  The event is triggered if any alarm sensor in range 1 to 9 changes its state to 1.
- `E.LCN.relay.0.10.(%d)=0`
  The event is triggered if any relay in a LCN module with the address 10 changes its state to 1.
- `E.LCN.output.10.(%d+).(%d)=100`
  This event is triggered if value of any dimmable output in the LCN segment No 10 has been set to 100.
- `E.IDS.(%a+).(%d+)=1`
  The event is triggered if any alarm identifier changes its state to 1.

8.4. Parameters

Parameters allow you to store items captured by patterns and their values. You can use several parameters within the definition of an event. In order to assign values matched by a pattern to a parameter, the parameter must be in parentheses

Parameter	Description
$C0	Stores full name (including identifier).
$C1–9	Stores element captured by a pattern. Numbering of parameters depends on number of used patterns. For example, if two patterns were used, captured items will be stored in parameters $C1 and $C2 and so on.
$D0	Stores value of a given event.
$D1–9	Stores value of an element captured by a pattern. This type of parameters can be used only in combination with patterns that captures values of events. Numbering of parameters depends on number of used patterns. For example, if three patterns were used, captured values will be stored in parameters $D1 and $D2 and $D3.

$C0 parameter always stores the full name of an event, whereas $D0 keeps its current value.

- E.LCN.relay.0.36.(%d)=1
  Let's assume that the relay 3 in a LCN module with the address 36 has just been turned on. **Base** generates an event E.LCN.relay.0.36.3=1. Item captured by the (%d) pattern, in our case 3, is stored in the $C1 parameter. In other words, the $C1 captures the number of a relay, which changed its state.
- E.LCN.output.0.10.(%d)=(%d+).
  Let's assume that the output 2 in a LCN module with the address 10 has been set to 50%. **Base** generates an event E.LCN.output.0.10.2=50. Item captured by the (%d) pattern, in our case 2, is stored in the $C1 parameter. The second item captured by the (%d+) pattern, 50 in this case, is stored in the $D1 parameter. Therefore, the $C1 stores the number of an output, which changed its state, while $D1 keeps value of that output.

8.5. Implementation Examples

In this section we present ready–to–implement examples of using **Events**.

Using SATEL PIR sensors to turn on lights.

- Add an event and set its attributes:
  - Enter a brief description in the **Label** field.
  - In the **Channel** field enter: `E.IDS.input.<PIR sensor number>`
  - In the **Value** enter *1*.
  - Optionally (if needed) you can enter additional condition.
  - In the **Actions** section, click on the **Add LCN** button. In the pop–up window choose the light that will be turned on, after the movement is detected.
  - If you need more actions to be executed, after the movement is detected, please repeat the preceding step.
- Click on the **Save** button to apply changes.

Displaying alarm notifications on mobile devices with the DOMIQ/Remote application.

- Add an event and set its attributes:
  - Enter a brief description in the **Label** field.
  - In the **Channel** field enter: `E.IDS.alarm.(%d+)` – this means *„ in any alarm zone"*.
  - In the **Value** enter *1*.
  - Optionally (if needed) you can enter additional condition.
  - In the **Actions** section, click on the **Add Channel** button. In the pop–up window, in the **Name** enter: *C.REMOTE.notify* and in the **Value** field, enter content of the notification, for example *Alarm in zone $C1*. By using the parameter $C1, you will be informed, in which zone the alarm has been infringed.

Automatic raising of shutters in case of opening a window

- Add an event and set its attributes:
  - Enter a brief description in the **Label** field.
  - In the **Channel** field enter: `E.IDS.input.<input number>`, for example *E.IDS.input.1*.
  - In the **Value** enter *1*.
  - Optionally (if needed) you can enter additional condition.
  - In the **Actions** section, click on the **Add LCN** button. In the window that appears, select the **Motors** tab and then choose the motor and the action to be executed.

For more practical usage examples refer to the tutorialials available on our webpage **www.domiq.eu**, **Tutorials** section.

Chapter 9
Timers

DOMIQ/Base allows to define unlimited number of timers. The **Timers** tab is used for this purpose. The definition of each timer event consists of the calendar date/time when the event should be triggered, an additional condition, and the sequence of commands that are executed each time when the timer event is triggered. Timers and events are powerful tools and they should be crucial in the process of building automation.

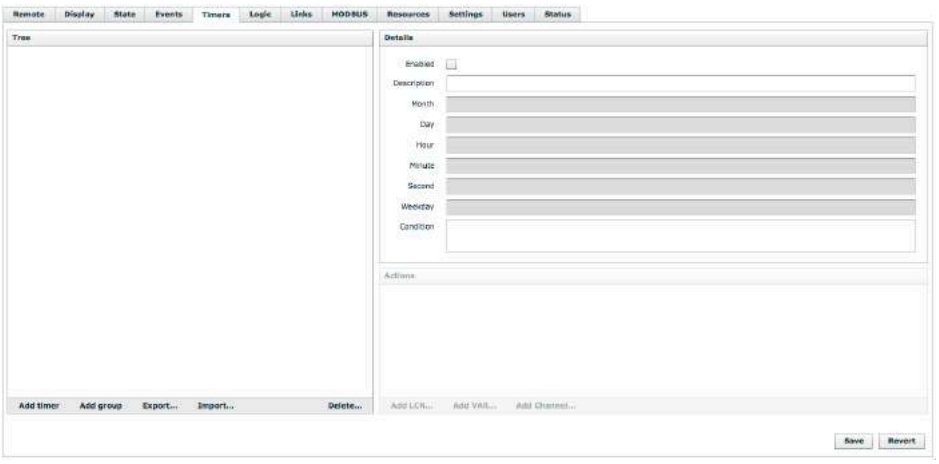


Fig 9.1: Timers definition window

The **Timers** tab is divided into three parts:

1. Tree
2. Details
3. Actions

9.1. Tree

In the **Tree** section, the structure of timers is displayed. To add a new timer just click on the **Add timer** button. Groups of timers facilitates controlling large amounts of timers related to a particular area of building automation. The new group is added by pressing the **Add Group** button. Timer can be added to the group in two ways: by dragging and dropping it on the name of the group (existing timers) or by selecting group, and then pressing the **Add timer** button (new timers).

Timers can be moved within a group by clicking on it and then dragging and dropping on the desired place.

To delete a timer, select it and then press the **Delete** button. Deleting a group, deletes all timers that belongs to a particular group.

The **Export** button allows to export a single timer or a group of timers and later import (the **Import** button) in any **Base** module. Using this functionality you can easily create prototypes of timers.

In order to export an event or a group of events click on them and then click on Export.

In order to import an event or a group of events click on Import and then indicate the .xml file that you have exported.

9.2. Details

In this section you define properties of timers and groups. All properties (excluding the **Label** and the **Condition**) are use to set the time/date when a timer should be triggered.

Groups have three properties:

- **Enabled**: This check box allows to activate/deactivate a group of timers. If it is not selected, then any timer that belongs to this group will not be triggered
- **Label**: A brief description of the group displayed in the **Tree**.
- **Condition**: Condition added to the group creates a a bitwise AND with timers within a group. Thus, if the group condition is not fulfilled, then none timer that belongs to a particular group will be triggered.

Each timer has a set of nine properties:

- **Enabled**: This check box allows to activate/deactivate a timer. If it this checkbox is not selected, then a timer will not be triggered.
- **Label**: A brief description of the timer displayed in the **Tree**.
- **Month**: The calendar month in range from 1 to 12. If you leave this field empty and fill the **Day** property, then the timer will be triggered each month in the selected day.
- **Day**: The calendar day of month in range from 1 to 31. If this field is left empty and the **Hour** is filled, then the timers will be triggered every day at selected hour(s).
- **Hour**: Hour in range from 0 to 23. You can enter multiple values separated by commas.
- **Minute**: Minute in range from 0 to 59. If the **Minute** field is filled, and the **Hour** field is left blank, then the timer will be triggered every hour, at the selected minute.
- **Second**: Second in range from 0 to 59. If the **Second** field is filled, and the **Minute** field is left blank, then the timer will be triggered every minute, at the selected second. You can enter multiple values separated by commas, however, the values must differ by 2
- **Day of week**: Day of week in range from 1 to 7, where 1 is Sunday, 2 is Monday, and so on. Leave this field blank, if you want a timer to be triggered every day.
- **Condition**: Additional condition that have to be fulfilled to trigger a timer.

In each of the above fields (exept the **Label** and **Condition**) you can enter multiple values separated by commas. If fields are related, then the combinations are created, base on the contents of the fields.

- You entered 16 and 20 in the **Hour** field, and 15 and 45 in the **Minute**. In this case, the timer will be triggered at: 16:15, 16:45, 20:15 and 20:45.
- You entered 1,2,3,4,5,6,7 in the **Day** field and 2,3,4,5,6 in the **Day of week**. In this case, the timer will be called on weekdays, in the first week of the month.

Fig 9.2: The Details section

9.3. Actions

In this section you can define one or more commands, that will be executed, each time the timer is triggered. The set of available command is identical as for the events described in the previous chapter.

All configuration changes made using the **Timers** tab are cached in web browser and do not affect the operation of the **DOMIQ** system until the **Save** button was pressed. The **Revert** button, restores last saved configuration of the **Timers** tab.

9.4. Implementation Examples

In this section we present ready–to–implement examples of using **Timers**.

Turning on a single relay, at a specified time, on certain day of week. In our example: Monday at 1.30 pm.

- Add a new timer.
- Set its attributes in the **Details** section:
  - Enter short description.
  - In the **Hour** field enter: *13.*
  - In the **Minute** enter: *30.*
  - In the **Day of week** field type: *2.*
- In the **Actions** section, add command to be executed, when the timer is triggered.

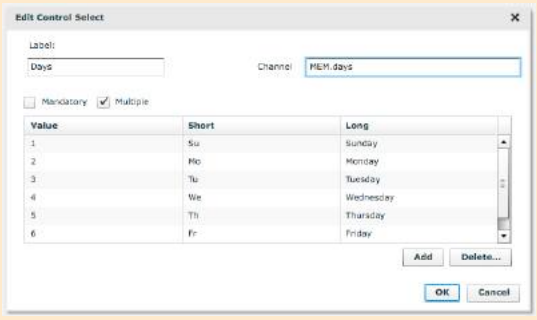
Turning on lights 30 minutes before sunset on weekdays.

- Add a new timer.
- Set its attributes in the **Details** section:
  - Enter a short description.
  - In the **Hour** field enter: *sunset*. **DOMIQ/Base** module has build–in astronomical clock. In order to trigger a timer at sunset, enter *sunset* in the **Hour** field. If you entered *sunrise*, timer will be triggered at sunrise.
  - In the **Minute** field enter *–30*. Entering value with the "–" is interpreted as "in advance".
  - Enter *2,3,4,5,6* in the **Day of week** field – timer will be triggered Monday to Friday.
- In the **Actions** section, define commands executed, when the timer is triggered.

Setting a timer using DOMIQ/Remote.

To achieve this functionality we need to use the **Timers** and the **Remote** tabs. The procedure is as follows:

- In the **Timers** tab, add a timer and set its attributes:
  - In the **Label** field enter a short description.
  - In the **Hour** field enter: *MEM.hour*.
  - In the **Minute** enter: *MEM.minute*.
  - In the **Day of week** enter: *MEM.days*.
    In this way we declared non–volatile MEM variables. We will use them to store values set in the **Remote** application. Names of the MEM variables can be any, without spaces.
- In the **Actions** section, define commands to be executed each time, when the timer will be triggered.
- Click on the **Save** button to apply changes.
- Select the **Remote** tab.
- Add a **Time** element. Double–click on it and set its features:
  - In the **Label** field enter a short description.
  - In the **Hour** field enter: *MEM.hour*.
  - In the **Minute** enter: *MEM.minute*.
- Add a **Select** element and fill its properties:
  - In the **Label** field type a short description, for example : *Days*.
  - In the **Channel** enter: *MEM.days*.
  - **Multiple**: When this box is checked, user can select multiple elements from a given list.
  - **Mandatory:** When the **Mandatory** box is checked, user has to choose at least one option from a given list.
  - Fill the the table as presented below:



  In this case, the **Select** element allows user to select day(s) of week, but the table may be filled with any values, from which user can select desired options, such as ventilation speed or days of month, etc.
- Click on the **Save** button to apply changes.

Fig 9.3: The result of the last example in the DOMIQ/Remote.

Chapter 10
Links

The **Links** tab has two uses. The first one allows you to define point–to–point connections between two **Base** modules, in order to create structural building automation network. The second function lets you to configure intersegmental communication using **Base** modules as **LCN** segment couplers (**LCN–SK**).

The overview of the **Links** tab is presented in the picture below:

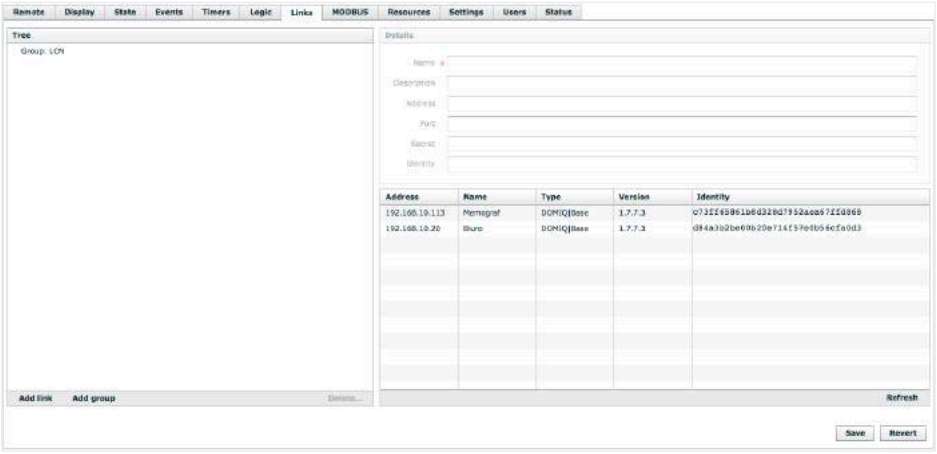


Fig 10.1: Links edition window

A framework of the **Links** tab is very similar to the **Events** and **Timers** tabs. The left side of the window is the **Structure**. It contains a list of defined groups and connections. Groups are used to connect and communicate bigger groups of **Base** modules, including intersegmental communication. Single connections are used for point–point connections.

In the right side of the **Links** tab, there is the **Details** section, in which connection parameters are configured.

Fig 10.2: Link parameters configuration window

Name: The name of the **Base** module with which you want to establish connection. The name can be any, without spaces. The name entered in this field has no relation with the name used in the **Settings** tab.

Description: A brief description of a given connection.

Address: IP address of the **Base** module with which you want to establish connection. In case of group connections, enter multicast IP address assigned to the group (see section 10.2 „Base as Segment Coupler" on page 94).

Port: This field should be filled if the connection to the **Base** module has been redirected to a port other than the default (44544).

Password: If you fill the **Password** field, the exchanged data will be encrypted.

Identity: Each **Base** module has its own unique ID. ID serves as collateral security for data transmission between Base modules. It prevents the use of the IP address assigned to the **Base** module with given ID by other network devices.

Below the **Details** section, there is a list of all **Base** modules available in the installation with their IDs.

The starred fields are obligatory.

10.1. DOMIQ structural network

As mentioned in the introduction of this chapter, you can create a point-to-point connections between **Base** modules. A single connection is called a link. Using links, **Bases** can exchange data and respond to each other's events.

This opens the way for the implementation of so far unavailable functionality. **Base** modules can connect with both local network and the Internet. **In the case of Internet connections it is necessary to open UDP port 4554**. Links have been designed for installations where will be installed at least two or more **Base** modules, and then based on them will be created an integrated automation network.

An excellent field of application of links are, for example blocks of flats or hotels. In such installations, usually a hierarchical structure is created, where one **Base** module acts as a master device and processes data received from slave modules (for example those installedi in flats).

Configuration of a single link between two Base modules.

Configuration of a single link is made crosswise – the **Name** and the **Address** of the first module should be entered in the configuration of the second module and vice versa. For this example, we assume that the names of modules are: `domiq1` and `domiq2`, and IP addresses are: `192.168.1.100` and `192.168.1.101`. The configuration procedure is as follows:

Module 1 (domiq1)

1. Select the **Links** tab.
2. Add a new connection and set its features:
   - In the **Name** field enter the name of the **Base** module with which you want to establish connection, in our case: `domiq2`.
   - In the **Address** field enter the IP address of the **Base** module with which you want to establish connection. In our case: `192.168.1.101`.
   - Fill the **Password** is you want data to be encrypted.
   - Optionally fill the **Identity**, **Description** and **Port** fields.

Module 2 (domiq2)

Repeat steps 1 and 2, remember to use the name and the IP address of the first module, in our case: `domiq1`, `192.168.1.100`.

As we mentioned earlier, the links allow to send commands and react on events from other **Base** modules. For this purpose, network identifiers are used. For complete description of the syntax of network identifiers and use examples see section 16.10 „Links" on page 164.

10.2. Base as Segment Coupler

The **Links** tab allows to define intersegmental communication, using the **DOMIQ/Base** as a segment coupler.

In order to provide intersegmental communication, **Base** module must have defined a group of links (the **Add group** button) and be configured as follows:

- Enter *LCN* in the **Name** field. **Any other name will prevent communication between LCN segments**.
- In the intersegmental communiacation **Base** modules use multicast IP to exchange information. In the multicast IP, according to RFC 3171, the IP addresses are assigned in range from **224.0.0.0** to **239.255.255.255**. By default **Base** uses multicast IP `239.255.255.44` and port: `44544`. You can leave the **Address** field blank if you want to use default IP and port. Otherwise enter IP from range mentioned above. Remember that each **Base** within particular group must have the same multicast IP address and password (if given).
- If you fill the **Password** field, the exchanged data will be encrypted.

Below **Details** section, there is a list of all **Bases** present in the installation.

Each **DOMIQ/Base** module with software 1.8.0.0 or newer includes a default configuration, which enables LCN intersegmental communication.

In order to assign a **Base** module to a particular **LCN** segment, follow these steps:

1. Choose the Settings tab, in the **Segment** field enter the number of the **LCN** segment.
2. Check **Events for other segments** to allow the **Base** module to react on events from other segments. It also enables **Base** module to keep state of LCN modules from other segments.

Fig 10.3: Assigning Base to a LCN segment.

Commands and events used in other segments have typical syntax. The only different is the number of the segment.

- `E.LCN.output.10.11.1=100`
  The output No 1 in a module with the address 11, in the segment No 10. has been turned on.
- `E.LCN.relay.10.11.5=0`
  The relay No 5 in a module with the address 11, in the segment No 10. has been turned off.
- `C.LCN.output.10.11.1=100`
  Turn on the output No 1 in a module with the address 11, in the segment No 10.

For more detailed information regarding using **DOMIQ/Base** as a LCN segment coupler refer to the tutorial „**Base as segment coupler**" on our website **www.domiq.eu**, **Tutorials** section.

10.3. Network Variables

In the previous section we presented the use of group connections, which allow the use of **Base** modules as segment coupler in the LCN installation. Another application of group connections is creating groups of **Base** modules that share variables, called network variables.

Network variables can be used in more complex building automation installations based on **Base** modules. In such installation one of the **Base** modules can (this is not required) play primary function and provide information to other modules (via network variables) e.g. about emergency situations (fire, flooding, etc.). Changing variables is bidirectional between all modules connected to the group. All modules within a group have access to the status of the network variable and can change the said status (using the commands) as well as respond to state changes (using events).

Network variable values are at the same time stored in all **Base** modules within the group. Restart of the **Base** module deletes variables from the memory. However, after restarting the **Base** module automatically synchronizes the variables by taking their values from another module belonging to the group. Network variable values will be completely erased only in the event of simultaneous reboot of all **Base** modules belonging to the group.

When you add a new group in **Links**, you should configure its parameters:

- In the **Name** field, type the name of the group. Enter the same names in all **Base** modules, which are to belong to the group.
- The remaining fields are optional.

NET identifiers are dedicated to support network variables. A detailed description can be found in section: 16.11 NET.

Chapter 11
MODBUS

DOMIQ/Base in combination with **DOMIQ/Serial–4MB** allows to integrate the **DOMIQ** system with devices using MODBUS protocol. Thanks to commonness of MODBUS devices, the integration can greatly extend the functionality of an intelligent system. It becomes possible to use devices such as PLCs, digital energy meters, sensors (temperature, humidity, VOC), recuperators drivers, weather stations and more.

All configuration changes made using the **Modbus** tab are cached in web browser and do not affect the operation of the **DOMIQ** system until the **Save** button was pressed. The **Revert** button, restores last saved configuration of the **Modbus** tab.

Integrating with MODBUS device comes down to defining an interface and then read/write device registers.

11.1. Interface

The first step of integration process is creation of an interface (the **Add interface** button) and next to fill its attributes. Three interface types are available:

1. **Serial**
2. **TCP**
3. **UDP**

The configuration window is presented below:

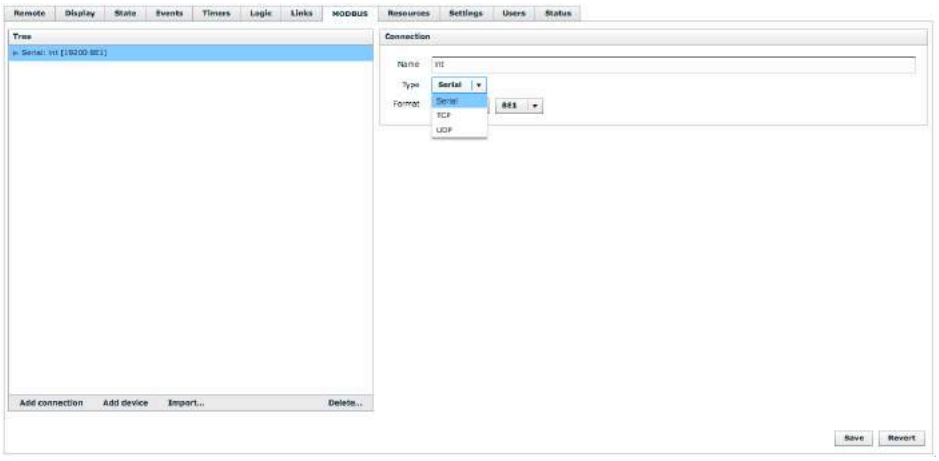


Fig 11.1: MODBUS interface configuration window

The configuration window is divided into two part. The left part shows the structure of defined interfaces, with added devices. In the **Connection** section you define type of an interface and its parameters.

Serial

Serial, in other words MODBUS RTU – serial communication based on RS–485 communication standard. The definition on this type consists of:

- **Name**: Enter the name (without spaces) that uniquely identifies the interface. The interface name is used in identifiers (see section 16.5 „MODBUS" on page 154).
- **Format**: The definition of a MODBUS packet consists of two parameters: transmission speed and frame format. The available transmission speeds are: **9600**, **19200**, **38400**, **57600**bit/s, and frame formats: **8N1**, **8N2**, **8E1**, **8O1**.

TCP and UDP

MODBUS/TCP is modification of standard MODBUS RTU protocol, in which transport layer based on RS–485 is replaced by TCP/IP protocol. MODBUS/TCP uses Ethernet as the physical layer of connections.

The definition of this type of interface consists of:

- **Name**: Enter the name (without spaces) that uniquely identifies the interface. The interface name is used in identifiers (see section 16.5 „MODBUS" on page 154).
- **Adres IP**: IP address of the slave device, with which you want to establish connection.
- **Port**: The port number to which MODBUS/TCP packets are sent. The default port is **502**, which is officially reserved for this purpose.

Some devices also use a custom variation, in which TCP protocol is replaced by connectionless UDP protocol. **UDP**–type interface has the same set of parameters as **TCP**.

11.2. Device

The second step of integration is to add a device, configure its parameters and then add registers readouts. To add a new device to an interface, click on the interface name and then click on the **Add Device** button. The new device will be listed in the **Structure** on the left of the configuration window.

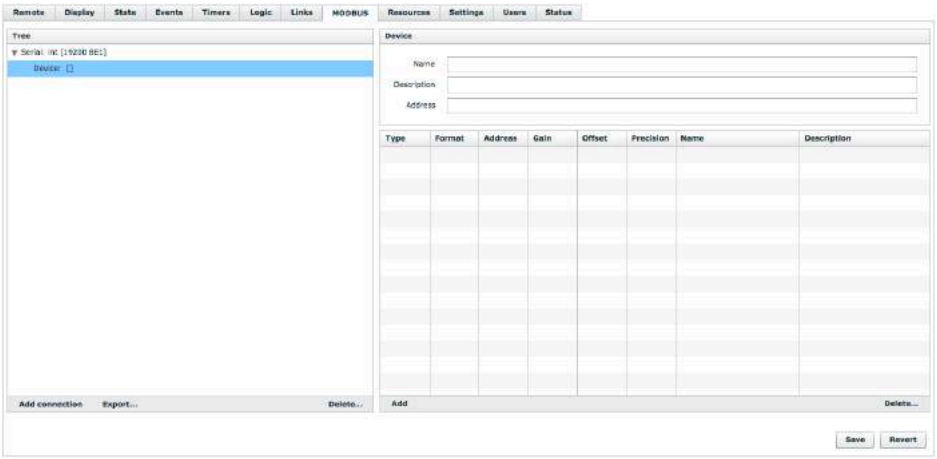


Fig 11.2: Device configuration window.

Configuration window is split into three sections. The left part shows the structure of present interfaces with added devices. In the **Device** section you define device parameters:

- **Name**: Enter the name (without spaces) that uniquely identifies the device. The device name is used in identifiers (see section 16.5 „MODBUS" on page 154).
- **Description**: A brief description of a device.
- **Address**: Address of a MODBUS slave device.

The third part of the configuration window is used to define registers readouts. To add a new register readout, click on the **Add** button. To delete readout, select the readout from the list and then click **Delete**. The section is divided into eight columns:

1. **Type**
2. **Format**
3. **Address**
4. **Gain**
5. **Offset**
6. **Precision**
7. **Name**
8. **Description**

Type

In the **Type** column you choose the type of read value and MODBUS function used for this purpose. Four option are available:

- **coil** – value of a single binary output (function 1)
- **input** – value of a single binary input (function 2)
- **inreg** – input register readout (function 4)
- **outreg** – holding register readout (function 3)

Format

Here you can select the format of a read register. Information about the register format should be provided in the device's manual.

The following options are available:

- **uint16** – 16–bit register of unsigned integers;
- **int16** – 16–bit integer register;
- **uint32** ––32–bit register of unsigned integers;
- **int32** – 32–bit integer register;
- **uint32ws** – 32–bit register of unsigned integers with swapped words;
- **int32ws** – 32–bit integer register with swapped words;
- **float** – 32–bit floating point register;

Address

In the **Address** column enter the numbers of registers you want to read (write). For 32–bit registers, enter only the address of the first register and the configurator will automatically fill the number of the adjacent register.

Gain

Here you can enter any positive number (including floating point numbers) by which, value of read register will be multiplied. In case of floating–point multipliers use a dot as a separator.

Offset

Here you can enter any number (including floating point numbers), relative to which, value of register will be displayed.

Precision

This field specifies the precision of the displaying measured value. For example if you enter 2, then the value will be displayed with two decimal digits and so on.

Name

Enter the name (without spaces) that uniquely identifies the register. The register name is part of MODBUS identifiers (see section 16.5 „MODBUS" on page 154).

Chapter 12
BACnet

From software version 1.8.0.0 the **Base** module allows integration with the BACnet system. Integrative functions have been developed in accordance with the guidelines of the ASHRAE–135–2012 norm and annex J "BACnet/IP" to the said norm. From the point of view of the **BACnet** installation, the **Base** module serves as a **B–ASC** (BACnet Application-Specific Controller) - a special driver. The special drivers are created to execute the in advance specified functionality, which is programmed by the manufacturer of the device. In these devices, the user-side configuration is limited only to change the parameters. In the case of the **Base** module, the functionality was implemented in two areas provided for in the norm:

1. **Data Sharing** - Exchange of data with other **BACnet** devices present in the installation.
   - **Data Sharing-ReadProperty-B (DS-RP-B)** - Read the value (e.g. temperature, status of the relay).
   - **Data Sharing-ReadPropertyMultiple-B (DS-RPM-B)** - Read multiple values at the same time.
   - **Data Sharing-WriteProperty-B (DS-WP-B)** - Save the value.
   - **Data Sharing-COV-B (DS-COV-B)** - Report the change of the selected value for request of another **BACnet** device.
2. **Device and Network Management**
   - **Device Management-Dynamic Device Binding-B (DM-DDB-B)** - Dynamically assign and manage devices.

In terms of the integration of the subsystems of an intelligent building, **Base** acts as a gateway that joins BACnet system with any subsystem (LCN, Satel, Modbus, etc.). The integration with **BACnet** system is available when you purchase additional licence (adding a licence to the **Base** module is presented in section 4.4. BACnet Configuration).

The available integration functions enable:

- **Saving the analog and binary inputs.** In this case any device in the installation can be a source of values for the input in the **BACnet** system, e.g. the temperature value measured by the LCN sensor, reed relay, relay, etc.
- **Bi-directional record of the analog and binary outputs.** This makes it possible to pass values between the **BACnet** device and any other device in the installation of an intelligent building bidirectionally. As an example, let's take any analog output in the **BACnet** system and the dimmable output in the **LCN** module. In the event of change in the value of output in the **BACnet** system, LCN will also change its value. It also works other way round, thus changing the value of the output in the **LCN** module will result in change in the **BACnet** system.

Configuration tab is divided into 3 main parts:

- **BACnet variable overview**,
- **Device configuration**,
- **Variable configuration.**

It is necessary to fill the starred fields.

All changes made in the **BACnet** are stored locally in your browser and do not affect the operation of the **DOMIQ** system. Just pressing the **Save** button saves the changes to the **DOMIQ**/**Base** module. The **Revert** button restores the last saved configuration.

12.1. BACnet variable overview

In the window there are all defined variables along with their key parameters. The window is divided into 4 columns:

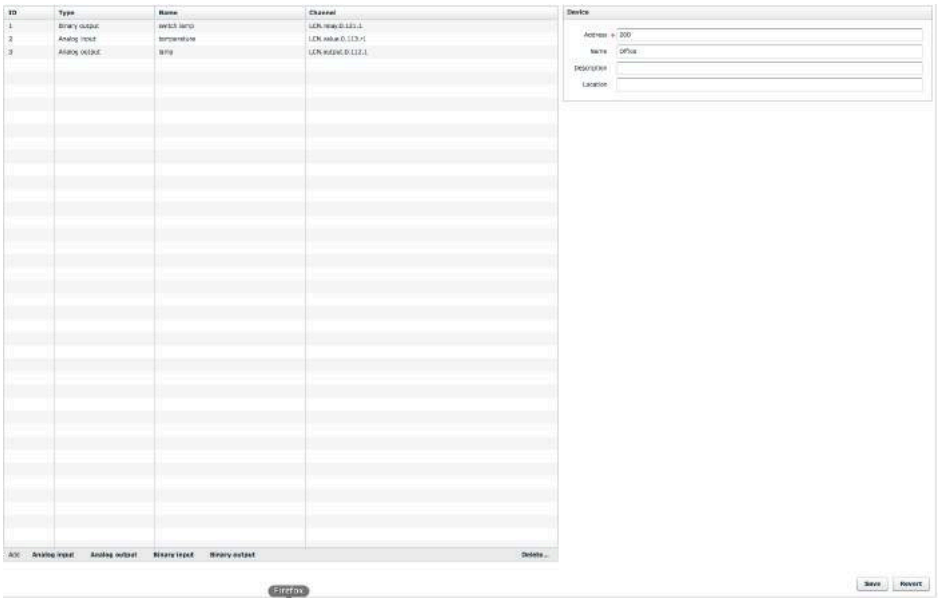


Fig 12.1: BACnet tab - general view

- **ID** – Unique identifier of the variable in the **BACnet** system.
- **Type** – Variable type.
- **Name** – Variable name. The name will be visible in the **BACnet** system configuration tools as well.
- **Channel** – ID in the **DOMIQ** system from which the value is read/to which the value is saved. In the case of inputs, the **Channel** is the source of the values that are passed to a variable in the **BACnet** system. In the case of outputs, values are transfered bidirectionally as described in the introduction of this chapter.

The menu with the buttons below the window overview allows you to add or remove variables:

- **Analog input** – The button that adds a new variable of the analog input type.
- **Analog output** – The button that adds a new variable of the analog output type.
- **Binary input** – The button that adds a new variable of the binary input type.
- **Binary output** – The button that adds a new variable of the binary output type.
- **Delete** – The button that removes the selected variable.

Each variable can be available in the whole the **BACnet** installation as a combination of a variable identifier and a device address.

12.2. Device Configuration

In this window you should enter the configuration parameters of the **Base** module to work in the **BACnet** system.

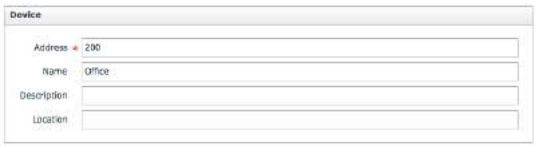


Fig 12.2: BACnet device configuration window

The following options are available:

- **Address** – Unique device address in the **BACnet** system that will be assigned to the **Base** module. In BACnet system can work up to 4194304 devices, which means that each one may have an address given between 0 and 4194303. The addressing is usually made as follows XXFFBBB, where:
  - XX stands for the device number from 0 to 40.
  - FF is the storey number from 1 to 35. The value 00 is reserved for the building backbone network.
  - BBB stands for the building number from 0 to 654.

  For example, a device with the address 234567 means the device No. 12 on the floor 34 in the building No. 567. The numbering system has a great advantage-it allows you to easily and quickly determine the physical location of the device. This kind of numbering allows 41 devices on each floor, 35 floors and up to 655 buildings. In cases requiring a different numbering, you can apply a different devices addressing type.
- **Name** – The device name that will be visible in the **BACnet** system configuration tools. Default value: *DOMIQ Base*.
- **Description** – A short description of the device that will be visible in the **BACnet** system configuration tools.
- **Location** – Information about the location of the device. Useful for extensive **BACnet** installations.

12.3. Variable configuration

Variable configuration window is used to determine the parameters of the **BACnet** variable.

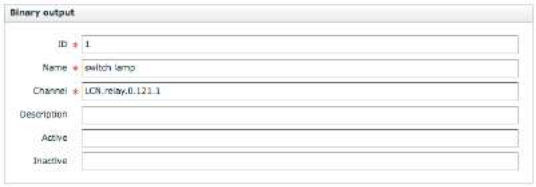


Fig 12.3: Binary output configuration window

Depending on the type of a variable type, the set of available properties will be slightly different.
For binary inputs and outputs the following set is available:

- **ID** – Unique variable number in the **BACnet** system.
- **Name** – Unique variable name in the **BACnet** system.
- **Channel** – ID in the **DOMIQ** system from which the value is read/in which the value is saved. In the case of inputs, the **Channel** is the source of the values that are passed to a variable in the **BACnet** system. In the case of outputs, values are transfered bidirectionally as described in the introduction of this chapter.
- **Description** – A short description of the variable that will be visible in the **BACnet** system configuration tools.
- **Active** –
- **Inactive** – .

For analog inputs and outputs the following set is available:

- **ID** – Unique variable number in the **BACnet** system.
- **Name** – Unique variable name in the **BACnet** system.
- **Channel** – ID in the **DOMIQ** system from which the value is read/in which the value is saved. In the case of inputs, the **Channel** is the source of the values that are passed to a variable in the **BACnet** system. In the case of outputs, values are transfered bidirectionally as described in the introduction of this chapter.
- **Description** – A short description of the variable that will be visible in the **BACnet** system configuration tools.
- **Unit** – Variable unit in the **BACnet** system.
- **Gain**: Here you can enter any positive number (including floating point numbers) by which the measured value will be multiplied. In case of floating–point multipliers use a dot as a decimal separator.
- **Offset** – Here you can enter any number (including floating point numbers and negative numbers which will be added to the variable).

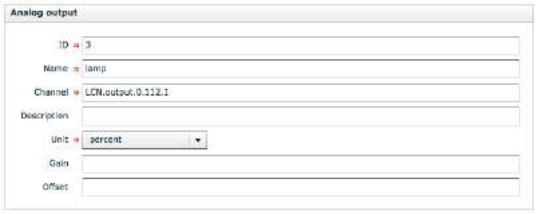


Fig 12.4: Analog output configuration window

Chapter 13
DALI

DOMIQ/Base in combination with **DOMIQ/Light** lets you control lighting installations based on the DALI protocol. In accordance with the DALI standard you can connect up to 64 DALI devices to the **DOMIQ/Light** driver. The module provides a full bidirectional communication, e.g. brightness status update, information about lamp failures.

DALI tab is divided into five main parts:

- **Installation overview** table.
- **Whole installation** panel.
- **Single ballast** panel.
- **Scenes** table.
- **Groups** table.

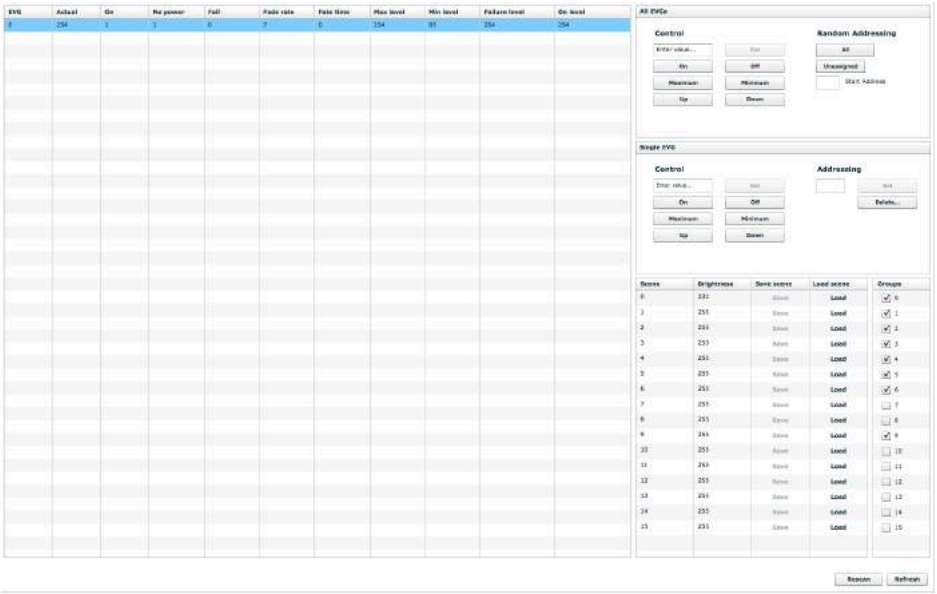


Fig 13.1: General view of the DALI tab

13.1. Installation preview

The installation preview window contains a table with the list of available DALI ballasts along with their individual parameters. The window is divided into 11 columns:

1. **EVG** – Ballast address (from 0 do 63).
2. **Brightness**– Actual brightness value (from 0 to 254).
3. **On** – Status flag indicating ballast activation. 0 – off, 1 – on.
4. **No power supply** – Status flag indicating power supply failure. 0 – OK, 1 – failure.
5. **Failure** – Status flag indicating ballast failure. 0 – OK, 1 – failure.
6. **Step**, **editable field** – Parameter specifying the brightness step change when the commands up and down are triggered. The limit value from 1 to 15. The following table shows step values and attributed to them number of the triggered commands (e.g. pressing the button) which is necessary to change the brightness fully (from 255 to 0). Depending on the manufacturer and model of the ballast this number may vary due to the physical value of the minimum brightness.

Step value	Number of triggered commands
1	4
2	6
3	8
4	10
5	15
6	20
7	29
8	43
9	51
10	85
11	102
12	127
13	170
14	254
15	508

7. **Ramp**, **editable field** – Time of the brightness changes from the current value to the set one. The ramp is independent on the level of changes, so in case of controlling many lamps with the same set ramp, changing ends at the same time. The following table shows the range of acceptable values and times attributed to them.

Ramp value	Setting time [s]
0	0,5
1	0,707
2	1
3	1,414
4	2
5	2,828
6	4
7	5,657
8	8
9	11,314
10	16
11	22,627
12	32
13	45,255
14	64
15	90,510

8. **Maximum level**, **editable field** – The maximum brightness level. If the new maximum level is lower than the minimum one, then the minimum level will be set as the new maximum. The limit value from 0 to 254. Default value: 254.
9. **Minimum level**, **editable field** – The minimum brightness level. Minimum brightness level cannot be lower than the minimum physical brightness level (depending on the model and manufacturer of the ballast). If you try to set the value of below the minimum physical level, the value will be automatically matched to the minimum physical level. If the minimum value was set above the maximum level, then the maximum level will be set as the new minimum. The limit value from 0 to 254. By default the minimum level is equal to the minimum physical level of the given ballast.
10. **Level in case of error**, **editable field** – The brightness level to which the lamp will be controlled in case of failure. The limit value from 0 to 255. In case of setting the value of 255, then the brightness will not change in case of failure. Default value: 254.
11. **Level after switching on the power supply**, **editable field** – The brightness level to which the lamp will be controlled after switching on the power supply. The limit value from 0 to 255. In case of setting the value of 255, depending on the manufacturer of the ballast, the last set brightness value or the value of the last sent command changing brightness will be restored. Default value: 254.

Changes in the editable fields will be approved when you move the focus from the edited fields in any other place within the window **DALI**, e.g. start editing another field etc.

13.2. The entire installation

The panel allows you to control the entire installation at the same time. The buttons are grouped in two areas: **Control**, **Random Addressing**.

Fig 13.2: Panel sterowania instalacją DALI

Control

The **Control** section contains buttons that control the entire lighting installation DALI. Commands are sent to all ballasts connected to the DALI bus (also those without assigned addresses). The following options are available:

- **On** – Turns all the lamps on. The lamps will be controlled to the value of 254 (maximum).
- **Off** – Turns all the lamps off.
- **Max** – Sets maximum brightness value.
- **Min** – Sets minimum brightness value.
- **Brighter**-Step brightening. Changes dependend on the setting of the parameter **Brightness step** of the individual ballasts. A command does not activate the inactive ballast.
- **Darker** – Step dimming. Changes dependend on the setting of the parameter **Brightness step** of the individual ballasts. Sending a command when the maximum brightness is set does not cause turning the ballast off.
- **Edition field and the Set button** – It allows setting any brightness value in the installation (range from 0 to 254).

Random addressing

The buttons contained in this section allow you to address the ballast in the entire installation automatically. The **All** button starts random addressing of the entire installation. After it is run all the ballasts in the installation (with and without addresses) are given new, random addresses. Starting this procedure in the already existing installation will result in removing the former addressing. The **Initial Address** field allows to determine the initial address, towards which the addressing will be initiated. This is especially important when you want to address only a certain part of the installation.

The **Unassigned** button starts the procedure of automatic addressing only for those ballasts that do not have the assigned addresses. **The successful address assignment will be indicated by turning on the the maximum brightness of the given lamps.**

13.3. Single EVG

The **Single EVG** panel controls the single DALI dimmer. The panel becomes visible after you select the specific ballast in the **Installation preview** window.

The window contains two sections:

- **Control** – Contains the same set of buttons as in the case of the entire installation control. Commands are sent to the given ballast.
- **Addressing** – Allows you to change the current address of the lamp or remove it from the installation.

Fig 13.3: Panel to manage a single ballast

13.4. Scenes

In accordance with the DALI standard each ballast may have 16 scenes saved. The table becomes visible after you select the certain ballast in the **Installation preview** window. It is divided into four columns:

The table contains four sections:

- **Scene**– Contains numbers of the certain scenes (from 0 to 15).
- **Brightness**, **editable column** – In this column you can enter the new brightness you want to save to the scene. The column also displays the current values of individual scenes.
- **Save scene** – Contains **Save** buttons which trigger the command **Save scene**. The button becomes active after the correct value is entered in the **Brightness** column. In order to save the scene press the **Save** button after you change the value in the **Brightness** column.
- **Load scene** – The set of buttons to trigger a certain scene.

Scene	Brightness	Save scene	Load scene
0	232	Save	Load
1	255	Save	Load
2	255	Save	Load
3	255	Save	Load
4	255	Save	Load
5	255	Save	Load
6	255	Save	Load
7	255	Save	Load
8	255	Save	Load
9	255	Save	Load
10	255	Save	Load
11	255	Save	Load
12	255	Save	Load
13	255	Save	Load
14	255	Save	Load
15	255	Save	Load

Fig 13.4: Table to manage DALI light scenes

13.5. Groups

According to the DALI norm each ballast can be assigned to max. 16 groups. The **Groups** table allows you to manage groups. Selecting the group number assigns the ballast to the group automatically. When you check the field off, the ballast is removed from the group. In the section DALI.1.group we presented how to control the DALI groups. The **Groups** table becomes visible after you select the certain ballast in the **Installation preview** window.

Fig 13.5: Groups management table

13.6. Control with LCN

The **Base** module enables to pass the DALI commands directly from the LCN installation to the **DOMIQ/Light** module and thus control the connected to it DALI installation. Therefore it enables the control directly from the LCN wall buttons, as a reaction to violation of the PIR sensor, etc. Configuration is limited only to entering the correct parameters of the LCN module in the LCN-Pro software. As an example, we will show you how to turn on a lamp using the wall button which is assigned to the A1 key. The action "**short**" will turn on the ballast with the address 0.

- Select the LCN module the buttons are connected to.
- Select the **Base** module as a recipient of the **A1 short** action (default address 254).
- Select **DALI** as the command type.
- Select the ballast address. **NOTICE!** In the LCN system the numbering begins with 1, while in **DOMIQ** it is counted from 0. Therefore in order to control the lamps with the address 0, select 1 in the LCN-Pro.
- The last step is to select the DALI command, that will be sent to the **DOMIQ/Light** module. In this case choose the **Set** and enter a value of 100%.

Chapter 14
Resources

The **Resources** tab allows you to add your own files to **Base** module and lets you manage all previously added files.

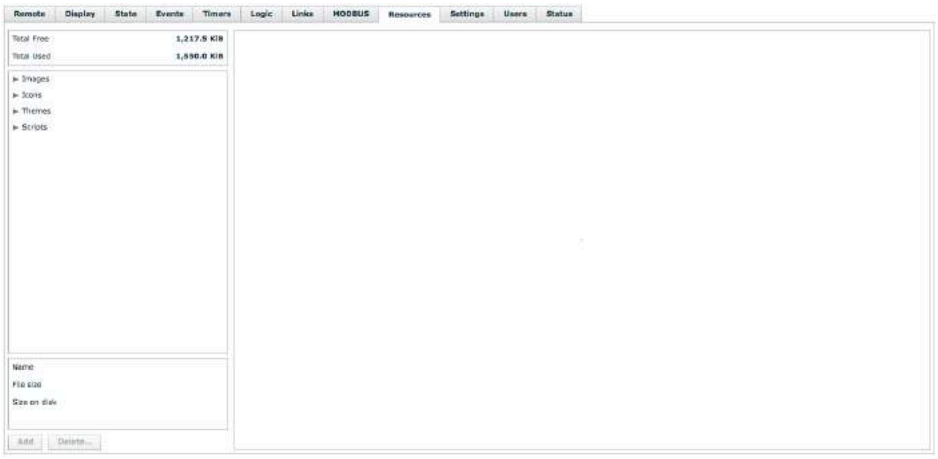


Fig 14.1: The Resources tab manager window

The window is divided into three parts:

1. Memory Statistics
2. File Structure
3. Preview Window

14.1. Memory Statistics

On the top left corner, there is a field with information about free and used space in the memory of the **DOMIQ/Base** module. In the case of images and icons are also displayed information about the file resolution. To preview information about a file, click on its name.

14.2. File Structure

This section shows the tree–view structure of files. Files are grouped into four folders:

1. Images
2. Icons
3. Themes
4. Scripts

Images

Here you can add and browse graphics and pictures, which can be used in the visualization.

To add a new file:

1. Click on the **Images**.
2. Then click the **Add** button.
3. In the window that appears, select the file you want to add.

The maximum file size supported by the **DOMIQ/Base** module is 120kB and maximum resolution is 800x600.To delete an image, select it and click the **Delete** button.

Icons

The icons are shown on the navigation bar in the **DOMIQ/Remote** application.

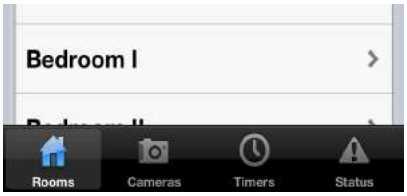


Fig 14.2: Pages icons in the DOMIQ/Remote

The icons are added and deleted identically as images.

Themes

Themes are used to represent active elements of visualizations, such as switches, buttons, lights, etc. Themes are added and deleted identically as images. Creating your own themes is described in the section 7.3 „Custom Themes" on page 69.

Scripts

Scripts are Lua programs, which execute a predetermined functionality. Scripts can be invoked in the **Logic** tab using the command: `import 'script name'`.

Adding and deleting scripts is the same as in case of images.

14.3. Preview Window

The preview window is on the right side of the screen. In this section you can preview all uploaded files.

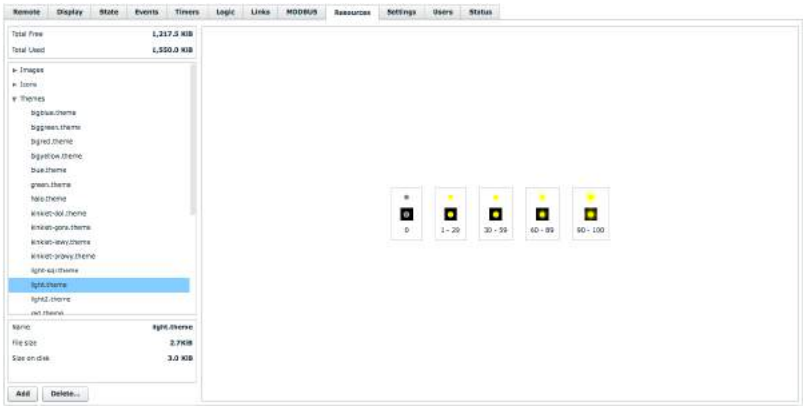


Fig 14.3: Preview of a theme in the Resources tab

Chapter 15
Status

The **Status** tab displays service information about the state of device and the individual components of operating system of the **DOMIQ/Base** module. The information contained in the **Status** can be helpful to diagnosis of failures in the **DOMIQ/Base** module, which could be a caused by errors in the handling of events, the **Logic** scripts, etc.

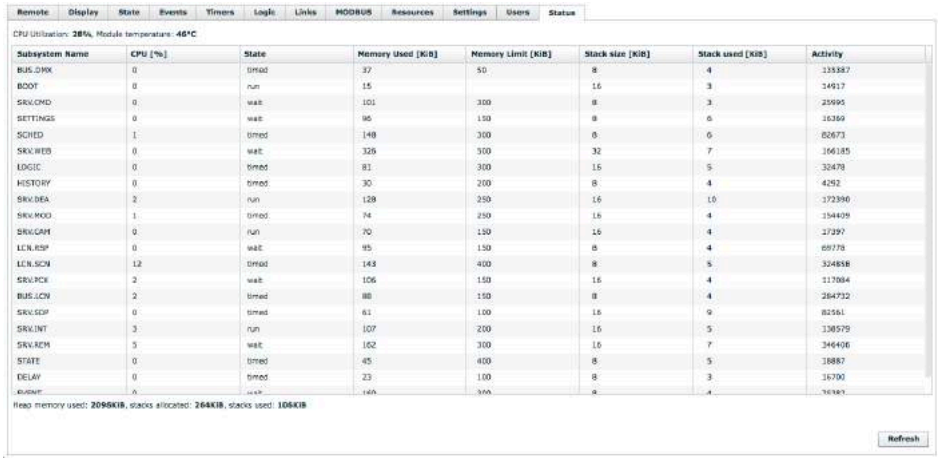


Fig 15.1: Status window

The **Status** tab contains following information:

- Complete and partial usage of the processor. If any subsystem uses a large portion of CPU resources during normal operation, it may be a symptom of improper operation.
- Module temperature
- Information on actual usage and the available memory for each subsystem.
- The number of queued commands. The maximum size is 32 commands. If the queue reaches this value and remains at this level, this means that the subsystem is not working correctly.

Chapter 16
Identifiers

This chapter covers the topic of state identifiers, commands and events which can be used within configuration of the **DOMIQ/Base** module. This chapter extends the chapter 4.5 „State, Commands and Events" on page 25.

16.1. Variables

Identifiers from the **Variables** group allow to store any numeric or string values. Variables names can be any, preceded by suitable prefix. Names cannot contain spaces. We recommend to assign short and self–descriptive names. If name contains multiple words use underscore sign or camel case: `my_variable_name` or `myVariableName`. Variables identifiers are often used in combination with timers, events and Lua scripts.

The table below contains overview of all identifiers from **Variables** group and their attributes. A detailed description of the individual identifiers, including use examples, is in the further part of this section.

Identifier	Brief description	State	Event	Command
MEM	Non–volatile variables	Y	Y	Y
VAR	Temporary variables	Y	Y	Y

MEM

`MEM` identifiers can be used to store any user defined variables such as numbers or strings. Value of a `MEM.` identifier preserves over a **`Base`** module restart. `MEM` identifiers are widely used in definition of timers, events and UI (User Interface) elements, whose settings shall be persistent.

Due to the limitation of non–volatile memory, always consider whether the use of `MEM.`identifiers is necessary.

	Identifier	Value	Description
State	`MEM.`**`name`**	any	Actual state of the MEM variable
Event	`E.MEM.`**`name`**	any	Information about the change of variable's state.
Command	`C.MEM.`**`name`**	any	Writing any value in a variable.

- `C.MEM.test=Testing MEM variables`
  Save to the `MEM.test` variable the following string: `Testing MEM variables`.
- `E.MEM.hour=10`
  The `MEM.hour` identifier has changed its state. The new value of the identifier is 10.
- `MEM.weekdays`
  State of the `MEM.weekdays` identifier.

VAR

The operation of **VAR** identifiers is very similar. The only difference is that, their values are discarded after restarting of the **`Base`** module.

16.2. LCN

Identifiers from the **LCN** group are dedicated to use in conjunction with the **LCN** system. Each identifier has a functionality related to the corresponding property in the **LCN** system.

The table below contains overview of all identifiers from **LCN** group and their attributes. A detailed description of the individual identifiers, including use examples, is in the further part of this section.

Identifier	Brief description	State	Event	Command
`LCN.output`	Controlling and state of a single dimmable output.	Y	Y	Y
`LCN.relay`	Controlling and state of a single relay.	Y	Y	Y
`LCN.relays`	Controlling of multiple relays.	N	N	Y
`LCN.sensor`	State of a binary LCN sensor.	Y	Y	N
`LCN.motor`	Controlling and state of a single shutter.	Y	Y	Y
`LCN.motors`	Controlling of multiple shutters.	N	Y	Y
`LCN.regulator`	Controlling and information about changes of state of a single regulator.	N	Y	Y
`LCN.value`	State and information about changes of state of a temperature sensor.	Y	Y	N
`LCN.key`	Information about pressing a LCN key.	N	Y	N
`LCN.scene`	Information about loading a LCN scene	N	Y	Y
`LCN.scenes`	Saving/loading light scenes.	N	N	Y
`LCN.transponder`	Information about receiving a LCN transponder code.	N	Y	N

LCN.output

This identifier is used to control and display state of a single dimmable output. It also informs about changes of value of an output. It's one of the most often used identifiers.

	Identifier	Value	Description
State	`LCN.output.`**`S`**`.`**`M`**`.`**`O`** **S** – Segment **M** – Module **O** – Output	`0–100`	Percentage value of a dimmable output.
Event	**`E`**`.LCN.output.`**`S`**`.`**`M`**`.`**`O`**	`0–100`	Percentage value of a dimmable output.
Command	**`C`**`.LCN.output.`**`S`**`.`**`M`**`.`**`O`**	`0–100`	Set value
		`on`	Turn on
		`off`	Turn off

- `C.LCN.output.0.10.1=on`
  Turn on the output 1, the module address: 10.
- `C.LCN.output.0.10.1=30;ramp:10`
  Set value of the output 1 to 30%, ramp=10, the module address: 10.
- `E.LCN.output.0.10.1=100`
  The output 1 in a module with the address 10 has turned on.
- `LCN.output.0.10.1`
  State of the output 1 in a module with the address 10.

In case of modules with 200-step-mode it is necessary to mark the option **200-steps-mode** in the **Settings** tab. Otherwise the commands will not be executed correctly and the status of the outputs will be displayed incorrectly. In case of installations with older generation modules (without 200-steps-dimming), reprogram the whole installation on the 50-steps-dimming.

LCN.outputs

The `LCN.outputs` allows to control all proportional outputs of a single LCN module simultaneously.

	Identifier	Value	Description
Command	**`C`**`.LCN.outputs.`**`S`**`.`**`M`**`.`**`O`** **`S`** – Segment **`M`** – Module **`O`** – Outputs	`toggle`	Set value to oposite
		`on`	Turn on
		`off`	Turn off

- `C.LCN.outputs.0.10.1-3=on; ramp=5`
  Turn on all outputs in a LCN module with the address 10, with ramp equal to 5.
- `C.LCN.outputs.0.10.1-3=off`
  Turn off all outputs in a LCN module with the address 10.
- `C.LCN.outputs.0.10.1-3=toggle`
  Toggle all outputs in a LCN module with the address 10.

LCN.relay

The `LCN.relay` identifier is used to control and display state of a single relay. It also informs about changes of value of a relay.

	Identifier	Value	Description
State	`LCN.relay.`**`S`**`.`**`M`**`.`**`R`** **S** – Segment **M** – Module **R** – Relay	`0`	Relay off.
		`1`	Relay on.
Event	**`E`**`.LCN.relay.`**`S`**`.`**`M`**`.`**`R`**	`0`	Relay has been turned off.
		`1`	Relay has been turned on.
Command	**`C`**`.LCN.relay.`**`S`**`.`**`M`**`.`**`R`**	`0,1`	Set value.
		`toggle`	Toggle value.
		`on`	Turn on
		`off`	Turn off

- `C.LCN.relay.0.10.3=on`
  Turn on the relay 3, the module address: 10.
- `C.LCN.relay.0.10.5=toggle`
  Toggle the relay 5, the module address: 10.
- `E.LCN.relay.0.10.1=0`
  The relay 1 has been turned off, the module address: 10.
- `LCN.relay.0.10.2`
  State of the relay 2, the module address: 10.

LCN.relays

The `LCN.relays` identifier is used to control max. 8 relays simultaneously.

Notice! Value of this command must contain 8 characters, where each character corresponds to a single relay.

	Idetifier	Value	Description
Command	`C.LCN.relays.`**`S`**`.`**`M`** **`S`** – Segment **`M`** – Module	`0`	Turn off
		`1`	Turn off
		`T`	Toggle
		`–`	Do not change

- `C.LCN.relays.0.10=––11––––`
  Turn on the relays 3 and 4 and do not change values of the remaining relays, the module address: 10.
- `C.LCN.relays.0.10=TT1100––`
  Toggle the relays 1 and 2, turn on the relays 3 and 4, turn the relays 5 and 6 off. Do not change value of the last two relays. The module address: 10.

LCN.sensor

This identifier was designed to display state and to inform about changes of state of a single LCN binary sensor.

	Identifier	Value	Description
State	`LCN.sensor.`**`S`**`.`**`M`**`.`**`N`** **S** – Segment **M** – Module **N** – Binary input number: 1 to 8	`0`	Input inactive
		`1`	Input active
Event	`E.LCN.sensor.`**`S`**`.`**`M`**`.`**`N`**	`0`	Input has become inactive
		`1`	Input has become active

- `E.LCN.sensor.0.12.2=1`
  The binary input 2 in a LCN module with the address 12 has become active.
- `LCN.sensor.0.10.5`
  State of the binary input 5, the module address: 10.

LCN.motor

The `LCN.motor` identifier is used to control a single shutter. It is recommended using this identifier for shutters with positioning. Otherwise use `LCN.relay` identifier.

	Identifier	Value	Description
State	`LCN.motor.`**`S.M.R`** **`S`** – Segment **`M`** – Module **`R`** – Shutter	`0–200`	0 – Shutter raised
			200 – Shutter closed
Event	`E.LCN.motor.`**`S.M.R`**	`0`	Shutter has stoped
		`1`	Shutter is moving
Command	`C.LCN.motor.`**`S.M.R`**	`up`	Move up
		`down`	Move down
		`stop`	Stop
		`learn`	Learning process of a LCN module: Shutter goes down and then up in order to measure movement time. Once a LCN module knows the movement time, you can move shutter to the fixed position.
		`0–200`	Moving a shutter to the fixed position.

- `C.LCN.motor.0.10.1=up`
  Move the shutter 1 up. The module address: 10.
- `C.LCN.motor.0.10.1=100`
  Set the shutter 1 to the center position. The module address: 10.
- `E.LCN.motor.0.10.1=1`
  The shutter 1 is moving. The module address: 10.
- `LCN.motor.0.10.1`
  Position of the shutter 1 in a LCN module with the address 10.

LCN.motors

This identifier is used to control up to 4 shutters simultaneously. It is recommended using this identifier for shutters with positioning. Otherwise use `LCN.relays` identifier.

	Identifier	Value	Description
Command	`C.LCN.motors.`**`S`**`.`**`M`**`.`**`R`** **S** – Segment **M** – Module **R** – Shutters **(1+2, 3+4, 1–4)**	`up`	Move up
		`down`	Move down
		`stop`	Stop
		`learn`	Learning process of a LCN module: Shutter goes down and then up in order to measure movement time. Once a LCN module knows the movement time, you can move shutter to the fixed position.
		`set:0–200`	Moving shutters to the fixed position.
		`0–200`	Moving shutters to the fixed position.

- `C.LCN.motors.0.10.1+2=up`
  Raise the shutters 1 and 2.
- `C.LCN.motors.0.10.1–4=100`
  Move all shutters to the center position.

LCN.regulator

The `LCN.regulator` identifier was designed to control LCN regulators. For temperature regulation, be aware that the LCN system uses its own temperature scale. The general formula for calculating temperature in the LCN is as follows: $t_{LCN}=t_m*10+1000$, where t_{LCN} is temperature in the LCN scale and t_m is temperature measured by a sensor.

Calculation examples:

- `21.5°C` equals `1215` in the LCN scale;
- `–11°C` equals `990` in the LCN scale;

	Identifier	Value	Description
Event	`E.LCN.regulator.`**`S`**`.`**`M`**`.`**`R`** **S** – Segment **M** – Module **R** – Regulator (1,2,t)	`0–65534`	Value in range from 0 to 65534 has been set.
Command	`C.LCN.regulator.`**`S`**`.`**`M`**`.`**`R`**	`lock`	Lock regulator
		`unlock`	Unlock regulator
		`current;change:`**`D`**	Change current value of a regulator by (+/–) **D**
		`programmed;change:`**`D`**	Change current value of a regulator by (+/–) **D**
		`set:0–65534`	Set regulator value
		`0–65534`	Set regulator value

- `C.LCN.regulator.0.10.1=1250`
  Set value of the regulator 1 to 1250, the module address: 10. For temperature control it equals to setting the temperature to 25°C.
- `C.LCN.regulator.0.10.1=lock`
  Lock the regulator 1 in a LCN module with the address 10.
- `C.LCN.regulator.0.10.1=current;change:–20`
  Set value of the regulator minus 20 relatively to the current setting. The regulator 1, the module address: 10.
- `E.LCN.regulator.0.10.1=1225`
  Value of the regulator 1 in a module with the address 10 has been set to 1225. For temperature control it equals to 22,5°C.

LCN.value

The `LCN.value` identifier is used to display and inform about changes of a temperature measured by a LCN sensors.

The temperature encoding in the LCN system was presented in the description of the `LCN.regulator` identifier.

	Identifier	Value	Description
State	`LCN.value.`**`S`**`.`**`M`**`.`**`C`** **S** – Segment **M** – Module **C** – Sensor (r1, r2, t – R1Var, R2Var, TVar).	`0–2000`	Current temperature
Event	`E.LCN.value.`**`S`**`.`**`M`**`.`**`C`**	`0–2000`	Informs about change of current temperature.

- `E.LCN.value.0.112.r1=1250`
  The temperature measured by the R1Var sensor of a LCN module with the address 112 reached value of 25 °C.
- `LCN.value.0.10.t`
  Temperature measured by the TVar sensor of a LCN module with the address 10.

LCN.variable

The new generation of LCN modules provides 12 variables (earlier 3). The `LCN.variable` ID enables to read the values of these variables and informs through events about change of the said values. Previously available R1Var, R2Var and TVar variables can still be read using `LCN.value` IDs, as described in the previous section, as well as using the `LLCN.variable` ID under variable numbers 1,2 and 3.

	Identifier	Value	Description
State	`LCN.variable.`**`S`**`.`**`M`**`.`**`Z`** **S** - Segment **M** - Module **Z**- Variable (from 1 to 12)	`0-2000`	Displays the current value of a given variable.
Event	`E.LCN.value.`**`S`**`.`**`M`**`.`**`Z`**	`0-2000`	The event informs about the value change of a given variable.

- `E.LCN.value.0.10.1=100`
  Variable 1 in the LCN module 10 reached value of 100.
- `LCN.value.0.10.5`
  Current value of the variable 5 in the module 10.

LCN.threshold (modules with firmware older than 17xxxx)

`LCN.threshold` identifier controls threshold values in LCN modules, informs about changes of the said values and dispalays their states. Threshold values are always changed towards the current value or the one which is programmed in the LCN module. The way of coding is analogous to the regulators. Therefore changing the value of 10 results in a change of 0.1 in the LCN module! 1111&%+6

	Identifier	Value	Description
State	`LCN.threshold.`**`S.M.W`** **`S`** - Segment **`M`** - Module **`W`** - Treshold value (`1-5`)	`0-65534`	Current state of the treshold value
Event	`E.LCN.threshold.`**`S.M.W`**	`0-65534`	The value from 0 to 65534 was set
Command	`C.LCN.threshold.`**`S.M.W`**	`current;change:`**`ZM`**	Change of **`ZM`** (+/-) value towards the current value
		`programmed;change:`**`ZM`**	Change of **`ZM`** (+/-) value towards the saved value
		`-1023 do 1023`	Change of the threshold value for the given number. Changes are coded identically as temperature in the case of regulators. This is an alternative command to the following command: `current;change:`**`ZM`**

- `C.LCN.threshold.0.10.1=-100`
  Reduce the current threshold value No. 1 by 1 in the LCN module 10.
- `C.LCN.threshold.0.10.2=current;change:20`
  Increase the value of the threshold No. 2 in module 10 by 0.2.
- `E.LCN.threshold.0.10.1=1225`
  The threshold value No. 1 in module 10 reached the value of 22.5.
- `LCN.threshold.0.10.1`
  The treshold value No.1 in the LCN module 10 is restored.

LCN.threshold (modules with 17xxxx firmware and newer)

`LCN.threshold` identifier controls threshold values in LCN modules, informs about changes of the said values and dispalays their states. Threshold values are always changed towards the current value or the one which is programmed in the LCN module. The way of coding is analogous to the regulators. Therefore changing the value of 10 results in a change of 0.1 in the LCN module!

	Identifier	Value	Description
State	`LCN.threshold.`**`S.M.R.W`** **S** - Segment **M** - Module **R** - Register (1-4) **W** - Treshold value (1 - 4)	`0-65534`	Current state of the treshold value
Event	`E.LCN.threshold.`**`S.M.R.W`**	`0-65534`	The value from 0 to 65534 was set
Command	`C.LCN.threshold.`**`S.M.R.W`**	`current;change:`**`ZM`**	Change of **ZM** (+/-) value towards the current value
		`programmed;change:`**`ZM`**	Change of **ZM** (+/-) value towards the saved value
		`-1023 do 1023`	Change of the threshold value for the given number. Changes are coded identically as temperature in the case of regulators. This is an alternative command to the following command: `current;change:`**`ZM`**

- `C.LCN.threshold.0.10.1.1=-100`
  Reduce the current threshold value No. 1 by 1 in the LCN module 10.
- `C.LCN.threshold.0.10.3.2=current;change:20`
  Increase the value of the threshold No. 2 in module in register 3 10 by 0.2.
- `E.LCN.threshold.0.10.4.1=1225`
  The threshold value No. 1 in module 10 in register 4 reached the value of 22.5.
- `LCN.threshold.0.10.1.1`
  The treshold value No.1 in the LCN module 10 in register 1is restored.

LCN.key

This identifier informs about receiving a „send keys" command sent to the address of a **Base** module. Using this feature you can perform some action after a LCN button was pressed.

In the **LCN** system there is no possibility for passive readout of which button was pressed. You should always use the „send keys" or „load scene" commands to assign actions to buttons.

	Identifier	Value	Description
Event	`E.LCN.key.`**`S.M.TP`** **S** - Segment **M** - Module **T** - Table: (A,B,C) **K** - Key: (1-8)	`hit/make/break`	„Send keys" packet has been received.

- `E.LCN.key=B3 make`
  The key 3 in the table B was pressed long. You can use this information to trigger another action in a **Base** module.
- `E.LCN.key=A1 hit`
  The key 1 in the table A was pressed briefly. You can use this information to trigger another action in a **Base** module.

LCN.sendkey

`LCN.sendkey` command executes a „send keys" command. This command can be sent to any LCN module, to each of four tables, with each of available actions (hit/make/brake). When a LCN module receives this command it executes action assigned to that particular key. Using this command you can define buttons on visualization or in the **Remote** application, which will trigger actions programmed directly in LCN modules.

	Identifier	Value	Description
Command	`C.LCN.sendkey.`**`S.M.TK`** **S** – Segment **M** – Module **T** – Table: (A,B,C,D) **K** – Key: (1–8)	`hit/make/brake`	„Send keys" command

- `C.LCN.sendkey.0.10.A1=hit`
  Send the `hit` command to the key 1, the table A, a LCN module with the address 10.
- `C.LCN.sendkey.0.10.B3=make`
  Send the `make` command to the key 3, the table B, a LCN module with the address 10.

LCN.scene

This event identifier informs about receiving a „load scene" packet sent to the address of a **Base** module.

	Identifier	Value	Desription
Event	`E.LCN.scene`	`1–100`	Number of the loaded scene.

- `E.LCN.scene=12`
  The command to load scene 12 has been received by the DOMIQ/Base.

LCN.scenes

The `LCN.scenes` identifier can be used to load/save LCN scenes. It allows to save/load scenes that includes bot dimmable outputs and relays. The way how LCN modules work makes saving/loading combined scenes (including both types of outputs) impossible. In order to load/save combined scenes you have to send two separate commands. By default the `LCN.scenes` command uses dimmable outputs.

	Identifier	Value	Description
Command	`C.LCN.scenes.S.M` **S** – Segment number **M** – Module number	**A:N;T:O;R** **A** – Action (load/save) **N** – Scene number (from 1 to 10) **T** – Output type (outputs, relays) **O** – Outputs numbers **R** (optional) – **ramp** parameter	Save or load a scene

All the examples refers to a module with the address 10, the segment number 0.

- `C.LCN.scenes.0.10=load:1`
  Load the scene 1, all dimmable outputs.
- `C.LCN.scenes.0.10=load:3;outputs:010;ramp:10`
  Load the scene 3, use the output 2, ramp=10.
- `C.LCN.scenes.0.10=save:1;outputs:110`
  Save the scene 1, use the outputs No. 1 and 2.
- `C.LCN.scenes.0.10=save:1;ramp:2`
  Save the scene 1, all dimmable outputs with the ramp=2.
- `C.LCN.scenes.0.10=load:1;relays:11------`
  Load the scene 1, use the relays No. 1 and 2.
- `C.LCN.scenes.0.10=load:2`
  Load the scene 2, use all relays.
- `C.LCN.scenes.0.10=save:2;relays:11------`
  Save the scene 2, use the first and the second relay.

LCN.transponder

The `LCN.transponder` event identifier informs about receiving a LCN transponder code. Using this identifier you can implement access control algorithms. Button presses are also send alongside transpoder codes, which are used in the IR remotes.

	Identifier	Value	Description
Event	`E.LCN.transponder.S.M` **S** – Segment **M** – Module	**KKKKKK K A** **KKKKKK** – Transponder code **K** – Key number **A** – Action (hit, make, break or other)	LCN module read a transpoder code (IR or RFID)

- `E.LCN.transponder.0.10=B308FF 1 hit`
  The transponder code B308FF has been received with pressing of the button 1 in the LCN module with the address 10. You can use this information to trigger any other actions in a **Base** module.

The action type is qualified as `other` when you use non–LCN transponder.

LCN.dali

The commands from the `LCN.dali` group are used to control the DALI ballasts connected to the outputs of the LCN modules. The implementation of the DALI standard in LCN enables sending commands to the DALI ballasts. However there is not any feedback from the ballasts. The commands can be divided in terms of application. The first subgroup are the commands sent to the individual DALI ballasts. The second subgroup are commands that are designed to control the entire DALI installation at the same time. The third group are group identifiers, which are used to control DALI lighting groups.

	Identifier	Value	Description
Command	`C.LCN.dali.`**`S.M.T.N`** **`S`** - Segment number **`M`** - Module number **`T`** - Command type. hree types of comands are available: `evg` - command sent to a single DALI ballast `group` - command controlling a group of DALI lighting `all` - command controlling the entire installation **`N`** - (only for commands type `evg` and `group`)- Ballast address/group number	`0-254`	Set brightness
		`on`	Switching on the given lamp. Brightness is set according to the **Level max.** parameter.
		`off`	The selected lamp is turned off.
		`up`	Step brightening
		`down`	Step dimming
		`stepup`	Step brightening by 1. The command does not result in turning on the lamp.
		`stepdown`	Step dimming by 1. The command does not result in turning off the lamp.
		`load:`**`xx`**	Loading the selected scene. **xx** stands for the scene number.
		`save:`**`xx`**	Saving the current scene. **xx** stands for the scene number.
		`dtr:`**`xx`**	Saving the value in the dtr. register. **xx** stands for the saved value.

- `C.LCN.dali.0.10.evg.1=100`
  Set value 100 (max) in ballast 1 connected to the LCN module 10.
- `C.LCN.dali.0.10.all=off`
  Turn off all the ballasts in the installation connected to the LCN module 10.
- `C.LCN.dali.0.10.group.5=load:3`
  Load the scene 3 in DALI ballasts which belong to group 5.

LCN.locks

This command blocks the buttons in any table of the LCN module.

	Identifier	Value	Description
Command	`C.LCN.locks.`**`S`**`.`**`M`**`.`**`T`** **S** - Segment **M** - Module **T** - Table (A,B,C lub D)	`0`	Do not lock
		`1`	Lock
		`T`	Switch
		`-`	Do not change the lock state

- `C.LCN.locks.0.10.A=0011----`
  Unlock the buttons No. 1 and 2, lock the buttons No. 3 and 4 in table A in module 10. Other buttons remain unchanged.

LCN.text

LCN.text command allows you to display any text on displays of the LCN wall buttons series GT-D.

	Identifier	Value	Description
Command	C.LCN.text.**S**.**M**.**L** **S** - Segment **M** - Module **T** - Text line (1-4)	any text	Displays any text in the given line on the display of the LCN-GT-D panel.

- C.LCN.text.0.10.1=”Any text”.
  Display "Any text" in the first line of the buttons display conntected to the LCN module 10.

LCN.groups

`LCN.groups` allows you to manage LCN groups.

	Identifier	Value	Description
Command	`C.LCN.groups.`**`S`**`.`**`M`** **S** - Segment **M** - Module **T** - Text line (1-4)	`clear`	Command removes assignment to all the groups.
		`join:xx`	Command assigns LCN module to the given group. **xx** stands for the group number.
		`leave:xx`	Command removes the assignment to a given group. **xx** stands for the group number.

- `C.LCN.groups.0.10=clear`
  Remove assignment to all groups of the module 10.
- `C.LCN.groups.0.10=join:5`
  Assign the LCN module 10 to group 5.
- `C.LCN.groups.0.10=clear:8`
  Remove assignment of the module 10 from group 8.

Group commands

All identifiers (with the exception of those that generate only events) from the LCN group have their group equivalent. The command sent using these identifiers are sent simultaneously to all LCN modules assigned to a given group. Group identifiers do not generate events and do not have the state. However, sending commands to the group generates the appropriate events within the modules which belong to this group and updates their status according to the received command. The syntax of the group commands is slightly different from the commands sent to the single LCN modules. There is no dot after the segment number and the number is preceded by the letter "g" (see examples below).

- `C.LCN.output.0g5.1=100`
  Set output 1 in all modules belonging to group 5 to value of 100.
- `C.LCN.regulator.0g8.1=1240`
  Set all r1 regulators in modules that belong to group 8 to the value of 24°C.

16.3. IDS

Identifiers from the **IDS** group are dedicated to integrate **DOMIQ** with the **SATEL** alarm system. Connections required for integration are presented in the following diagram. A **SATEL INT–IORS** module is shown as example and is not needed for integration.

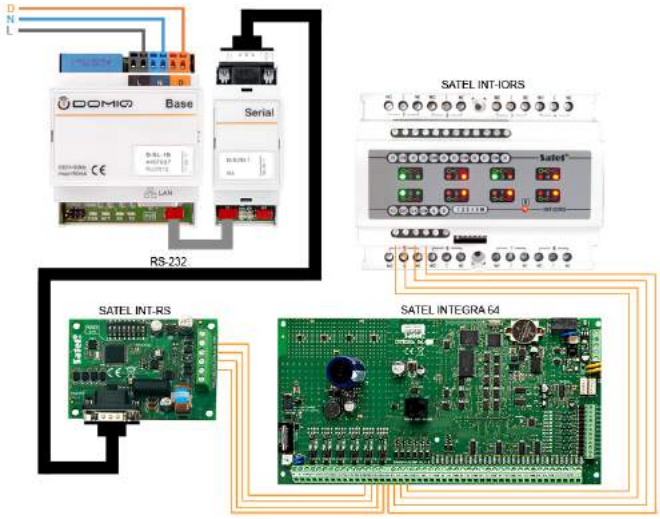


Fig 16.1: Wiring diagram of DOMIQ and Satel devices

The table below contains overview of all identifiers from IDS group and their attributes. A detailed description of the individual identifiers, including use examples, is in the further part of this section.

Identifier	Brief description	State	Event	Command
`IDS.input`	State and information about changes of state of a single alarm panel input.	Y	Y	N
`IDS.output`	Controlling and information about changes of state of a single alarm panel output.	Y	Y	Y
`IDS.armed`	Controlling and state of a single alarm zone.	Y	Y	Y
`IDS.entry`	Informs about presence of a human in a zone with entry time.	N	Y	N
`IDS.exit`	Informs about presence of a human in a zone with exit time.	N	Y	N
`IDS.alarm`	Informs about alarm event in a particular zone.	N	Y	N

IDS.input

The `IDS.input` identifier is used to display state and to inform about any changes of state of a single alarm panel input.

	Identifier	Value	Description
State	`IDS.input.`**N** **N** – Input number	`0`	Input inactive
		`1`	Input active
Event	`E.IDS.input.`**N**	`0`	Input has become inactive
		`1`	Input has become active

- `E.IDS.input.1=1`
  Alarm input 1 has changed its value to 1. An alarm input can be for example a PIR sensor or a reed relay etc. If we assume that our alarm input is a reed relay mounted on a window frame, then the above example should be understood as: *Window No.... has been opened.*
- `IDS.input.16`
  State of the alarm input 16.

IDS.output

The `IDS.output` identifier was designed to control a single alarm output. It is also used to display state of an output and informs about any changes of state of an output.

Any change of the output value requires authentication using the PIN code of user authorized to control of given output.

	Identifier	Value	Description
State	`IDS.output.`**N** **N** – Output number	`0`	Output inactive
		`1`	Output active
Event	`E.IDS.output.`**N**	`0`	Output has been deactivated
		`1`	Output has been activated
Command	`C.IDS.output.`**N**	`0;pin:xxxx`	Turn the output off
		`1;pin:xxxx`	Turn the output on

- `C.IDS.output.1=1;pin:1234`
  Turn the output 1 on with PIN code: 1234.
- `E.IDS.output.12=0`
  The alarm output 12 has been deactivated.
- `IDS.output.8`
  The state of the output 8.

IDS.armed

The `IDS.armed` identifier can be used to arm/disarm a single alarm zone, to inform about any changes of state of an alarm zone and also to display its current state.

Any change of an alarm zone state requires authentication using the authorized user PIN code.

	Identifier	Value	Description
State	`IDS.armed.`**Z** **Z** – Zone number	`0`	Zone disarmed
		`1`	Zone armed
Event	`E.IDS.armed.`**Z**	`0`	Zone has been disarmed
		`1`	Zone has been armed
Command	`C.IDS.armed.`**Z**	`0;pin:xxxx`	Disarm zone
		`1;pin:xxxx`	Arm zone

- `C.IDS.armed.1=1;pin:1234`
  Arm the zone 1, use the PIN code: 1234.
- `E.IDS.armed.2=0`
  The alarm zone 2 has been disarmed
- `IDS.armed.3`
  State of the alarm zone 3.

IDS.entry

The `IDS.entry` event identifier is used to inform about starting the countdown the entry time, after someone entered a zone with entry time. The *entry time* means the time after which an alarm will start, unless user disarms it. If you define a zone with the entry time, you can create a **Base** module event, which will be triggered when someone enters this zone, for example turning on the lights in the hall etc.

	Identifier	Value	Description
Event	`E.IDS.entry.`**Z** **Z** – Zone number	`0`	The countdown ended. In a moment you will get an event informing about disarming an alarm or that an alarm went off.
		`1`	Someone entered the zone. The countdown has started.

- `E.IDS.entry.1=1`
  Someone has entered the zone 1, which has set the entry time.

IDS.exit

The `IDS.exit` identifier works similarly to `IDS.entry`. It informs about starting the countdown the exit time. The *exit time* means the time after which the alarm will start, if you do not leave a zone after it was armed. If you define a zone with the exit time, you can create a **Base** module event, which will be triggered after leaving this zone, for example turning off all lights in a building.

	Identifier	Value	Description
Event	`E.IDS.exit.`**Z** **Z** – Zone number	`0`	The countdown ended. In a moment you will get an event informing that an alarm zone has been armed.
		`1`	The countdown the exit time has started.

- `E.IDS.exit.1=1`
  The countdown the exit time in the zone 1 has started.
- `E.IDS.exit.2=0`
  The countdown the exit time in the zone 1 has ended.

IDS.alarm

The aim of this identifier is to inform about an alarm in a particular zone.

	Identifier	Value	Description
Event	`E.IDS.alarm.`**Z** **Z –** Zone number	`0`	Alarm has been turned off
		`1`	Alarm!

- `E.IDS.alarm.1=1`
  Alarm in the zone 1.
- `E.IDS.alarm.1=0`
  Alarm in the zone 1 has been turned off.

16.4. DMX

The DMX identifier is dedicated to integrate the **DOMIQ** devices with devices that use the DMX512 protocol for control of the LED lighting.

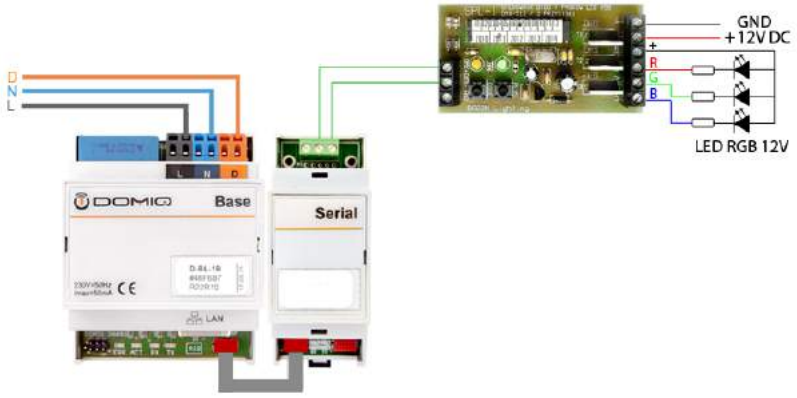


Fig 16.2: Wiring diagram of DOMIQ devices and DMX512 RGB LED driver.

The DMX identifier is used to control a single DMX512 slot, displaying its state and to inform about changes of value of a slot. Analogically to other light controls, DMX values use range from 0 to 100, but internally they are converted into typical DMX range – 0 to 255. It is possible to enter fractional values.

More detailed information regarding the integration of **DOMIQ** with DMX512 devices is in the **„RGB LED with DMX"** tutorial. The tutorial is available on our website **www.domiq.eu**, **Tutorials** section.

	Identifier	Value	Description
State	`DMX.N` **N** – Slot number	`0–100`	Current slot value
Event	`E.DMX.N`	`0–100`	Slot value has changed
Command	`C.DMX.N`	`0–100`	Change slot value

- `DMX.1`
  Value of the slot No.1.
- `C.DMX.100=50.3`
  Set value of the slot No.100 to 50.3
- `E.DMX.2=2`
  Value of the slot 2. has changed to 2.

16.5. MODBUS

The MODBUS identifiers are dedicated to integrate **DOMIQ** with devices that use the MODBUS protocol.

Identifiers names are based on the names of a particular MODBUS device attributes, given by user during setting integration parameters (the **MODBUS** tab).

	Identifier	Value	Description
State	MODBUS.**I.D.R** **I** – Interface name **D** – Device name **R** – Register name	Number	Current value of a MODBUS register
Event	E.MODBUS.**I.D.R**	Number	Information about change of value of a MODBUS register
Command	C.MODBUS.**I.D.R**	Number	Setting value of a MODBUS register

- MODBUS.ser.meter.energy
  State of the register called energy in the device called meter, which is assigned to the ser interface.
- E.MODBUS.tcp.meter.energy=125.7
  The energy register in the device called meter, which is assigned to the tcp interface has reached value of 25.4.
- C.MODBUS.int.heat.temp1=25
  Set value of the temp1 register to 25. Register is in the device named heat, which is assigned to the int interface.

Practical examples of using MODBUS identifiers can be found in the following tutorials: **„Energy Meters and MODBUS"**, **„Base and MODBUS weather station"**. The first tutorial shows how to integrate **DOMIQ** with a digital energy meter using MODBUS protocol. The second presents integration with the **Elsner Elektronik P03/3 MODBUS weather station**. Both tutorials are available on our website **www.domiq.eu**, **Tutorials** section.

16.6. SER

Identifiers presented in this section are used to integrate **DOMIQ** with devices using standard serial interface: RS–232 and RS–485.

In order to integrate a **DOMIQ/Base** module with RS–232 devices **DOMIQ/Serial–2SG** is required. **DOMIQ/Serial–4SG** is used to connect devices using RS–485.

A typical application of RS–232 is short distance, point to point communication with a single device such as. multiroom audio system (NuVo, Crestron, AMX), GSM modem or printer.

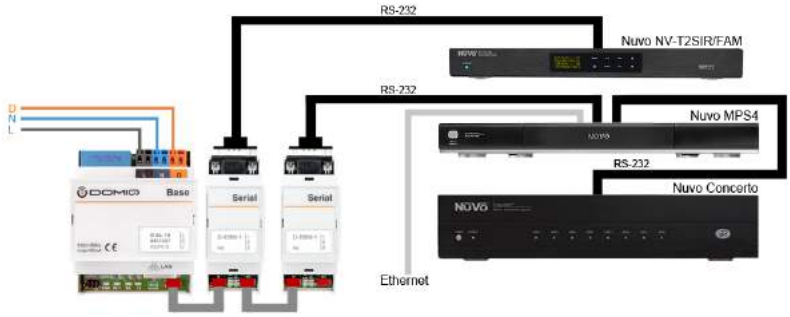


Fig 16.3: Integration of DOMIQ with NuVo multiroom audio system.

The table below contains overview of all identifiers from **SER** group and their attributes. A detailed description of the individual identifiers, including use examples, is in the further part of this section.

Identifier	Brief description	State	Event	Command
`LC.SER.config`	Connection parameters configuration	N	N	Y
`LC.SER.line`	Sending or receiving a full line of text	N	Y	Y
`LC.SER.send`	Sending data	N	N	Y

LC.SER.config

The LC.SER.config identifier is used to configure communication parameters of RS–232 or RS–485 interfaces.

	Identifier	Value	Description
Command	LC.SER.config.**N** **N** – Module number (1,2)	**S F** **P** – Transmission speed: (9600, 19200, 38400, 57600) **F** – Frame format: (8N1, 8N2, 8O1, 8E1) Notice! There is a space between transmission speed and frame format parameters.	Interface configuration parameters

The following examples are taken from the tutorial **"Multiroom NuVo and RS–232"** describing the integration of **DOMIQ** with **NuVo** multiroom audio system. The tutorial can be downloaded from our website **www.domiq.pl**., **Tutorials** section.

- LC.SER.config.1=57600 8N1
  Set transmission speed to 57600bit/s, frame format 8N1 in a **DOMIQ/Serial–SG** with the address 1.

This command have to be sent each time, when a **Base** module is restarted. To solve this issue, the most convenient way is to place a single command in the **Logic** tab. Use the following syntax: command ("LC.SER.config.A=P F"), replace A, P and F using proper configuration values.

LC.SER.line

This identifier is used to send any text to a RS–232 or RS–485 interface. It also informs about receiving data.

	Identifier	Value	Description
Command	`LC.SER.line.`**`N`** **`N`** – Module number (1,2)	`Text`	Sending any text to a serial interface
Event	`LE.SER.line.`**`N`**	`Tekst`	Information about receiving a message

The following examples are taken from the tutorial **"Multiroom NuVo and RS–232"** describing the integration of **DOMIQ** with **NuVo** multiroom audio system.

- `LC.SER.line.1=*T'A'FM98.8`
  Set the tuner A to the frequency of 98.8MHz.
- `LC.SER.line.1=*Z1ON\r*Z1SRC5\r*Z1VOL50`
  It is an example of sending multiple commands in a single message: turn on the first music zone and assign the audio source No. 5 to it, set volume to 50.
- `LE.SER.line.1=#Z1OFF`
  The music zone No.1 has been turned off.

LC.SER.send

This identifier is used to send any text to a RS–232 or RS–485 interface. In contrast to the `LC.SER.line`, this command does not contain an end–of–line marker. It allows to send several independent data (text) portions as a single line.

	Identifier	Value	Description
Command	`LC.SER.send.`**`N`** **`N`** – Module number (1,2)	`Text`	Sending any text to a serial interface

- Five independent `LC.SER.send` commands have been sent:
  - `LC.SER.send.1=An`
  - `LC.SER.send.1=example`
  - `LC.SER.send.1=of a`
  - `LC.SER.send.1=RS232`
  - `LC.SER.send.1=message`

As a result we will get a single message: *An example of a RS232 message.*

16.7. TCP

TCP identifier is used to integrate the **DOMIQ** system with other devices using the TCP protocol. The `C.TCP.send` command is used for that purpose. It allows to send any commands using the TCP protocol.

	Identifier	Value	Description
Command	`C.TCP.send.`**`A.B.C.D:P`** **`A.B.C.D`** – IP address **`P`** – Port number	`Data`	Sending any data using TCP protocol.

- `C.TCP.send.192.168.10.180:5004=anyData`
  Sending `anyData` message to a device with the IP address `192.168.10.180:5004` using the TCP protocol.

16.8. UDP

UDP identifier is used to integrate the **DOMIQ** system with other devices using the UDP protocol. The `C.UDP.send` command is used for that purpose. It allows to send any commands using the UDP protocol.

	Identifier	Value	Description
Command	`C.UDP.send.`**`A.B.C.D:P`** **`A.B.C.D`** – IP address **P** –Port number	`Data`	Sending any data using TCP protocol.

- `C.UDP.send.192.168.10.180:1234=HELLO`
  Sending `HELLO` message to a device with the IP address `192.168.10.180:1234` using the UDP protocol.

16.9. Communication

Identifiers from the **Communication** group was designed for interaction of the **DOMIQ** system with users. The **Communication** identifiers are usually associated with occurrence of another event in a building automation system.

The table below contains overview of all identifiers from **Communication** group and their attributes. A detailed description of the individual identifiers, including use examples, is in the further part of this section.

Identifier	Brief description	State	Event	Command
DISPLAY.screen	Toggles visualization screen	N	N	Y
?	Displays any message on a **DOMIQ/Display** screen.	N	N	Y
?	Displays any messages on devices with a **DOMIQ/Remote** application, located within the local network, to which a **Base** module is connected.	N	N	Y
REMOTE.notify	Displays any messages on devices with the **DOMIQ/Remote** application.	N	N	Y

REMOTE.message

REMOTE.message identifier allows you to display any message on the screens of mobile devices (iPhone, iPad, iPod with your registered **DOMIQ/Remote** application), which can be found in the local network that the **DOMIQ/Base** module is connected to.

In order to display the message, the **DOMIQ/Remote** application must be turned on and connected to the **Base** module. You can connect triggering of this identifier with other events, similarly as in the case of the IDISPLAY.message identifier.

	Identifier	Value	Description
Command	C.REMOTE.message	Text	Displays any text information

- C.REMOTE.message=Alarm in zone 1.
  Display the following message: "Alarm in zone 1" on the screen of all mobile devices with registered **DOMIQ/Remote** application, which are located in the local network.
- C.REMOTE.message=Welcome in DOMIQ
  Display the following message: "Welcome in DOMIQ" on the screen of all mobile devices with registered **DOMIQ/Remote** application, which are located in the local network.

REMOTE.notify

The `REMOTE.notify` identifier allows to display any notification on mobile devices (iPhone, iPad, iPod) with paired **DOMIQ/Remote** application. Notification are displayed on the screen even when the **Remote** app is not running. Occasionally there could be delays in notifications delivery. This is due to path that notification has to go through to reach the target destination :

DOMIQ/Base > **DOMIQ** Server > **Apple** Server > User's Mobile Device (iPhone/iPad/iPod).

Notification are usually triggered by other events in a building automation system.

	Identifier	Value	Description
Command	`C.REMOTE.notify`	`Text`	Displays any messages on mobile devices with paired **DOMIQ/Remote** application.

- `C.REMOTE.notify=Fire alarm!`
  Display on each mobile device a notification with the following content: *„Fire alarm!"*.

Apple software iOS 5 and newer displays **Remote** notifications in the **Notification Center** of your mobile device. Clicking on a notification in the **Notification Center** will launch the **DOMIQ/Remote** application.

Fig 16.4: DOMIQ/Remote notifications

16.10. Links

Identifiers from the **Links** group are used to send network commands and to inform about events from other **DOMIQ**/**Base** modules, that are present in an installation.

	Identifier	Value	Description
Command	`NC.M.C` **M** – The name of a **Base** module to which the command is sent. **C** – Command	any	Any network command
Event	`NE.M.E` **M** – The name of a **Base** module to which the command is sent. **E** – Event	any	Any network event

- `NC.reception.C.VAR.fire.apartment1=1`
  Set the `VAR.fire.apartment1` variable to 1 in the **Base** module named `reception`.
- `NE.apartment1.E.IDS.alarm.1=1`
  An event informing about an alarm in the zone 1 has been sent by the **Base** module named `apartment1`.

16.11. NET

Identifiers from the NET group are used to support network variables. Each **Base** module assigned to a given group has the full access to the network variables which belong to a given group. The full access means:

- ability to read the state,
- ability to react on state changes (events),
- ability to change the state (commands),
- ability to create new network variables. The network variables do not require any declaration. In order to create a network variable, send the command to the non-existent network variable (see examples below).

	Identifier	Value	Description
State	NET.**G**.**N** **G** - Group name **N** - Variable name	any	Network variable state
Event	**E**.NET.**G**.**N**	any	This event informs about change of the network variable state
Command	**C**.NET.**G**.**N**	any	This command changes the state of the network variable.

- `C.NET.ground floor.wind = 10`
  Assign the value of 10 to the network variable `wind`. The network variable belongs to the group `ground floor`. If the variable `wind` does not exist, it will be created and the value of 10 will be assigned to this variable.
- `E.NET.ground floor.wind = 10`
  The network variable `wind`, which belongs to the group `ground floor` changed its value to 10.
- `NET.ground floor.wind`
  The current state of the network variable `wind` which belongs to the group `ground floor`.

16.12. DALI

The identifiers from the **DALI** group are used to control DALI lighting using the **DOMIQ/Light** module. Each identifier has a specific function which is connected with the DALI interface. Identifiers from the **DALI** group can be divided in terms of the the area of application. The first subgroup includes `DALI.1.all` identifiers which are designed to control the entire DALI installation at the same time . The second subgroup includes `DALI.1.evg` identifiers, which are dedicated to control a single DALI ballast and to inform about the status of ballast and its changes. The third group includes `DALI.1.group` identifiers, which are used to control DALI lighting groups.
The following table provides an overview of the types of DALI identifiers and their main features. A detailed description of the individual identifiers along with examples of application can be found in the following section of this chapter.

Group	Short description	State	Events	Command
`DALI.1.evg`	Group of identifiers to control the single DALI ballast.	Y	Y	Y
`DALI.1.group`	Group of identifiers to control lighting groups.	N	N	Y
`DALI.1.all`	Group of identifiers to control the entire DALI installation.	N	N	Y

DALI.1.evg

`DALI.1.evg` identifier is used to control and display the state of the single DALI ballast as well as to inform (event) about the change of its state.

	Identifier	Value	Description
State	`DALI.1.evg.`**`adr`** **`adr`** - DALI ballast address	`0-254`	Current brightness
Event	**`E`**`.DALI.1.evg.`**`adr`**	`0-254`	Event informs about the change of the current value.
Command	**`C`**`.DALI.1.evg.`**`adr`**	`0-254`	Set brightness
		`on`	Turning the selected lamp on. Brightness is set according to the **Max. value.**
		`off`	Turns the selected lamp off.
		`max`	Set max. brightness
		`min`	Set min. brightness
		`up`	Step brightening. The step depends on the **Brightness step** parameter.
		`down`	Step dimming. The step depends on the **Brightness step** parameter.
		`stepup`	Step brightening by1. The command does not result in turning the lamp on.
		`stepdown`	Step dimming by 1. The command does not result in turning the lamp off.
		`onandstepup`	Step brightening by1. Invoking this command when the lamp is turned off will result in its switching on and setting minimum brightness.
		`stepdownandoff`	Step dimming by 1. When the brightness reaches the minimum level, the command should turn the lamp off.
		`load:`**`xx`**	Loads the selected scene. **`xx`** stands for the scene number.
		`save:`**`xx`**	Saves the current brightness to the scene. **`xx`** stands for the scene number.
		`fadetime:`**`xx`**	The command sets the ramp in a given ballast. **`xx`** stands for the ramp value.
		`0-254;fadetime:`**`xx`**	By default the ramp is saved in the ballast memory and then each command changing the brightness is executed using a given ramp. In case of this command the brightness is changed with the ramp one time. After each command the ramp is reset to zero. **`xx`** stands for the value of ramp.
		`faderate:`**`xx`**	The command sets the step by which the brightness will be changed after the commands `up` or `down` are invoked. **`xx`** stands for the step value. The identifier equivalent of the command setting the **Brightness step** available in the **Installation preview** window.
		`dtr:`**`xx`**	Saves the value in the dtr. register. **`xx`** stands for the saved value.

- `DALI.1.evg.10`
  Restores the current value of the ballast 10.
- `E.DALI.1.evg.0=159`
  Lamp 0 was set to the value of 159.
- `C.DALI.1.evg.1=254`
  Set max. brightness in lamp 1.
- `C.DALI.1.evg.5=on`
  Switch the lamp 5 on. Brightness of the lamp will be set according to the parameter **Max**. **level** of the given ballast.
- `C.DALI.1.evg.1=off`
  Switch lamp 1 off.
- `C.DALI.1.evg.10=load:2`
  Call scene 2 in lamp 10.
- `C.DALI.1.evg.0=save:2`
  Save current value as scene 2.
- `C.DALI.1.evg.1=fadetime:10`
  Set **Ramp** in lamp 1 to the value of 10.
- `C.DALI.1.evg.1=200;fadetime:5`
  Set ballast 1 to the value of 200 with ramp 5.

In addition to the above listed identifiers, the **Base** module has three DALI status flags which have their status and can generate events.

	Identifier	Value	Description
State	`DALI.1.evg.`**`adr`**`.on` **`adr`** - address of the DALI ballast	`0-1`	Status flag informs that the ballast was switched on. 1 means that the lamp was turned on.
	`DALI.1.evg.`**`adr`**`.lampfail`	`0-1`	Status flag informs about ballast failure. 1 stands for failure.
	`DALI.1.evg.`**`adr`**`.powerfail`	`0-1`	Status flag informs about power supply failure. 1 stands for failure.
Event	`E.DALI.1.evg.`**`adr`**`.on`	`0-1`	Event informs that the lamp was switched on.
	`E.DALI.1.evg.`**`adr`**`.lampfail`	`0-1`	Event informs about lamp failure.
	`E.DALI.1.evg.`**`adr`**`.powerfail`	`0-1`	Event informs about power supply failure.

DALI.1.group

Identifiers from the `DALI.1.group` group are used to control the DALI lighting groups. The set of available commands is identical, as in the case of `DALI.1.evg` group but the commands are sent to all ballasts belonging to the given group at the same time. Identifiers from this group do not generate any events and do not have the state. However, sending commands to the group results in generating the appropriate events around the ballasts which belong to this group and update of their status according to the received command.

- `C.DALI.1.group.5=on`
  Turn on all lamps in the installation.
- `C.DALI.1.group.0=100`
  Set all the lamps belonging to the 0 group to the value of 100.
- `C.DALI.1.group.15=save:3`
  Save the current brightness of the ballasts which belong to group 15 as scene 3.

DALI.1.all

Identifiers from this group are dedicated to control the entire DALI installation at the same time. Commands which are sent using these identifiers are executed by all ballasts available in installation, regardless of whether the ballast has an assigned address or not.

The set of available commands is identical, as in the case of `DALI.1.evg` group but the commands are sent to all ballasts available in the installation at the same time. Sending the `C.DALI.1.all=on` command results in switching off all the ballasts in the installation. Identifiers from this group do not generate any events and do not have the state. However, sending commands to the group results in generating the appropriate events around the certain lamps and update of their status according to the received command. For instance if the `C.DALI.all=on` command was sent, each ballast will generate the event which informs about switching it on and update the state of brightness.

- `C.DALI.1.all=on`
  Turn on all lamps in the installation.
- `C.DALI.1.all=200`
  Set all the lamps in the installation to the value of 200.
- `C.DALI.1.all=load:5`
  Load the scene 5 in all ballasts.

16.13. UAV

Identifiers from the UAV group are designed to support audio/video devices using the UPNP protocol. The identifiers were created in order to integrate the **DOMIQ** system with **SONOS** players, however they are also supported by other devices that use UPNP Protocol. Identifiers from the UAV group can be divided into two groups:

- **Information identifiers** which provide information about the current state of the device and generate events which inform about the change of the state.
- **Control identifiers** which enable controlling of the UPNP devices.

Both groups have an identical syntax: `UAV.function.device`. Here `function` means the function name in **DOMIQ** and `device` is the name which is assigned to a given UPNP device, e.g. `UAV.volume.office`. UAV is the type of the identifier just as `LCN`, `IDS` etc.

Presented examples show integration of **DOMIQ** with **SONOS** players.

Information identifiers

The following table presents all the information identifiers together with examples of usage. In the following examples the name of the device *office* was used. Depending on your device the name is assigned on a permanent basis or you can modify it (for example, SONOS players). You can view the information with the most important information about the currently playing audio source.

Function	Example	Description
`album`	`UAV.album.office`	Restores the name of the currently played album
`creator`	`UAV.creator.office`	Name of the artist of the currently played song
`duration`	`UAV.duration.office`	Duration of the song
`mode`	`UAV.mode.office`	Play mode (e.g. normal, repeat, random etc.)
`state`	`UAV.state.office`	Current playback state: stop, pause, play.
`title`	`UAV.title.office`	Title of the played song.
`trackno`	`UAV.trackno.office`	Number of the played file.
`tracks`	`UAV.tracks.office`	Total file number in the given folder.
`uri`	`UAV.uri.office`	The path to the file/audio stream (e.g. online radio).
`volume`	`UAV.volume.office`	Current volume.

If the device does not provide the given information, then the identifier takes the value of 0.

Control identifiers

The functions from this group enables control of UPNP devices. As these are the commands they must be preceded by the prefix "C".

Function	Example/Syntax	Description
`control`	`C.UAV.control.office=play`	The function controls playing. The following values are available: • `stop` - stops playing. Playing after the `stop` command is started from the beginning of the list. • `play` - starts playing. • `pause` - pauses playing. • `next` - next song. • `prev` - previous song.
`mute`	`C.UAV.mute.office=1`	Playing faded out. The function may have the following values: 1 and 0. 1 means fading out.
`uri`	`C.UAV.uri.office=<track>`	The player is indicated the tractk to the file that you want to play. The track must indicate a file on your hard disk or network drive. This can be the address of the Internet radio stream. In the case of files on disks, you can specify the track or file with the playlist in m3u format. **IMPORTANT:** Invoking of the uri function does not automatically start playing. To start playing, after calling the uri function, call the control funtion with the play value.
`volume`	`C.UAV.volume.office=30`	The command controls the volume. After the equal sign enter the volume between 0 to 100.

Full application description of the UAV identifiers can be found in the tutorial '**Integration with SONOS** ' that describes the basic functions which integrate the **DOMIQ** and **SONOS** systems. The tutorial is available in **Tutorials** section on our website.

16.14. DISPLAY

Identifiers from the DISPLAY group are used to control the **DOMIQ/Display** touch panels.

Identifier name	Short description	State	Events	Command
DISPLAY.screen	Switches the visualization screen.	N	N	Y
DISPLAY.message	Displays the message on the **DOMIQ/Display** displays. Detailed description in chapter DISPLAY.message	N	N	Y
DISPLAY.switch	Switches the visualization screen without turning off the screensaver.	N	N	Y
DISPLAY.screensaver	Sets the time of turning off the screensaver.	N	N	Y
DISPLAY.sleep	Turns the screen off.	N	N	Y
DISPLAY.wake	Turns the screen on.	N	N	Y
DISPLAY.reload	Command restarting the panel.	N	N	Y
DISPLAY.calibrate	Invokes the calibration window of the touch screen.	N	N	Y
DISPLAY.volume	Sets click volume of the panel.	N	N	Y
DISPLAY.brightness	Sets the brightness of the screen.	N	N	Y
DISPLAY.name	Changes the name of the panel.	N	N	Y

The above mentioned identifiers are commands so they need to be preceeded with C .

NOTICE: All above commands may additionally contain the name of the panel at the end of the identifier, according to the following syntax: `command.panel_name`, e.g. `C.DISPLAY.screen.living_room`. This is particularly useful in the installations, in which more than one panel is connected to one **Base** module. The possibility of giving the names of the panels in commands allows you to send commands to each panel. If the command is called without specifying the name of the particular panel, then the command will be sent to all panels connected to the **Base** module. If the **Base** module is connected to one panel, you do not need to enter its name in commands.

DISPLAY.screen

`DISPLAY.screen` identifier is used to switch visualization screens. Invoking of the `DISPLAY.screen` command can be connected with other events in the building automation system, e.g. conditional events, timers and logical functions. The command interrupts the operation of the screensaver. Names of the screens are the same as the ones given in the visualization structure in the **Display** tab.

	Identifier	Value	Description
Command	`C.DISPLAY.screen`	`screen_name`	Displays the screen with a specific identifier without turning off the screensaver.

- `C.DISPLAY.screen=gate`
  Display the visualization with the `gate` identifier. The `gate` screen can display e.g. the view from the camera above the gate.
- `C.DISPLAY.screen.living_room=ground_floor`
  Display the visualization with the `ground_floor` identifier on the panel with the assigned name `living_room`.

DISPLAY.switch

`DISPLAY.switch` identifier is used to switch visualization screens. Invoking of the `DISPLAY.switch` command can be connected with other events in the building automation system, e.g. conditional events, timers and logical functions. The command does not interrupt the operation of the screensaver. Names of the screens are the same as the ones given in the visualization structure in the **Display** tab.

	Identifier	Value	Description
Command	`C.DISPLAY.switch`	`screen_name`	Displays the screen with a specific identifier without turning off the screensaver.

- `C.DISPLAY.switch=gate`
  Display the visualization with the `gate` identifier. The `gate` screen can display e.g. the view from the camera above the gate. The visualization will be switched without interrupting operation of the screensaver.
- `C.DISPLAY.switch.living_room=ground_floor`
  Display the visualization with the `ground_floor` identifier on the panel with the assigned name `living_room`. Do not interrupt operating of the screensaver.

DISPLAY.screensaver

`DISPLAY.screensaver` identifier is used to set the time after that the screensaver is turned on. The time is given in seconds and is counted from the last interaction of the user with the panel. The default value of the screensaver is 600s.

	Identifier	Value	Description
Command	`C.DISPLAY.screensaver`	`0-99999999`	Sets the time of turning on the screensaver.

- `C.DISPLAY.screensaver=60`
  Set the time of turning on the screensaver for 60 s in all panels.
- `C.DISPLAY.screensaver. office=120`
  Set the time of turning on the screensaver for 120 s in the `office` panel.
- `C.DISPLAY.screensaver=0`
  Turn off the screensaver in all panels.

DISPLAY.sleep

`DISPLAY.sleep` identifier is designed to turn off the screen of the panel.

	Identifier	Value	Description
Command	`C.DISPLAY.sleep`	`1`	Turns off the screen of the panel.

- `C.DISPLAY.sleep=1`
  Turn off the screens of all panels connected to the **Base** module.
- `C.DISPLAY.sleep.bedroom=1`
  Turn off the screen of the panel `bedroom`.

DISPLAY.wake

`DISPLAY.wake` identifier is the inverse of the `DISPLAY.sleep` command and is used to turn the screen on. The command has the same effect as touching the turned off screen.

	Identifier	Value	Description
Command	`C.DISPLAY.wake`	`1`	Turns on the screen of the panel.

- `C.DISPLAY.wake=1`
  Turn on the screens of all panels connected to the **Base** module.
- `C.DISPLAY.sleep.bedroom=1`
  Turn the screen of the panel `bedroom`.

DISPLAY.reload

The `DISPLAY.reload` command is used to restart the panel. Invoking this command equates pressing the **Restart** button in the **Display** tab.

	Identifier	Value	Description
Command	`C.DISPLAY.reload`	`1`	Restart of the panel

- `C.DISPLAY.reload=1`
  Restart all panels connected to the **Base** module.
- `C.DISPLAY.reload.bedroom=1`
  Restart the panel `bedroom`.

DISPLAY.calibrate

The command `DISPLAY.calibrate` enables invoking the calibration window of the touch screen.

	Identifier	Value	Description
Command	`C.DISPLAY.calibrate`	`1`	Invokes the calibration window of the touch screen.

- `C.DISPLAY.calibrate=1`
  Invoke the calibration window of the touch screen in all panels connected to the **Base** module.
- `C.DISPLAY.calibrate.bedroom=1`
  Invoke the calibration window of the touch screen in panel `bedroom`.

DISPLAY.volume

The `DISPLAY.volume` command controls the click volume in case of the user interaction with the panel. The volume can also be set in the configuration menu of the panel. The configuration menu is invoked by pressing the **Setup** button at the time of the turning on the Panel. The volume value is saved in the memory of the panel.

	Identifier	Value	Description
Command	`C.DISPLAY.volume`	`0-100`	Sets click volume when the panel is touched.

- `C.DISPLAY.volume=100`
  Set the maximum value in all panels connected to the **Base** module.
- `C.DISPLAY.volume.living_room=50`
  Set the volume to half in the panel `living_room`.

DISPLAY.brightness

The `DISPLAY.brightness` command controls the brightness of the screen. This is particularly useful in order to diversify the panel brightness between day and night mode (so that it does not blind at night). Then using two timers you can define a command which will change the brightness of the panel/panels. Brightness can also be set in the configuration menu of the panel. The configuration menu is invoked by pressing the **Setup** button at the time of switching on the panel. The brightness value is saved in the memory of the panel.

	Identifier	Value	Description
Command	`C.DISPLAY.brightness`	`0-100`	Sets the brightness of the screen.

- `C.DISPLAY.brightness=100`
  Set the brightness in all panels connected to the **Base** module.
- `C.DISPLAY.brightness.living_room=50`
  Set the brightness to half in the `living_room` panel.

DISPLAY.name

The command `DISPLAY.name` allows you to change the name of the **DOMIQ/Display** panel. The default name of each new panel is `DOMIQ`. In installations where the **Base** is connected to several **Display** panels, it is required to give different names to all panels. The name of the panel can also be set in the configuration menu of panel. The configuration menu is invoked after pressing the **Setup** button at the time of the turning on the panel. The name is saved in the panel memory.

	Identifier	Value	Description
Command	`C.DISPLAY.name`	`name`	Assigns/changes the name of the panel

- `C.DISPLAY.name.living_room=bedroom`
  Change the panel name from `living_room` to `bedroom`.