

Welcome

to our

 **OPUS Projektor 2020** Training



## Preliminary notes

Hardware: OPUS A3/A6/A8 Hardware Manual

Software: Operating System (OS): OPUS C / C++ Developer Guide  
Projektor-Tool: HTML Help  
CODESYS: Codesys Operator Manual  
OPUS Devices Update Manual

Operators must read these installation instructions, particularly the safety information, and must be familiar with the operation of the equipment.

The following topics among others can be found in the above hardware and software documents : Used Instructions Types, Safety instructions, guarantee and liability, Intended Use, Getting Started and Technical Documentation

## Legal notes

All operating manuals and products names cited in the operating Trademarks manual are registered trademarks of Topcon Electronics GmbH & Co KG All rights reserved, including translation.

No part of this operating manual may be reproduced, in any form, (print, photocopy, microfilm, or a different process), nor may it be processed, duplicated, or distributed using electronic systems, without written permission from Topcon Electronics GmbH & Co KG, Geisenheim.

All previous versions are rendered obsolete with publication of new manuals. All information contained herein is subject to correction, manufacturer is not liable for any errors in this presentation

# Agenda

1. Introduction
2. Installation of the Projektor Tool
3. Getting to know the Projektor Tool
4. Updating your device
5. First project
6. CAN communication
7. Extended agenda
8. FAQ

# 1. Introduction – About me

- Dipl.-Ing. Electrical Engineer
- Over 9 years experience for Wachendorff Elektronik / Topcon Electronics
- Main work areas:
  - Customer support via phone / email
  - Trainings and workshops
  - Creation and maintenance of trade show / example projects and technical documentation

# 1. Introduction – Training targets

Training targets:

- Understanding the project design concept of the Projektor Tool
- Learning the basic objects and their usage
- Being able to configure the CAN protocol you need
- Understanding what JavaScript is needed for

# 1. Introduction – Projektor vs. PClient

The Toolchain consists of:

Projektor Tool on your PC



PClient\* and Linux OS on your device



- Used to create projects
- One tool for A3, A6 G1, A6 G2 & A8

Your project

PClient

Linux OS

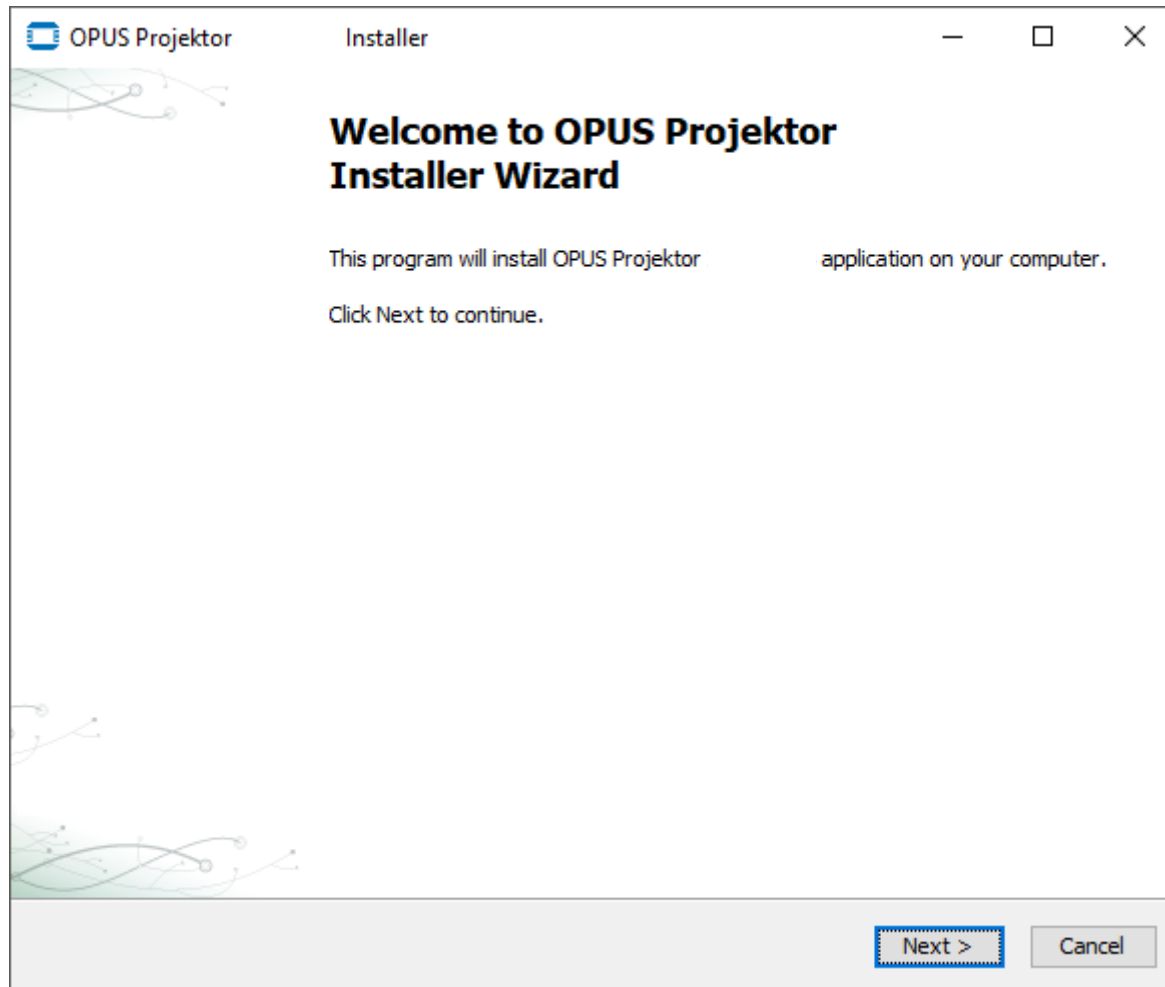
\*Projektor Client

# Agenda

1. Introduction
2. Installation of the Projektor Tool
3. Getting to know the Projektor Tool
4. Updating your device
5. First project
6. CAN communication
7. Extended agenda
8. FAQ



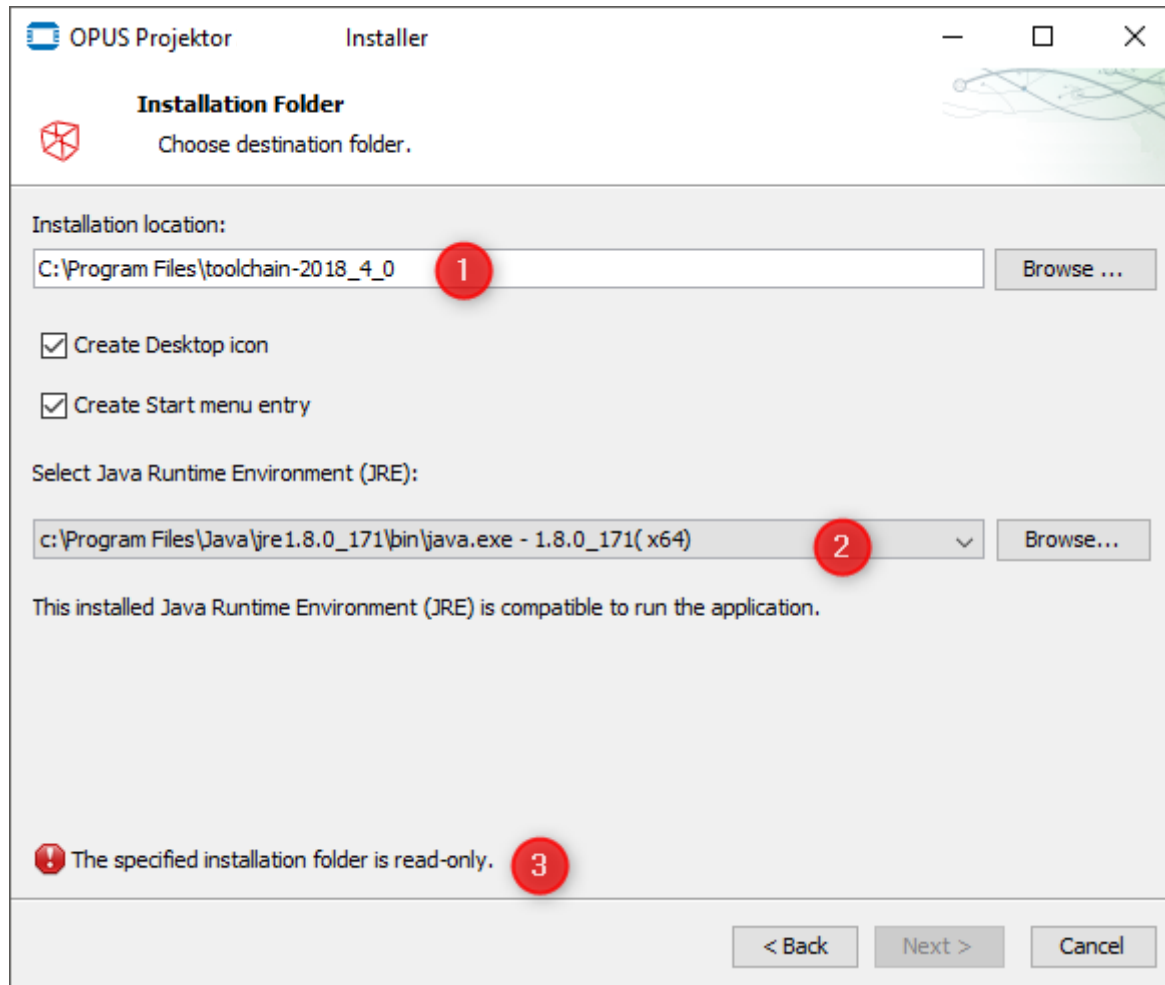
## 2. Installation of the Projektor Tool



The installation file for the Projektor Tool can be found under \Projektor

Please install the Projektor Tool by double-clicking the installation file

## 2. Installation of the Projektor Tool



Click through the steps in the installation:

- Accept the license agreement
- Choose an installation folder (1)  
(depending on the user rights, you can't install to program files (3), choose a folder where you have full rights)
- Choose the latest Java **8** version (**64 bit!**)
- Click Next to continue, then Install to start the installation

start the Projektor Tool after the installation (first start may take longer)

Copy the training material folder from the USB stick to your hard drive

# Agenda

1. Introduction
2. Installation of the Projektor Tool
3. Getting to know the Projektor Tool
4. Updating your device
5. First project
6. CAN communication
7. Extended agenda
8. FAQ



### 3. Getting to know the Projektor Tool – Program overview I

OPUS Projektor v2019.4.0.4 Licensed - BETA

File Edit Simulation Update View Communication Tools Window Help

36817685.0.kb

Current Language: english (en) Current Unit: metric Simulate:  PClient  Variables (Re)Start Default Theme: modern\_dark

**Projects**

- Demo\_project
  - Pages (1)
    - Home Page [35] **1**
    - DataMask
      - Frame 1 [36]
      - Softkey Frame Left [37]
      - Softkey Frame Right [38]
    - Alarms (0)
    - Communication
    - Virtual Keyboards (0)
    - JavaScripts (2)

**Start Page** Home Page [35]

**Palette**

- Container
  - Frame
  - Container **3**
- Common Controls
  - Button
  - String Field
  - Numeric Field **2**
  - Picture Graphic
  - Table
  - List
- Measurement Controls
  - Meter
  - Gauge
  - Gauge 270
  - Gauge 180
  - Gauge 90
  - Arched Bargraph
  - Linear Bargraph
  - Graph
- Lamps and Switches
  - Lamp
  - Power Switch

**Properties**

<No Properties>

**Variable View - Demo\_project** Output - Information

Var  Hide pre-defined variables **5**

Group Name	Name	Length (bits)	Index	SubIndex	Owner
Alarms	@AlarmCurrent	16	0x2430	0x02	PClient
Alarms	@AlarmCurrentPriority	8	0x2430	0x05	PClient
Alarms	@AlarmEnqueued	16	0x2430	0x04	PClient
Alarms	@AlarmShow	16	0x2430	0x01	PClient
Alarms	@IsAlarmEnqueued	16	0x2430	0x03	PClient

**Satellite Window** Navigator

- 1 – Project Tree
- 2 – Main Window
- 3 – Object Palette
- 4 – Object Properties
- 5 – Variable View

### 3. Getting to know the Projektor Tool – Program overview I



File Edit Simulation Update View Communication Tools Window Help → Menu bar

→ Tool bar

Projects X Demo\_Project\_1 → Project name

Pages (1) → Homepage

- Home Page [41]
- DataMask
  - Frame 1 [42]
  - Softkey Frame Left [47]
  - Softkey Frame Right [48]
- Function Key [43]
- Home Key [44]
- Escape Key [45]
- Encoder [46]

Object IDs

Image Library tab

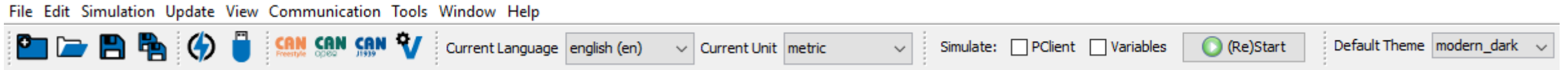
Symbol Library tab

Palette objects

Object properties and events

Home Page [35] - Properties	
Properties	Events
General	
ID	35
Type ID	1
Name	Home Page
Object Type	Page
Skin Properties	
Background Color	255,255,255
Background Image	
Page Specific	
Connect All Softkey Frames	<input checked="" type="checkbox"/>
Display Video	None
Video Rotation	0
Video Camera	Camera 1
Video Type	Analog Video Input
Video Post	None

# 3. Getting to know the Projektor Tool – Program overview II



New project  
 Open project  
 Save project  
 Project download  
 OS/PClient download  
 CAN mappings  
 Variable manager

PClient Simulation Current Theme

Search box

Variable View - Demo\_Project\_1

Output

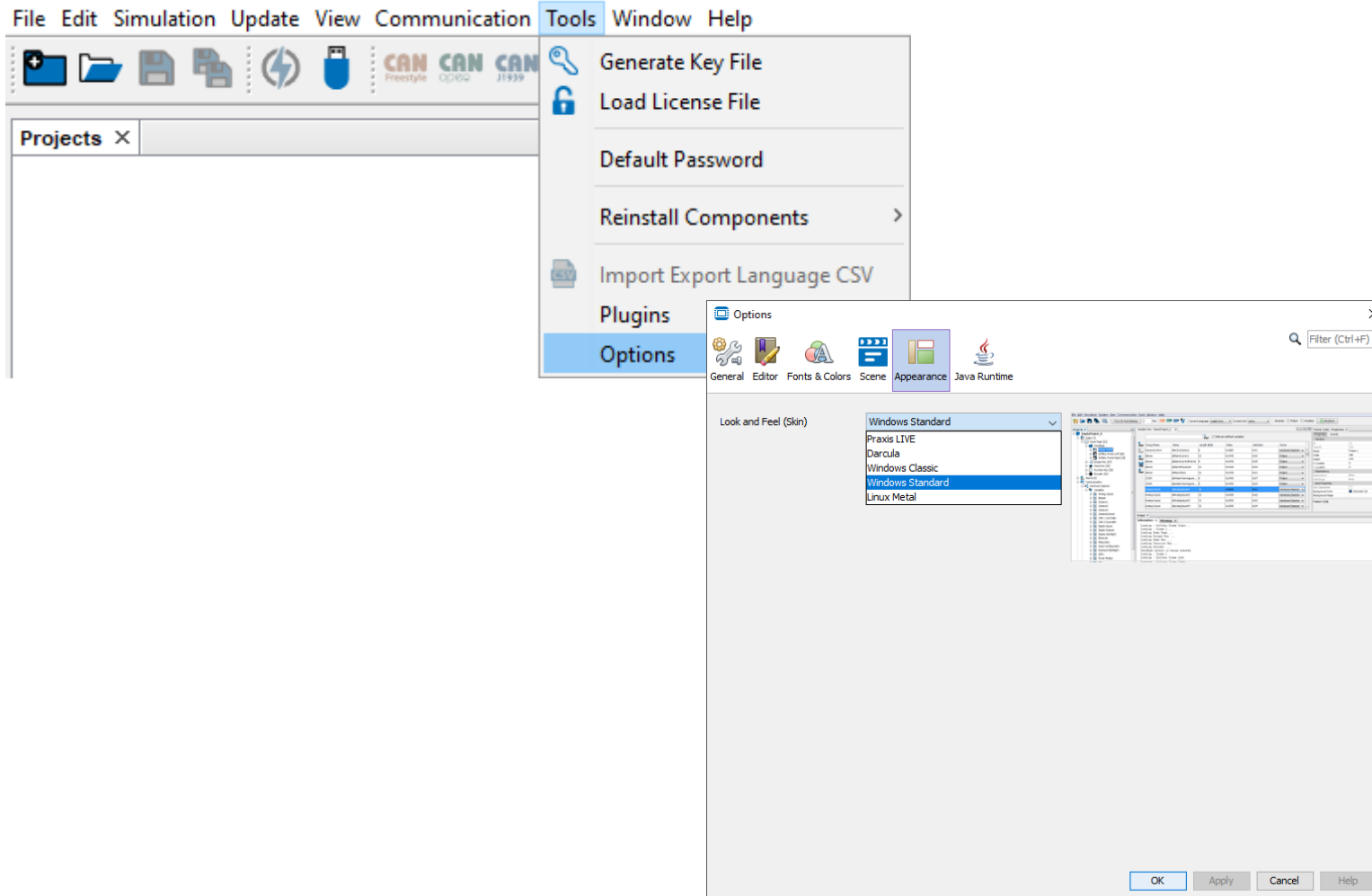
Var  Hide pre-defined variables

	Group Name	Name	Length (Bits)	Index	SubIndex	Owner
+	Alarms	@AlarmCurrent	16	0x2430	0x02	PClient
-	Alarms	@AlarmCurrentPriority	8	0x2430	0x05	PClient
+	Alarms	@AlarmEnqueued	16	0x2430	0x04	PClient
+	Alarms	@AlarmShow	16	0x2430	0x01	PClient
+	Alarms	@IsAlarmEnqueued	16	0x2430	0x03	PClient

New variable  
 Find variable usages

Variables

### 3. Getting to know the Projektor Tool – Changing the skin



If you don't like the dark look, go to  
Tools -> Options  
and in the dialog to the tab Appearance

Choose a skin that you like

# Agenda

1. Introduction
2. Installation of the Projektor Tool
3. Getting to know the Projektor Tool
4. Updating your device
5. First project
6. CAN communication
7. Extended agenda
8. FAQ



## 4. Updating your device

All elements of the Toolchain should be on the same level, as they are tested and released by us:

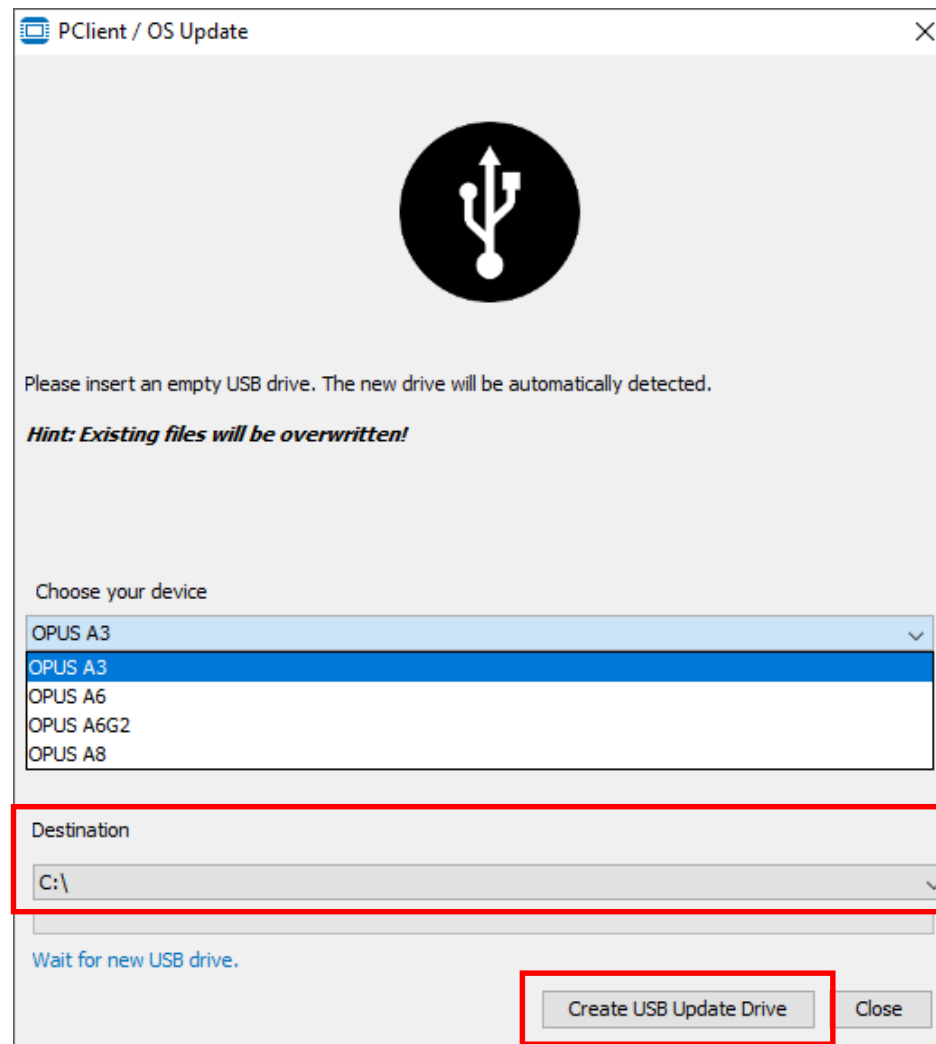
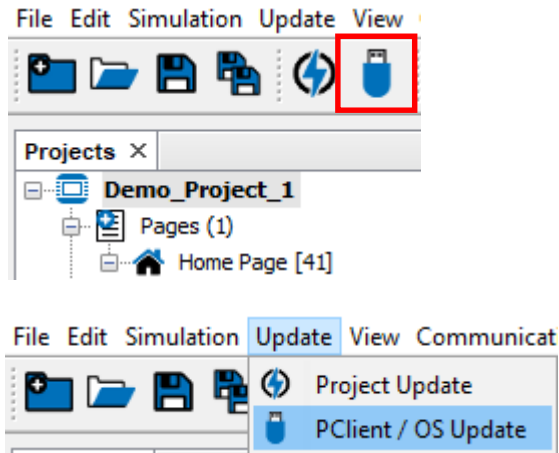
**Projektor Tool** on your PC

**OS** on the device

**PCClient** on the device

**-> When you purchase new devices, always perform a software update first!**

## 4. Updating your device – Creating the USB stick



In the Projektor Tool, click the PClient / OS Update Button or go to the menu *Update -> PClient / OS Update*

Insert a USB stick into your PC  
-> FAT 32 formatted!

In the dialog, choose the device you are working with

Leave *PClient* and *Operating System* checked

Select the correct drive and click *Create USB Update Drive*

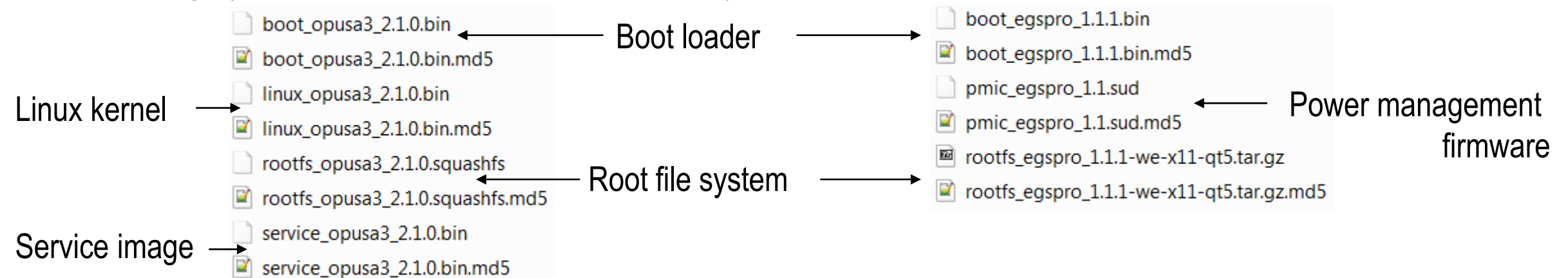
Let's take a look at the created files 18

## 4. Updating your device – The update files

The update packet for your device consists of:

### A3 / A6 G1

**OS – Operating System.** A customized embedded Linux system.

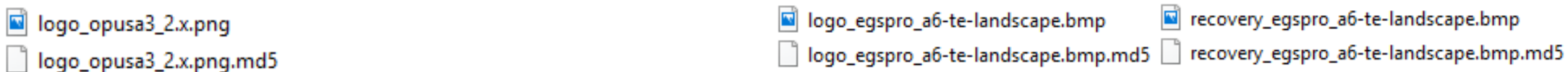


Service files to update from OS 1

**PClient – Projektor Client.** The application that displays your project on the device.



**Boot logo –** The image that is displayed when the device is started -> created by the customer (see chapter 8)



## 4. Updating your device – The developer cable

What are all those clamps on the developer cable?!

A/DI 1 – 4 – Analog / Digital Inputs

DO 1 – 3 – Digital Outputs

RS232 – serial console of the Linux system

CAN1/2 – CAN ports

USB – USB

Clamp 30 – Battery plus

Clamp 15 – Ignition (Should be connected into Clamp 30)

GND / Car GND – Ground (Should be connected if separate clamps)

Serv\_EN – Service Enable

cable color

brown,green,yellow,grey

white,lilac,blue

RxD-red-blue,TxD-red,GND-pink

CAN1-pink(high),blue(low)

CAN2-green(high),yellow(low)

VCC-red,GND-black,Low-white,High-green

red

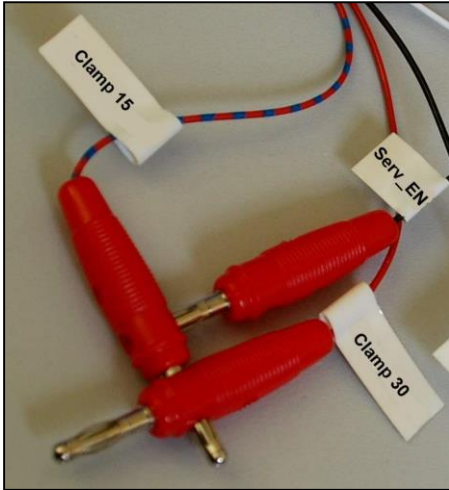
red-blue

GND-black,CarGND-white, black connector(s)

black

## 4. Updating your device – Preparations

3 a)

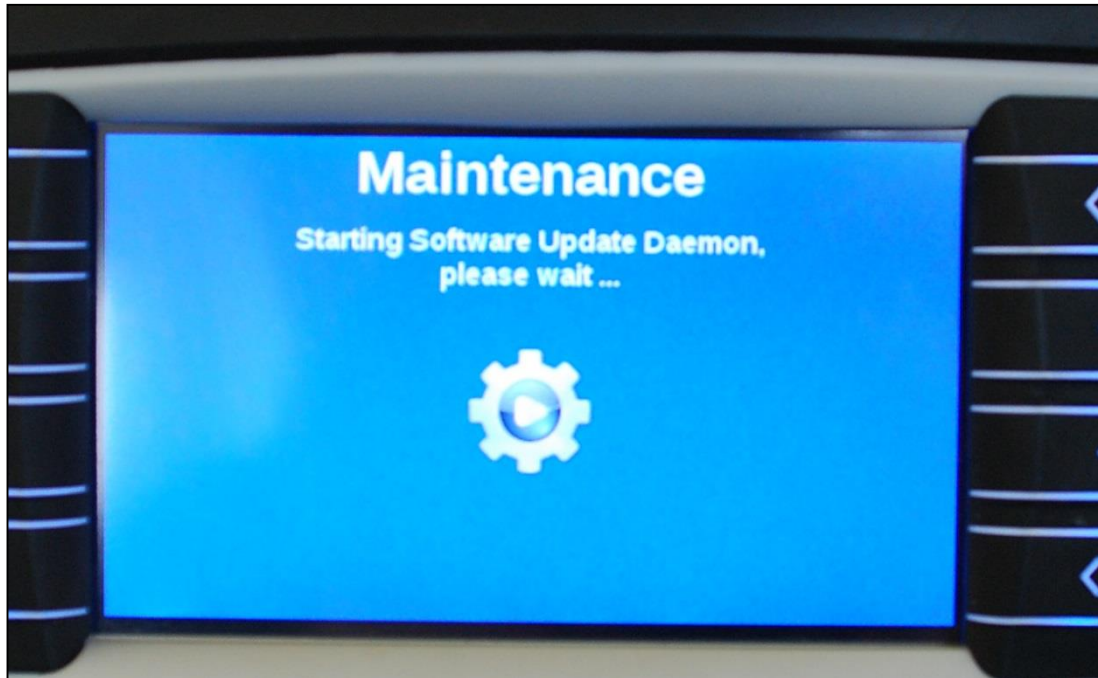


3 b)



1. Power off the device
2. Connect the USB stick containing all update files with the device
3. Power up the device in service mode by
  - a) Connecting clamp service enable  
or
  - b) Pressing the left keys 1 + 2 during power up (hold for 3 seconds)  
(for all devices but the A8)  
Key 1 + Stop key (A8)

## 4. Updating your device – Almost done



When you see this screen (or similar), let go of the keys or disconnect the clamp service enable

The update takes 3-5 minutes after which the device will restart and, after the touch screen calibration, show the welcome project.

Please never disconnect the power from the device during the update process!

Always wait until the device has restarted!

# Agenda

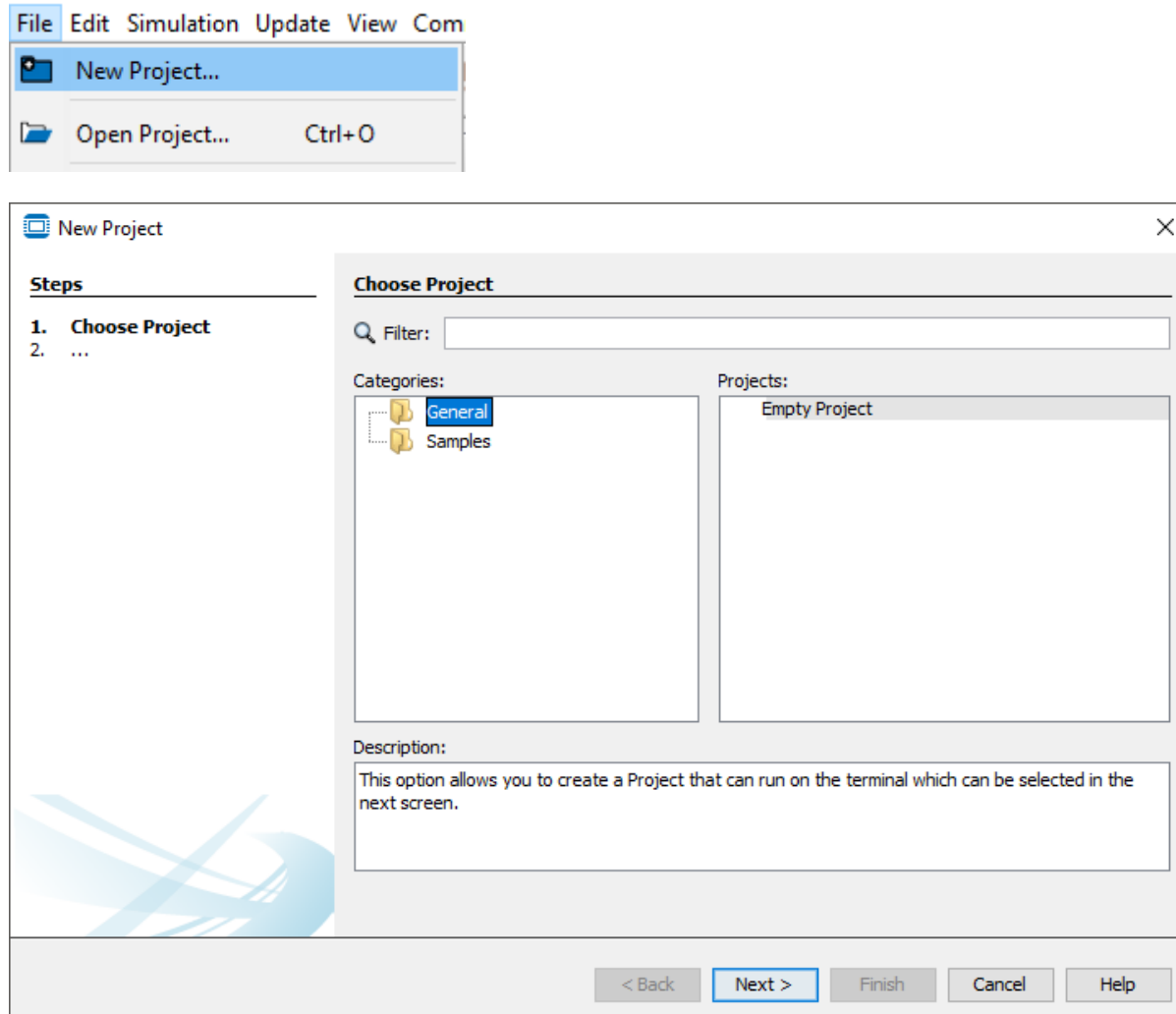
1. Introduction
2. Installation of the Projektor Tool
3. Getting to know the Projektor Tool
4. Updating your device
- 5. First project**
6. CAN communication
7. Extended agenda
8. FAQ

## 5. The first project

1. **Create a new project**
2. The first page
3. Meter object
4. Download the project
5. Linear Bar graph
6. Numeric field
7. The second page – Navigation
8. Buttons
9. List objects
10. 2D graph



## 5.1 The first project – Create a new project



To create a new project, click the  button in the toolbar or use the menu *File -> New Project...*

In the first step of the dialog, you can choose between an empty project or some sample projects

Choose Empty Project and click Next\*

\*be sure to check out the sample projects when you're back home

## 5.1 The first project – Create a new project

**New Project**

**Steps**

1. Choose Project
2. **Name and Location**

**Name and Location**

Name: Demo\_project\_1

Location: C:\Users\cst\Documents\projektor\_projects

Folder: C:\Users\cst\Documents\projektor\_projects\Demo\_project\_1

Unit: metric

Panel: OPUS A3 Standard

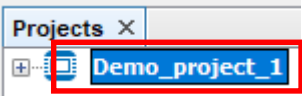
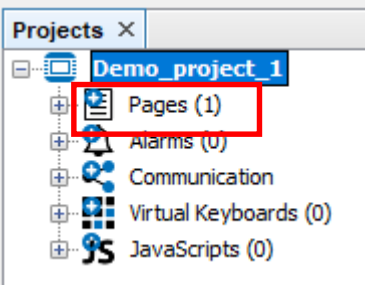
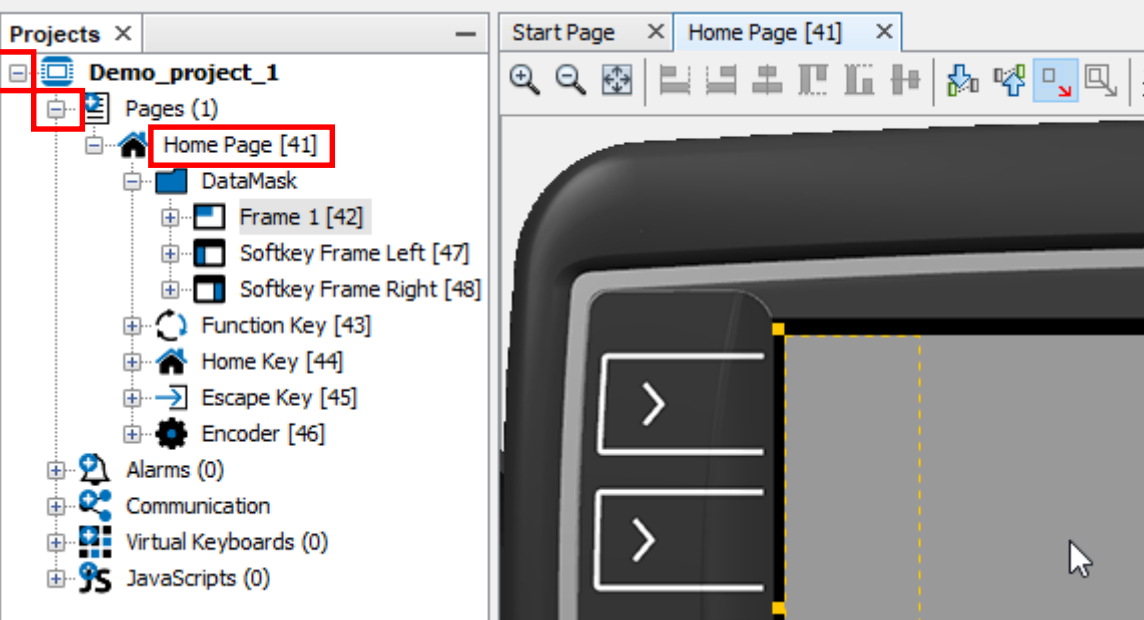
Use ISOBUS Application

Choose a project name and location and choose your device

Click finish to create the project.

All other settings can be changed later.

## 5.2 The first project – Project structure

- 1. In the Project Tree, the project name 'Demo\_project\_1' is highlighted with a red box.
- 2. In the Project Tree, the 'Pages (1)' entry under 'Demo\_project\_1' is highlighted with a red box.
- 3. In the Project Tree, the 'Home Page [41]' entry is highlighted with a red box. The editor window shows a virtual device interface with a gray frame.

In the Project Tree on the left, open the project by double-clicking the project name

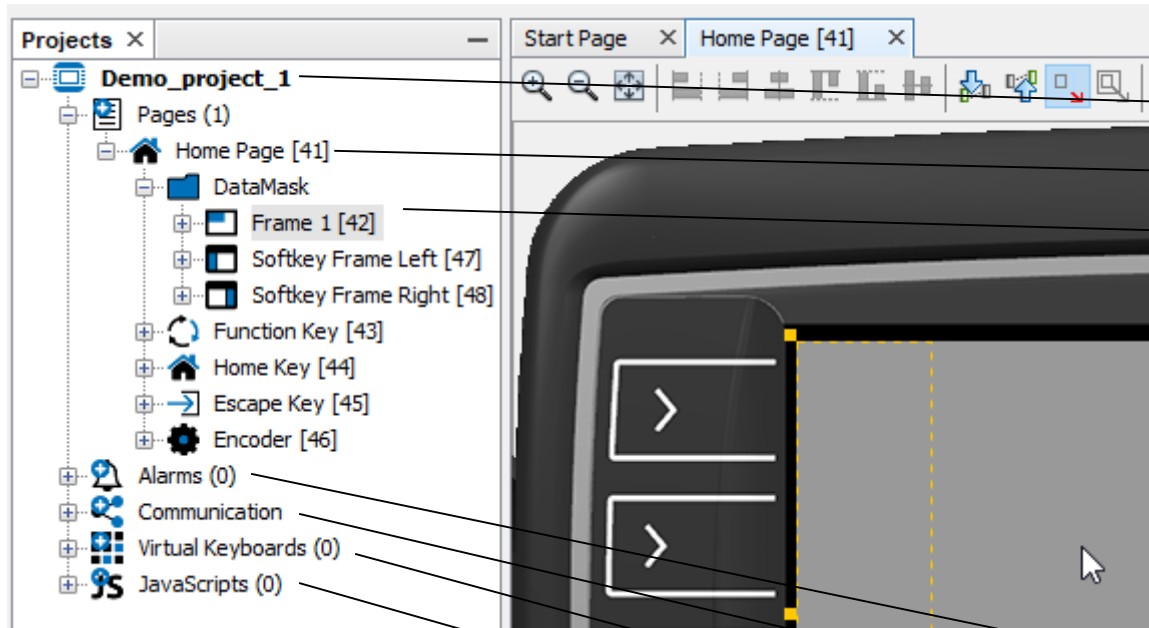
Then open the Pages by double-clicking the Pages (1) entry

Now double click the Home Page to view it in the editor window

Click on the gray frame in the virtual device



## 5.2 The first project – The first page



We now have:

a project

with one page which is called Home Page

In the Home Page there is a Frame

The frame is the primal object on every page –  
all other objects can only be inside of frames!

The project also has

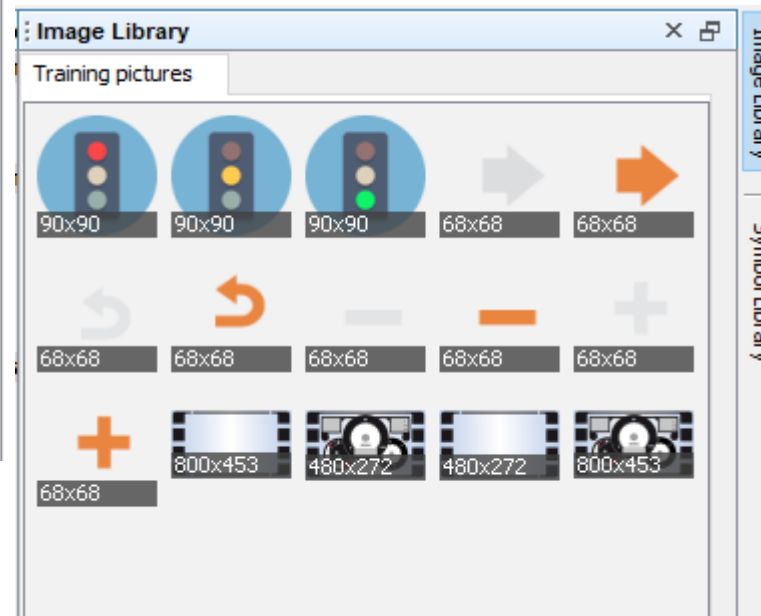
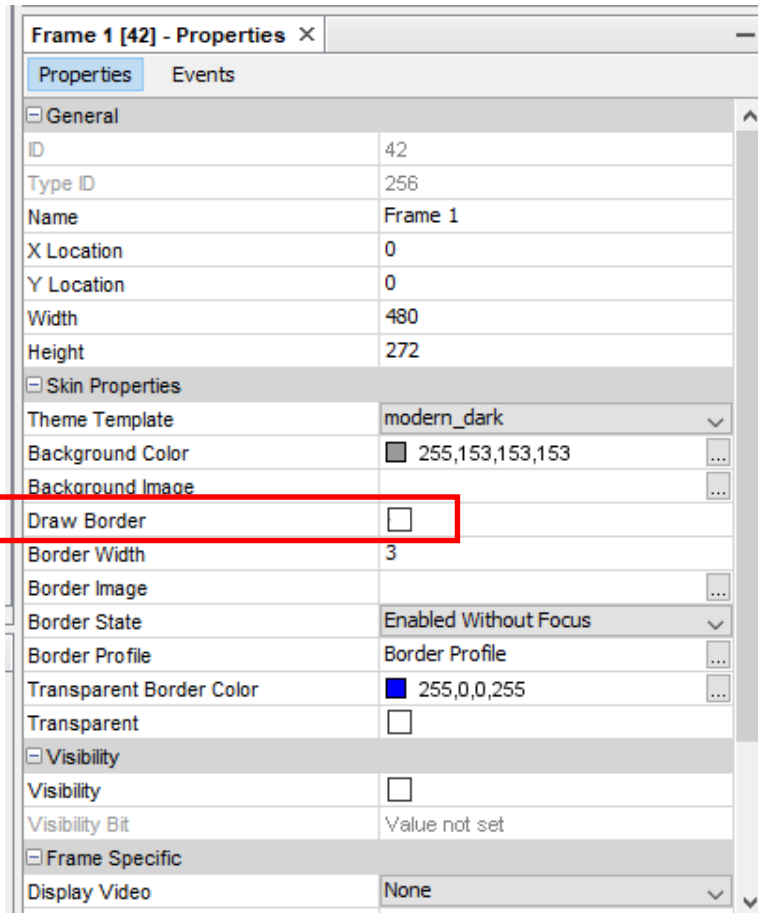
Alarms category

Communication (for CAN Mappings)

Virtual Keyboards category

JavaScripts category

## 5.2 The first project – The first page – Frame object



Click in the center of the gray area to select the frame

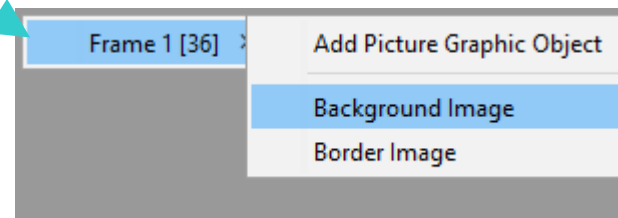
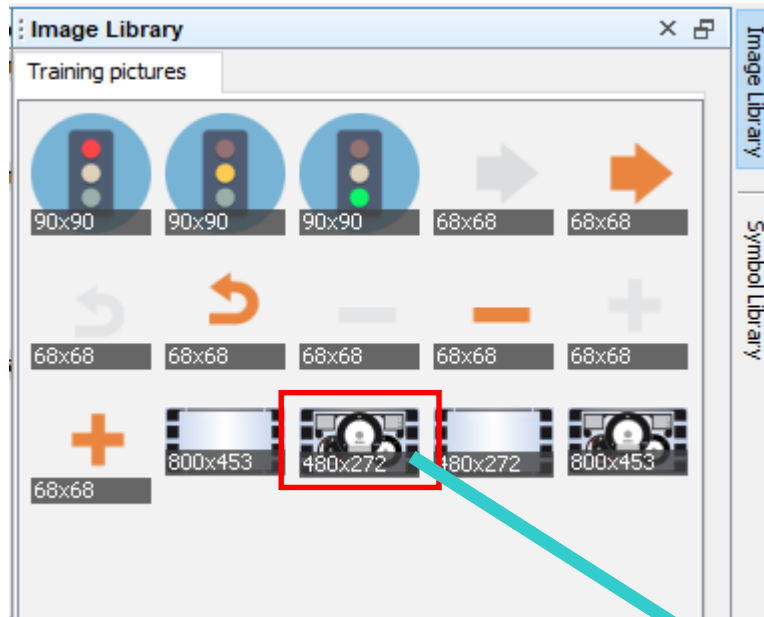
Now the properties of the frame are shown

Un-Check the property *Draw Border*

Open the Image Library tab

Load the Training pictures folder  
(inside the training material folder you copied from the USB stick)

## 5.2 The first project – The first page – Frame object



Drag & Drop the marked image on the frame (the gray area)

In the context menu, select the Frame and then Background Image

## 5.2 The first project – The first page – Frame object

Name	Frame 1
X Location	0
Y Location	0
Width	480
Height	272
Skin Properties	
Theme Template	modern_dark
Background Color	255,153,153,153
Background Image	Training_Background_1_480_272.png
Draw Border	<input type="checkbox"/>
Border Width	3
Border Image	
Border State	Enabled Without Focus
Border Profile	Border Profile
Transparent Border Co	255,0,0,255

The result should look like this.

Background Image	Training_Background_1_480_272....
Draw Border	<input checked="" type="checkbox"/>
Border Width	3

Draw Border  
**Saved Value:**  
*true*

The modified properties will be marked blue. If you select a modified property, the last saved (or default) value will be shown in the description area



## 5.2 The first project – The first page – Information about images

Most image formats are supported: JPG, BMP, GIF, TIFF, PNG, SVG

Internally all bitmap images are converted to PNG in the Projektor, so use PNG if you create graphics

Animated GIF files are also possible -> See example project





## 5.3 The first project – Meter object

Goals:

Create a meter object and configure it

Reference a variable to the meter

Make the meter editable

## 5.3 The first project – Meter object – Creation and variable reference

The screenshot shows the OPUS software interface. The main workspace displays a dashboard with several gauges. A context menu is open over one of the gauges, with 'Meter' and 'Variable Reference' highlighted. A search bar in the 'Variable View' panel shows 'disp', and a table below lists variables, with '@DispBacklightIntensity' highlighted.

Group Name	Name	Length (Bits)	Index	SubIndex	Owner
Display Backlight	@DispBacklightIntensity	8	0x20B0	0x01	Hardwar...
Display Backlight	@DispBacklightIntensity0	8	0x20B0	0x05	Hardwar...

Meter objects are the classic instrument to visualize values graphically

Useful for speed, RPM, fuel levels

Can be used full circle or just a part

Drag & drop a meter object into the frame

Resize it so it fits into the inner white circle

Search for the variable `@DispBacklightIntensity` and drag & drop it on the meter. In the context menu, choose the Meter object and then *Variable Reference*

## 5.3 The first project – Meter object – Properties

Start here ▶

Skin Properties	
Theme Template	modern_bright
Background Color	255,255,255,255
Background Image	
Draw Border	<input checked="" type="checkbox"/>
Border Width	3
Border Image	
Border State	Enabled Without Focus
Border Profile	Border Profile
Transparent Border Color	255,0,0,255
Transparent	<input checked="" type="checkbox"/>
Visibility	
Visibility	<input type="checkbox"/>
Visibility Bit	Value not set
Input Configuration	
Set As Input	<input checked="" type="checkbox"/>
Connect To Encode	<input checked="" type="checkbox"/>
Send Value Directly	<input checked="" type="checkbox"/>
Encoder Movement	Linear
Value Change Factor	1000

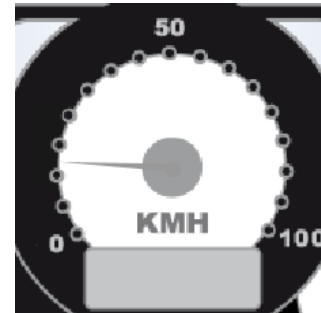
Continue here →

Value Related	
Absolute Max Value	100
Color Above Maximum	255,153,153,153
Max Value	80
Min Value	20
Absolute Min Value	0
Color Below Minimum	255,153,153,153
Preview Value	15
Activate Low Pass Filter	<input type="checkbox"/>
Filter Factor	500
Meter Specific	
Start Angle	325
End Angle	215
Deflection Direction	ClockWise
Needle Color	255,153,153,153
Needle Image	
Draw Arc	<input type="checkbox"/>
Arc Color	255,204,204,204
Draw Ticks	<input type="checkbox"/>
Tick Color	255,102,102,102
Number Of Ticks	12

Make the marked settings in the properties of the meter object.

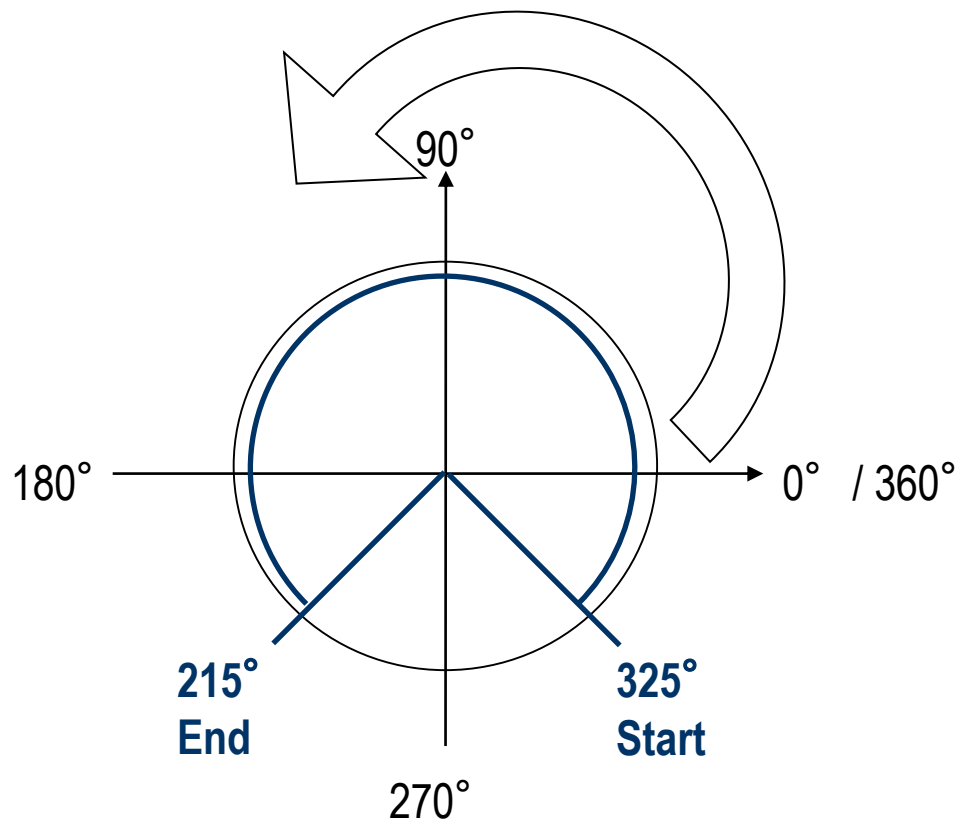
A warning dialog will appear for this property. Press OK, it will be explained later.

Result:





## 5.3 The first project – Meter object – Start and end angle



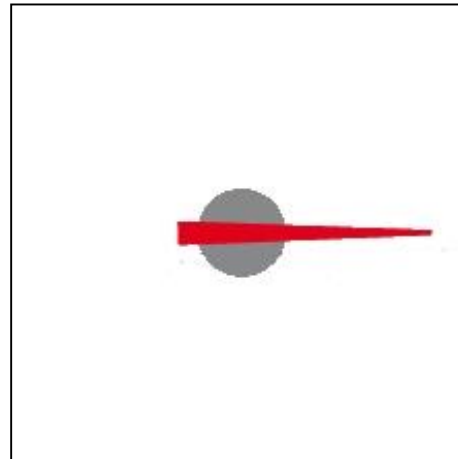
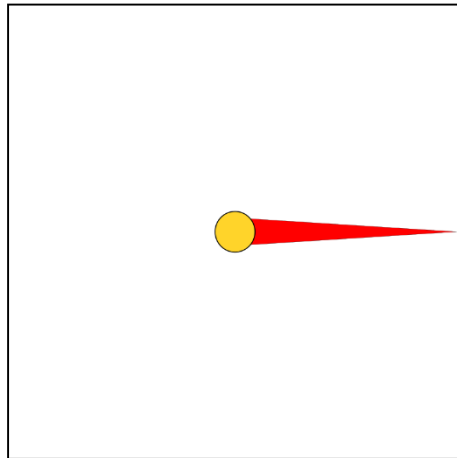
The spanning direction for the meter area is always counterclockwise

This has nothing to do with which way the needle will move!

Start Angle	325
End Angle	215



## 5.3 The first project – Meter object – Needle image



Possible to use images as needles instead of the standard needle

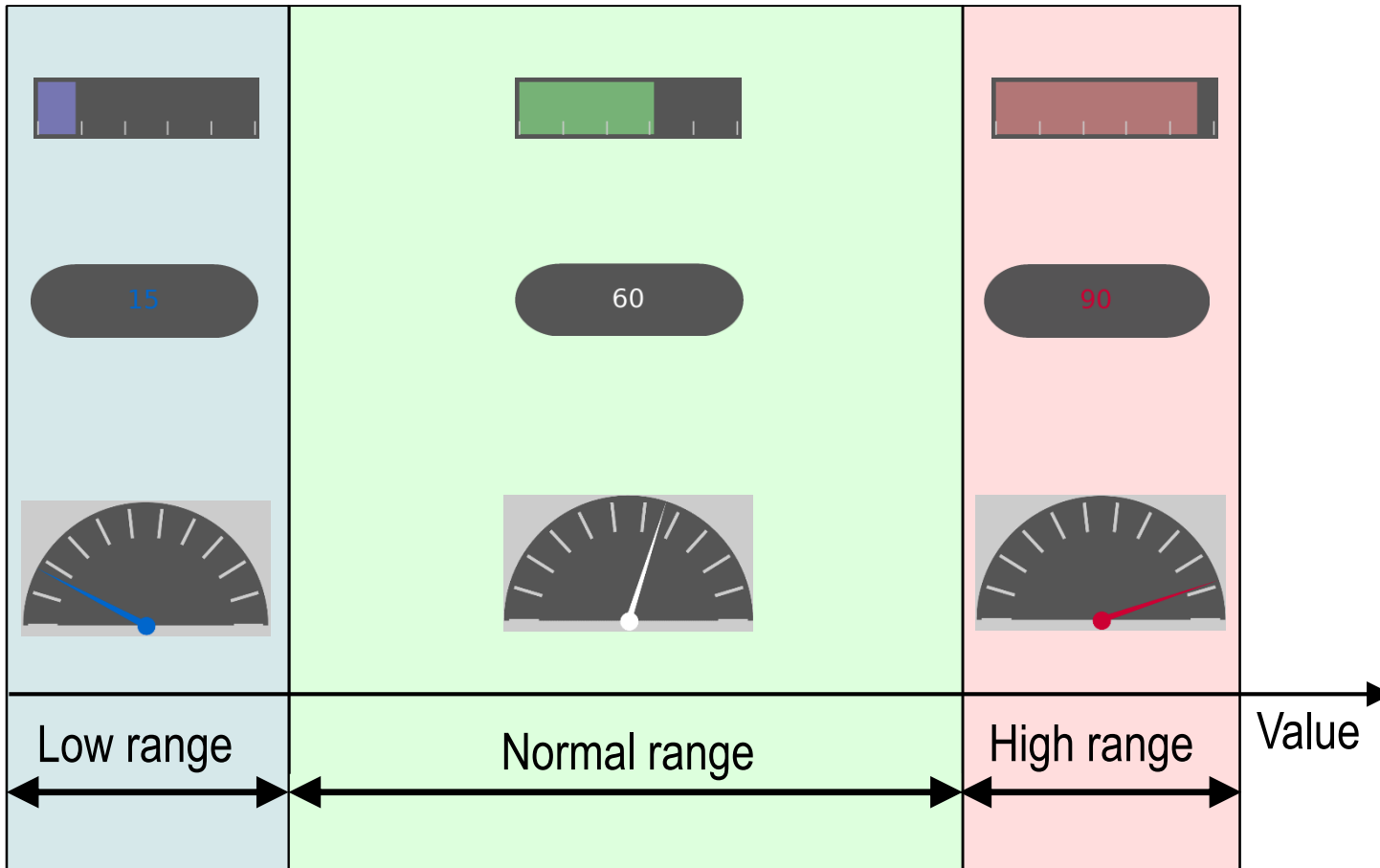
Image has to be squared

Needle has to point to the right

Center of the image is the turning point



### 5.3 The first project – Meter object – Minimum and maximum



Three value regions

Configurable color changes

Absolute limitations are for user entries

Value changes over CAN can exceed the limits

Absolute minimum

Minimum

Maximum

Absolute maximum

## 5. The first project

1. Create a new project
2. The first page
3. Meter object
4. **Download the project**
5. Linear Bar graph
6. Numeric field
7. The second page – Navigation
8. Buttons
9. List objects
10. 2D graph

## 5.4 The first project – Download the project - @EnableUpdater

@EnableUpdater value	Update over USB	Update over CAN	Update over Ethernet
0	✗	✗	✗
1 (default)	✓	✗	✗
2	✗	✓	✗
3	✓	✓	✗
4	✗	✗	✓
5	✓	✗	✓
6	✗	✓	✓
7	✓	✓	✓

The variable @EnableUpdater controls if the update over USB, CAN and Ethernet are allowed

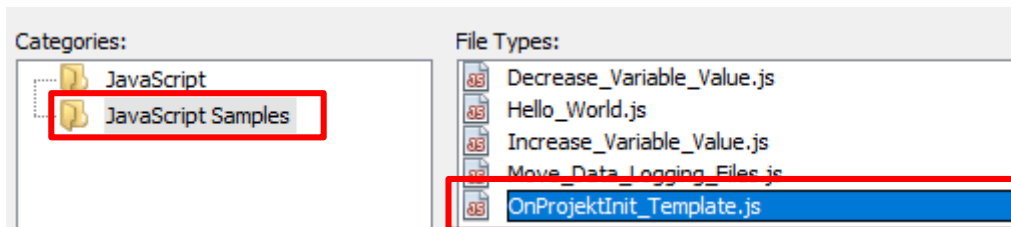
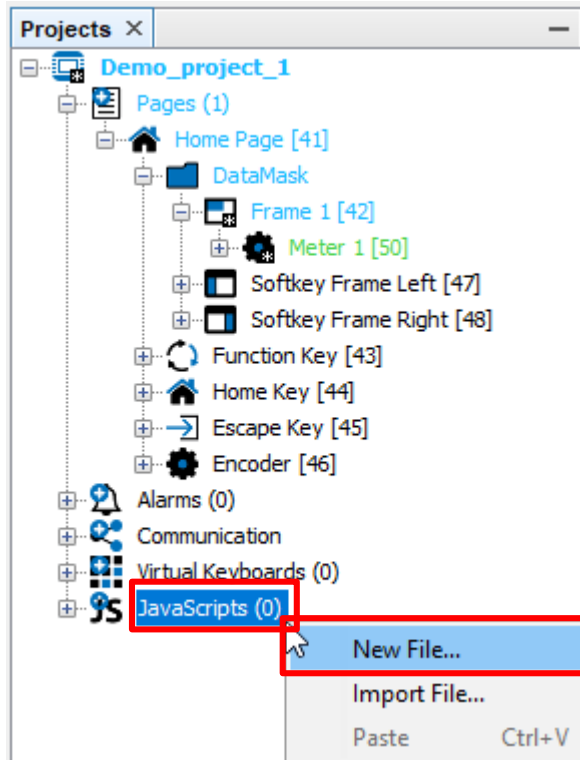
Bit 0: USB

Bit 1: CAN

Bit 2: Ethernet/LAN



## 5.4 The first project – Download the project - @EnableUpdater



We will create a script that is executed when the device starts

Right-click on JavaScripts and click New File...

Choose JavaScript Samples and select *OnProjektInit\_Template*

Click *Finish* to create the script file

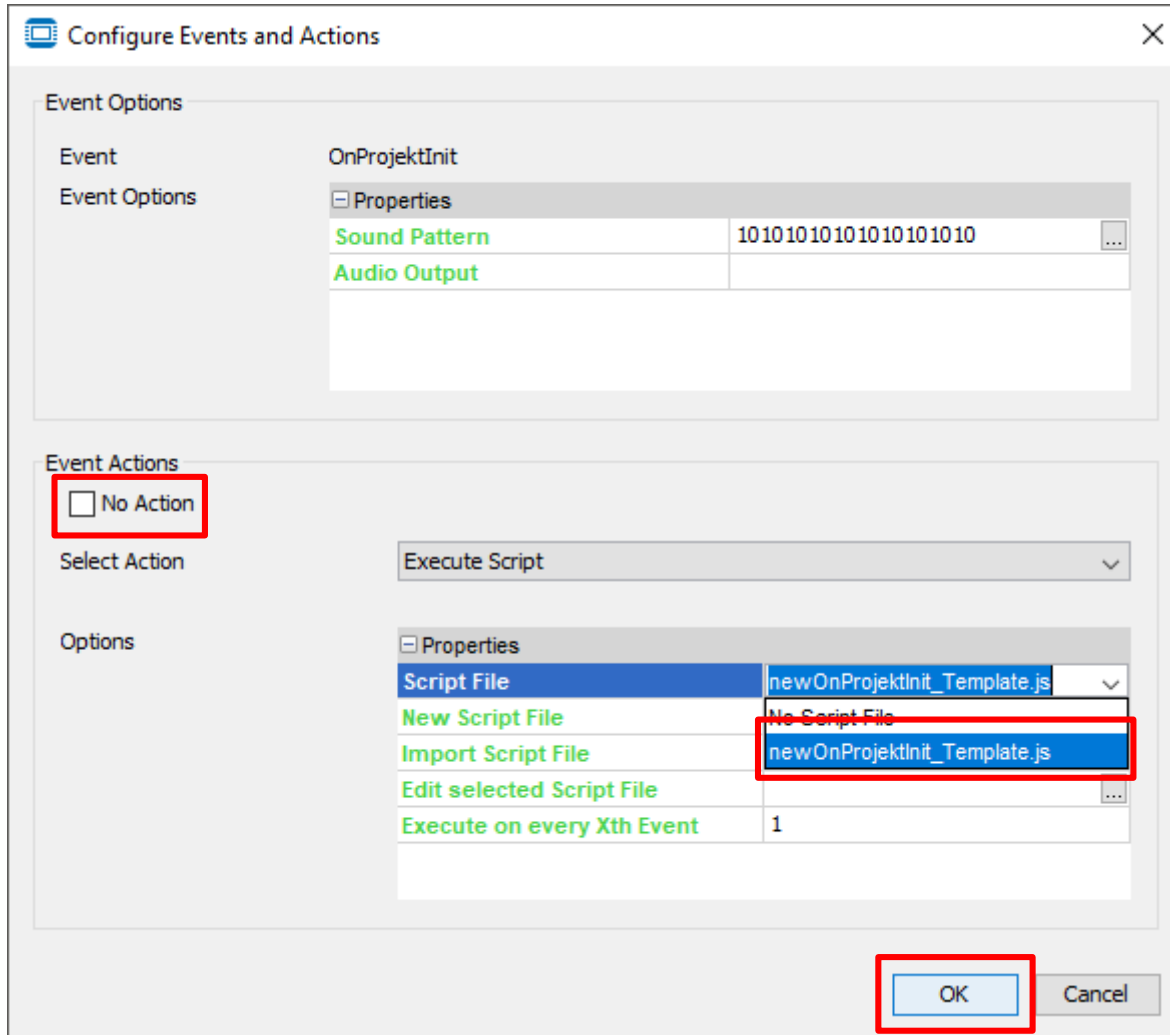
## 5.4 The first project – Download the project - @EnableUpdater

```
1 //Powermanagement settings
2 //setVariableValue("@PWR_TimeToSleepMode", 3600); //change time until sleep mode (in second
3 //setVariableValue("@PWR_TimeToPowerOff", 28800); //change time until power off (in seconds
4 ////Display settings
5 //setVariableValue("@DispBacklightStartupIntensity", 100); // set the screen to maximum bri
6 //Updater Settings
7 setVariableValue("@EnableUpdater", 7); // enable project update over USB, CAN and Ethernet
```

Demo_project 1 - Properties	
Properties	Events
Events	
OnProjektInit	No Action
OnProjektExit	No Action
OnProjektRepeat	No Action
OnPowerOn	No Action
OnLowPower	No Action
OnSleep	No Action

Click on the project name in the project tree and click the  button next to OnProjektInit in the Events tab

## 5.4 The first project – Download the project - @EnableUpdater



In the dialog, uncheck No Action and choose the script file

Click OK to close the dialog

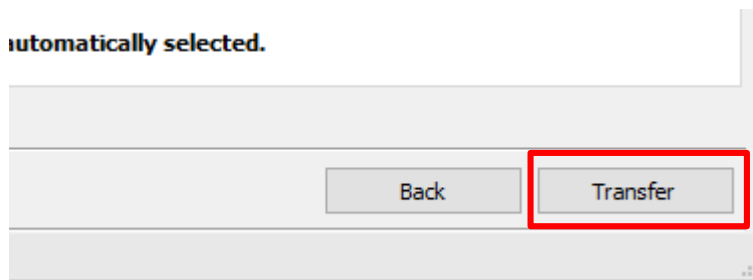
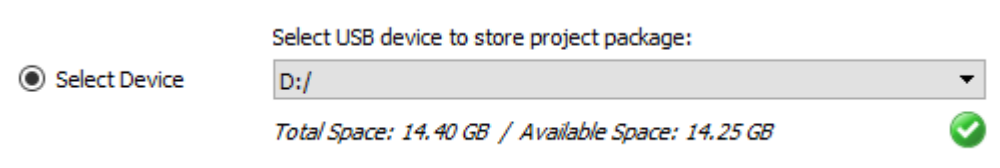
## 5.4 The first project – Download the project – Save!




Before downloading a project, ALWAYS save it, otherwise you will only get the last saved state of the project

If elements of your project are colored blue or green, they are modified or new, so you need to save before downloading.

## 5.4 The first project – Download the project - USB



To download the project, connect a USB stick (FAT 32 formatted) with your PC

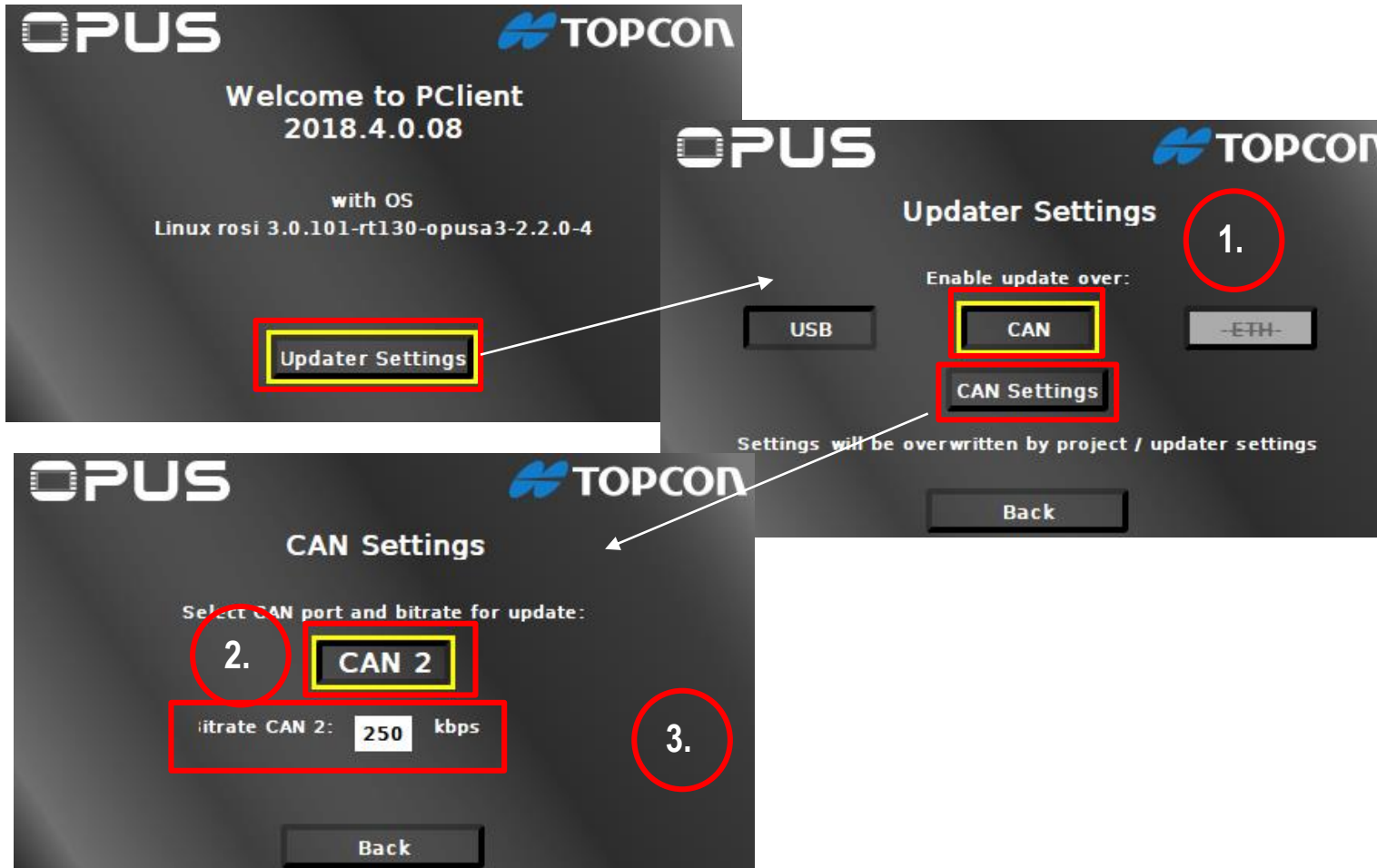
Press the Project Update button  or go to the menu *Update -> Project Update*  
Select USB in the dialog  
Select the drive of the stick to download the project.

Click Transfer to install the project

When the download has finished, put the USB stick into the operating device and the project will be loaded automatically.



## 5.4 The first project – The first page – Download the project - CAN



To download the project via CAN, some settings have to be made:

1. Setting the variable @EnableUpdater
2. The CAN port that should be used
3. The baud rate

For your first transfer, you can make these settings in the welcome project on the device

But the settings still need to be made in the Projektor, or the settings made above will be reverted after the project download!

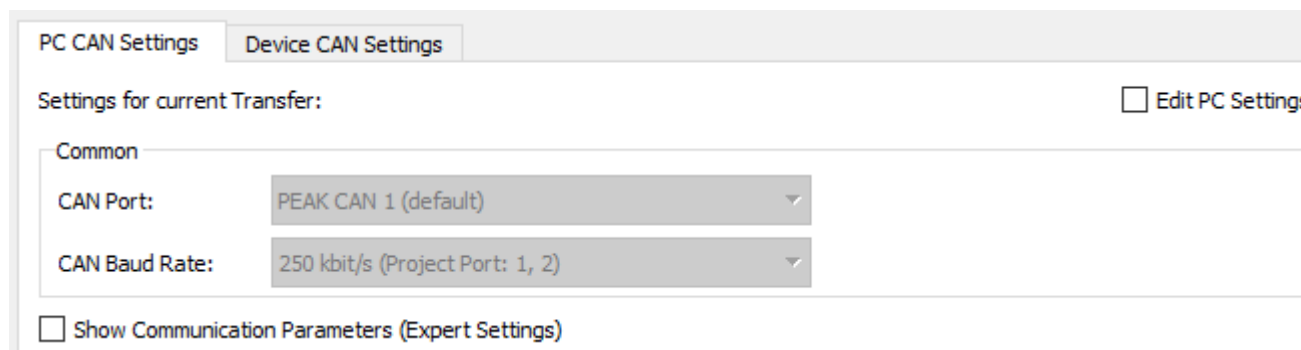
## 5.4 The first project – The first page – Download the project - CAN



Press the Project Update button 



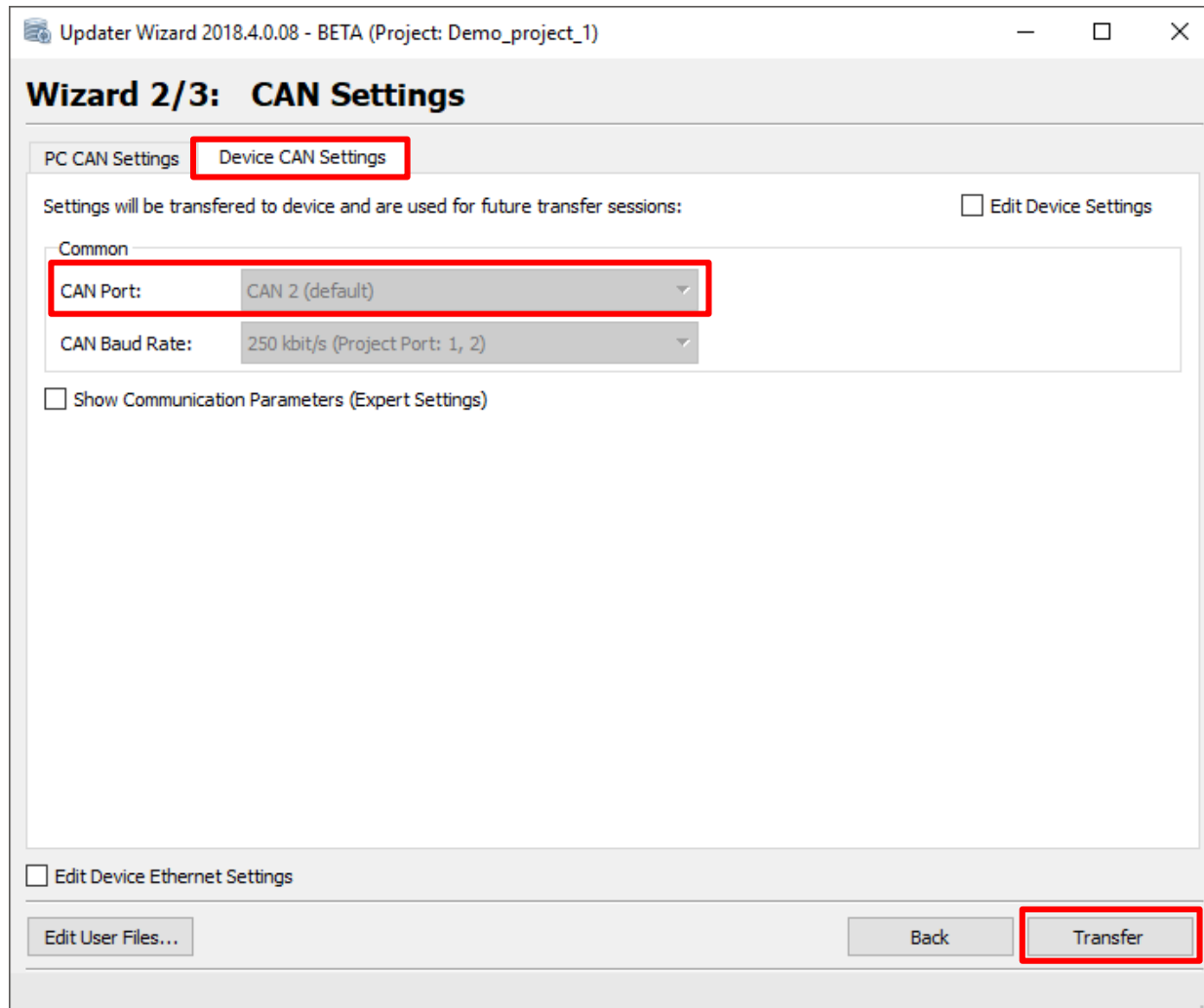
Select CAN in the dialog



Agree to the warning

The PC settings normally don't have to be changed

## 5.4 The first project – The first page – Download the project - CAN



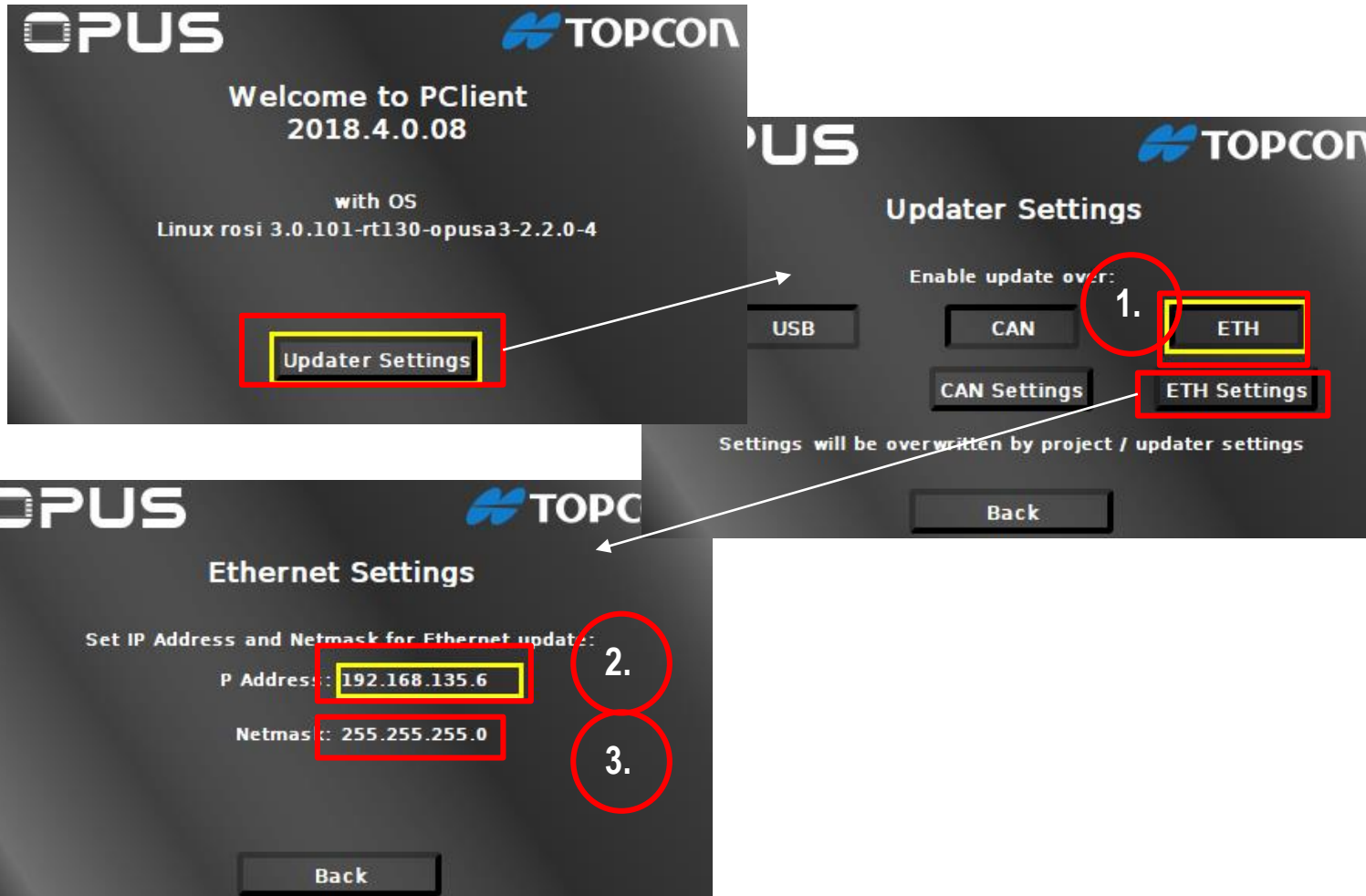
In the Device CAN Settings, check Edit Device Settings and set the CAN Port that should be used for the next update (should be the same as the setting in the welcome project)

The baud rate is set according to the project setting in Communication -> Port Configuration...

Click Transfer to install the project



## 5.4 The first project – The first page – Download the project - Ethernet



To download the project via Ethernet, some settings have to be made:

1. Setting the variable @EnableUpdater
2. The IP address that should be used
3. The netmask address that should be used

For your first transfer, you can make these settings in the welcome project on the device

But the settings still need to be made on the PC, or the settings made above will be reverted!

## 5.4 The first project – The first page – Download the project - Ethernet

The screenshot shows the software interface with the 'Update' menu open. The 'Project Update' option is highlighted. Below the menu, there are three buttons: a USB icon, 'CAN', and a network icon. The network icon is highlighted. Below this, the 'PC Ethernet Configuration' dialog is shown with the 'Device Ethernet Settings' tab selected. The 'Scan Network...' button is highlighted. Below the dialog, the 'New Device Settings' section is visible with the following fields:

Project Supported	Device Type	IP-Address	Command Port	Available Space (MB)

New Device Settings

These settings will be available after transfer!  Keep existing Device Ethernet Settings

IP-Address: 192.168.135.6

Command Port: 5000

Stream Port: 5001

Broadcast Port: 5010

Press the Project Update button 

Select Ethernet in the dialog

Use Scan Network... to look for the connected device

In the Device Ethernet Settings tab, set the IP address that the device should have.

Click Transfer to install the project

## 5. The first project

1. Create a new project
2. The first page
3. Meter object
4. Download the project
5. **Linear Bar graph**
6. Numeric field
7. The second page – Navigation
8. Buttons
9. List objects
10. 2D graph

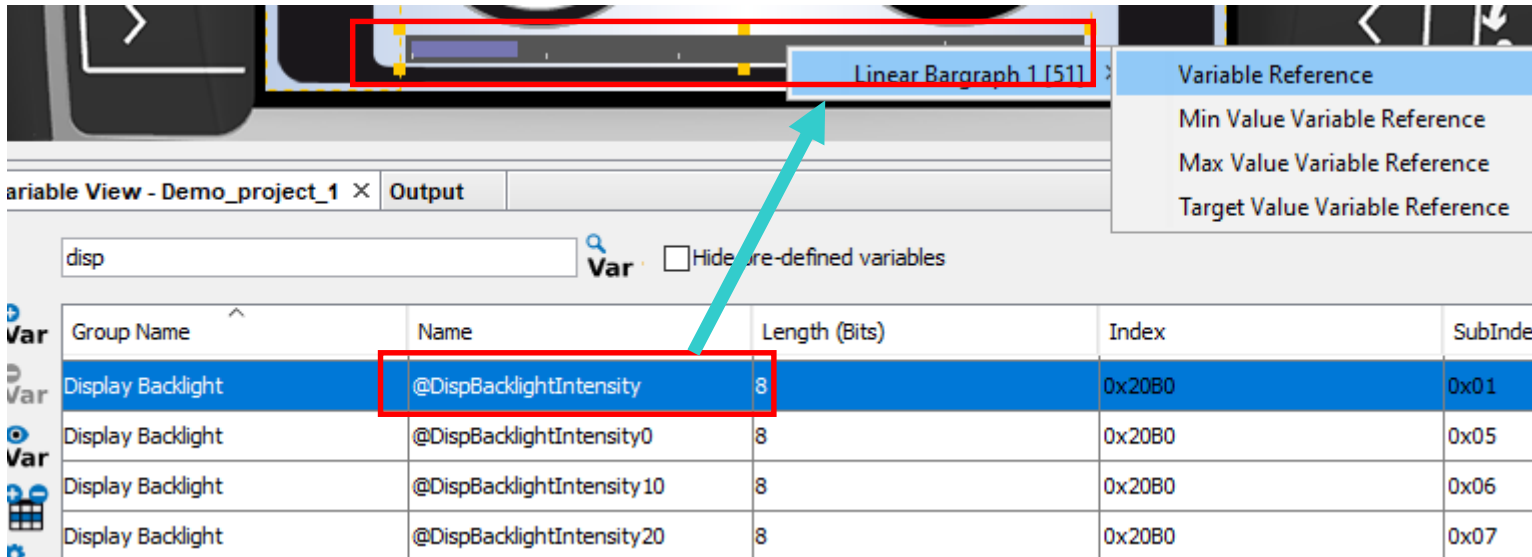
## 5.5 The first project – Linear Bar graph

Goals:

Create a linear bar graph object

Reference a variable to it

## 5.5 The first project – Linear Bar graph



Variable View - Demo\_project\_1 × Output

disp  Hide pre-defined variables

Group Name	Name	Length (Bits)	Index	SubIndex
Display Backlight	@DispBacklightIntensity	8	0x20B0	0x01
Display Backlight	@DispBacklightIntensity0	8	0x20B0	0x05
Display Backlight	@DispBacklightIntensity10	8	0x20B0	0x06
Display Backlight	@DispBacklightIntensity20	8	0x20B0	0x07

Drag & Drop a Linear Bargraph object into the frame

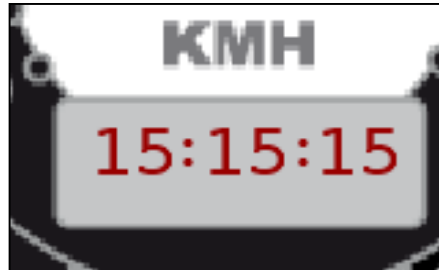
Modify size and position as seen in the picture

Reference the Bargraph with the variable *@DispBacklightIntensity*

Save and download the project to the device and test it



## 5.6 The first project – Numeric field

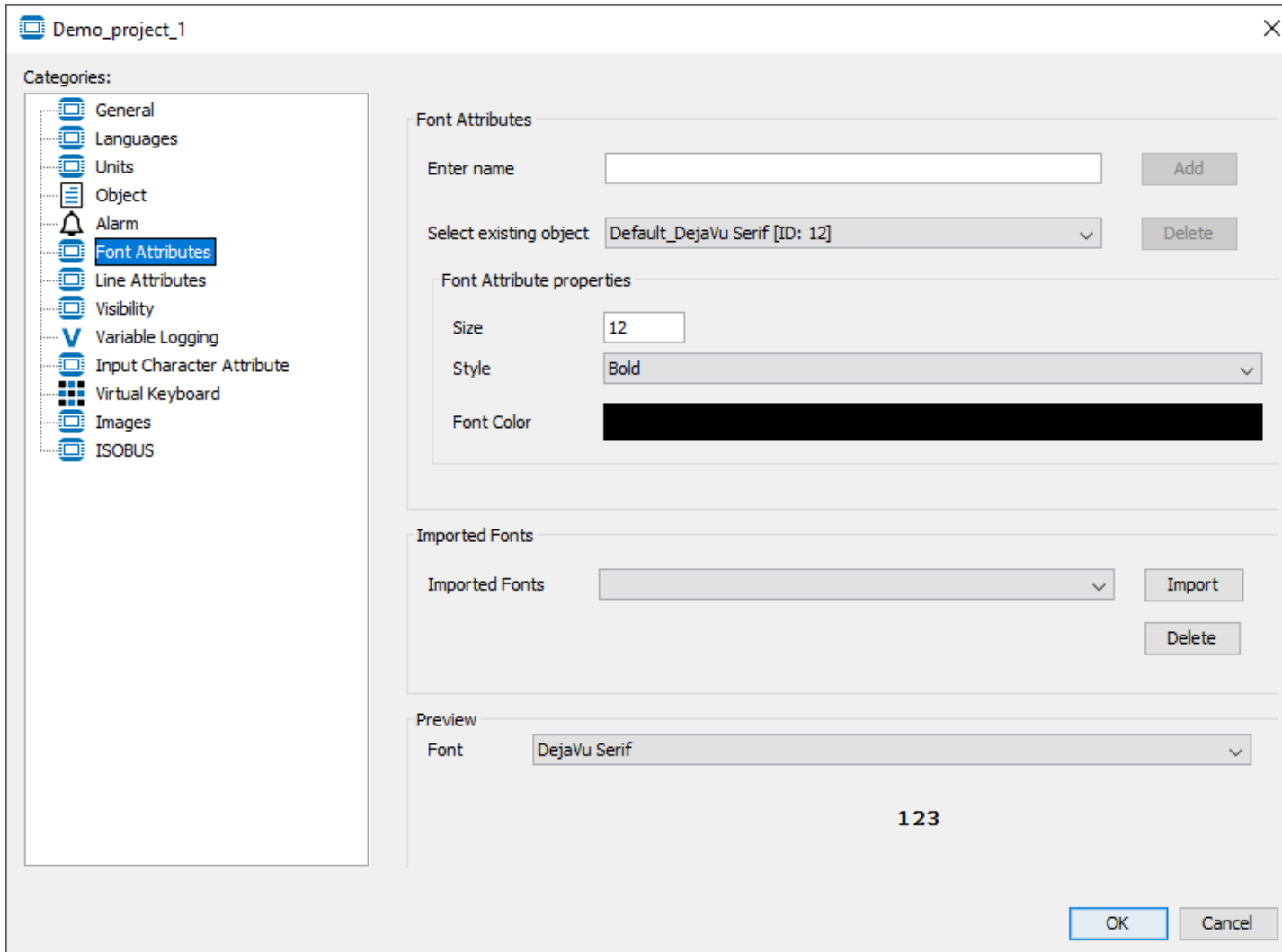


Goals:

Create a new font attribute

Create a clock with 3 numeric fields

## 5.6 The first project – Numeric field – A new font attribute



Formatting texts is done with font attributes

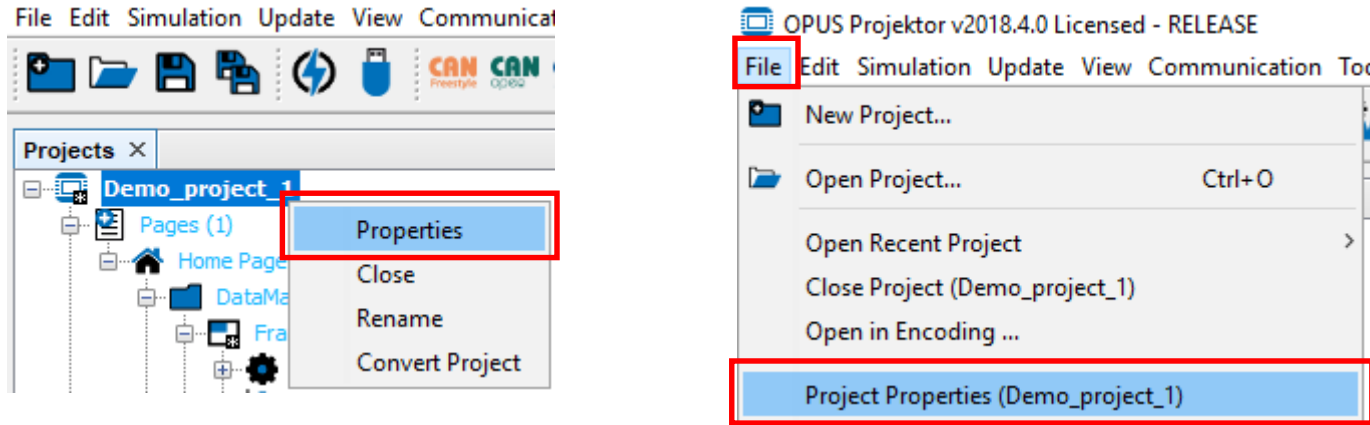
A font attribute consists of

- size
- color
- style (bold and / or italic)

Advantage: changes throughout the project with one click

Font files can be imported (think about copyrights)

## 5.6 The first project – Numeric field – A new font attribute



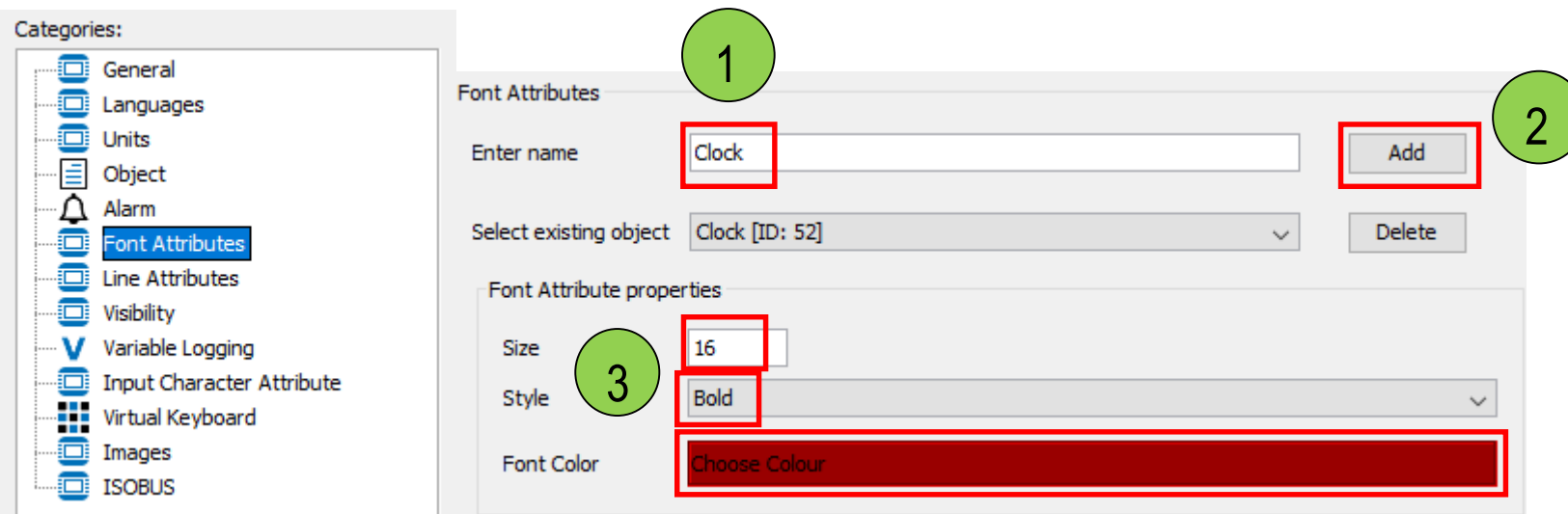
Open the project properties by right-clicking the project name and selecting “Properties”

or by using the menu  
File -> Project Properties

Choose the category “Font Attributes”

Enter a name for a new font attribute (1) and click “Add” (2)

Change  
Size to 16  
Style to Bold  
Font Color to a dark red (3)





## 5.6 The first project – Numeric field – Creation and variable reference

Search for RTC variables

Group Name	Name	Length (Bits)	Index	SubIndex	Owner
RTC	@RTC_AMP	8	0x2100	0x09	Hardware ...
RTC	@RTC_Day	8	0x2100	0x03	Hardware ...
RTC	@RTC_Hours	8	0x2100	0x05	Hardware ...
RTC	@RTC_HoursAM...	8	0x2100	0x08	Hardware ...
RTC	@RTC_Minutes	8	0x2100	0x06	Hardware ...
RTC	@RTC_Month	8	0x2100	0x02	Hardware ...
RTC	@RTC_Seconds	8	0x2100	0x07	Hardware ...

Drag & drop a Numeric field in the page

Search for RTC variables

Reference the variable *@RTC\_Seconds*

## 5.6 The first project – Numeric field – Properties

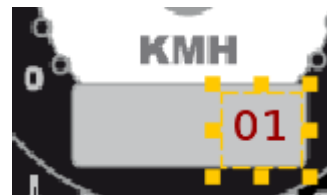
Properties	Events
Transparent	<input checked="" type="checkbox"/>
Corner Radius	50
[-] Visibility	
Visibility	<input type="checkbox"/>
Visibility Bit	Value not set
[-] Input Configuration	
Set As Input	<input type="checkbox"/>
Connect To Encoder	<input type="checkbox"/>
Send Value Directly	<input checked="" type="checkbox"/>
Encoder Movement	Linear
Value Change Factor	1000
Enabled	<input checked="" type="checkbox"/>
Inplace Editor	Encoder
[-] Value Related	
Absolute Max Value	100
Max Value	90
Font Attribute Above Max Value	Text White
Min Value	0
Font Attribute Below Min Value	Text white
Absolute Min Value	0
Preview Value	1
Activate Low Pass Filter	<input type="checkbox"/>

Activate Low Pass Filter	<input type="checkbox"/>
Filter Factor	500
Step Width	1
[-] Formatting	
Font Attribute	Clock
Font	DejaVu Sans
Horizontal Alignment	Center
Vertical Alignment	Center
[-] Numeric Field Specific	
Format	Decimal
Leading Zeros	<input checked="" type="checkbox"/>
Display Zeros As Blank	<input type="checkbox"/>
Offset	0
Scale	1.0
Offset 2	0
No Of Decimals	0

Make the marked settings in the properties of the numeric field.

Adjust the size so that only one zero is displayed before the 1.

Result:



## 5.6 The first project – Numeric field – Complete the clock

Variable View - Demo\_project\_1 x Output

rtc Var  Hide p

Var	Group Name	Name
Var	RTC	@RTC_AMPM
Var	RTC	@RTC_Day
Var	RTC	@RTC_Hours
Var	RTC	@RTC_HoursAMPM
Var	RTC	@RTC_Minutes

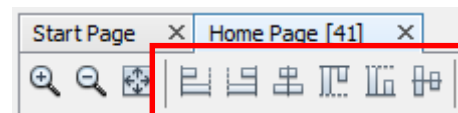
Use copy & paste to duplicate the configured numeric field

-> After copy, select the frame and then paste

Use the variables *@RTC\_Minutes* and *@RTC\_Hours* and reference them to the numeric fields

You can move objects with the cursor keys (1 pixel)

- with ALT pressed: 5 pixels
- with CTRL pressed: 10 pixels
- with SHIFT + ALT pressed: 50 pixels
- with SHIFT + CTRL pressed: 100 pixels

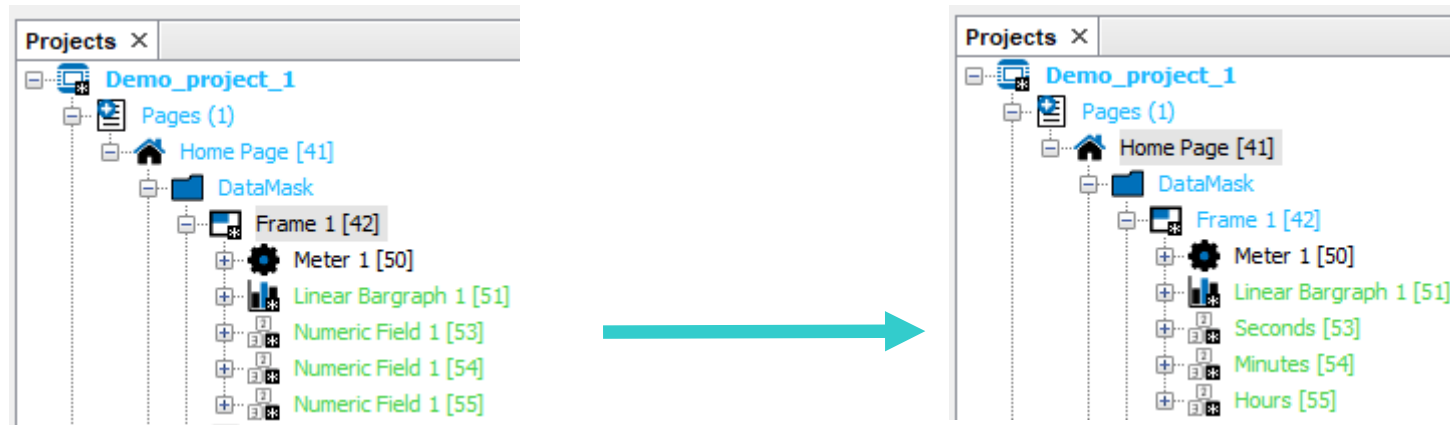


Use the alignment buttons to align selected objects

Save and download the project to the device and test it



## 5.6 The first project – Numeric field – Complete the clock

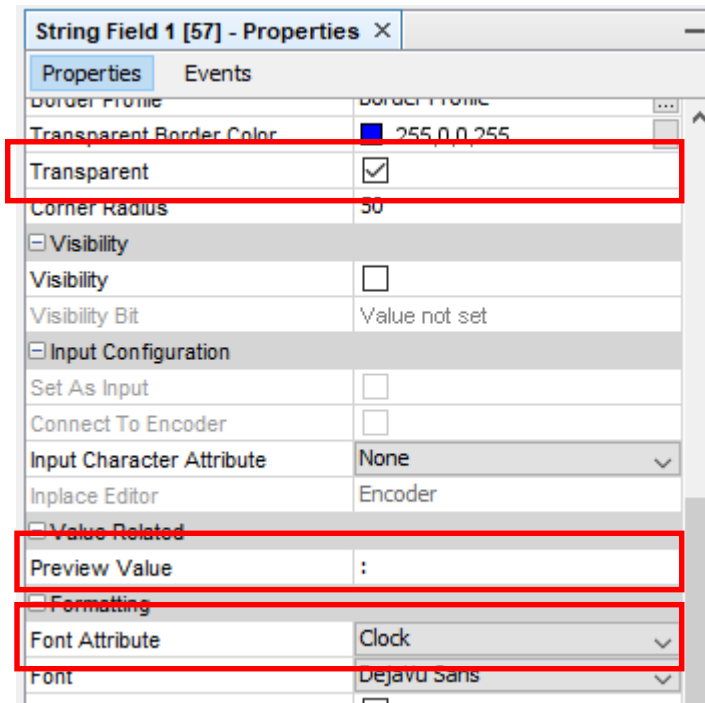
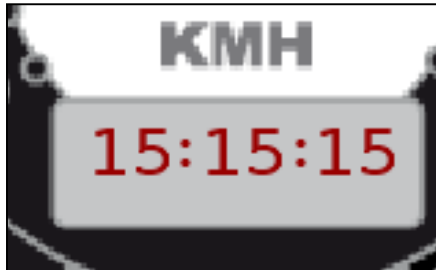


Hint:

Change the name of the objects in the properties to something meaningful

Helpful for larger projects, for colleagues to understand and for our technical support

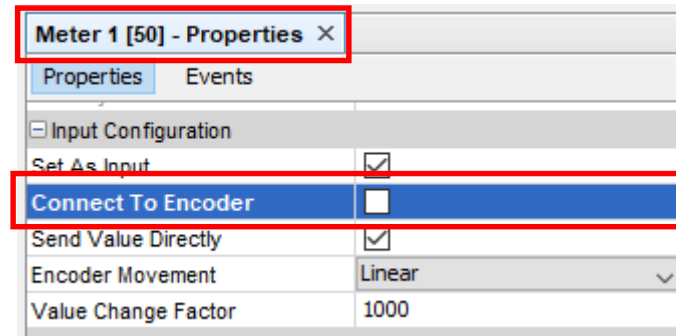
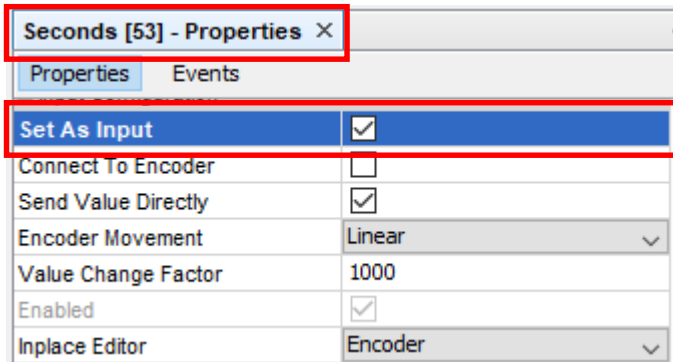
## 5.6 The first project – Numeric field – Complete the clock



Try using two string fields to create the colons

-> The content of the string fields can be changed with a double click or by changing the property *Preview Value* of the String Fields

## 5.6 The first project – Numeric field – Let's set the clock



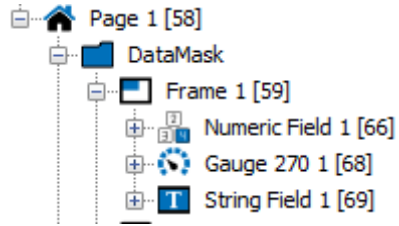
Sometimes the clock will be set wrong, so we want to be able to set it

Select the numeric fields and activate the property *Set As Input*

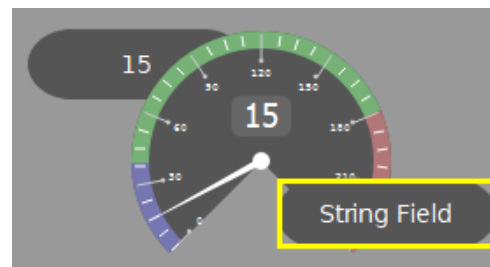
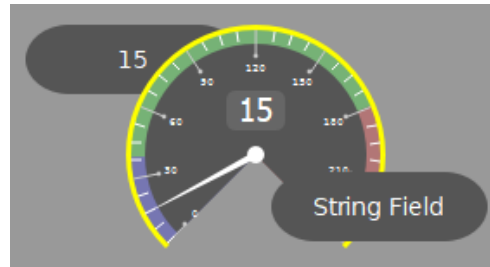
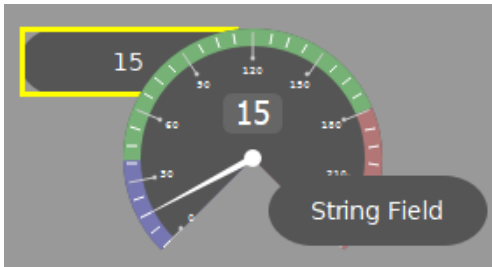
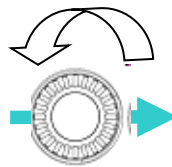
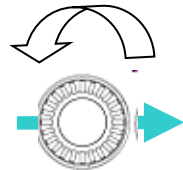
-> Remember the property *Connect To Encoder* from the meter object? Deactivate it to be able to access the numeric fields

-> Did you have trouble clicking on the numeric fields because of the string fields?

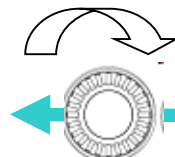
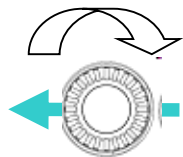
## 5.6 The first project – Numeric field – Z-Order



Objects have an order reflected by the position in the project tree

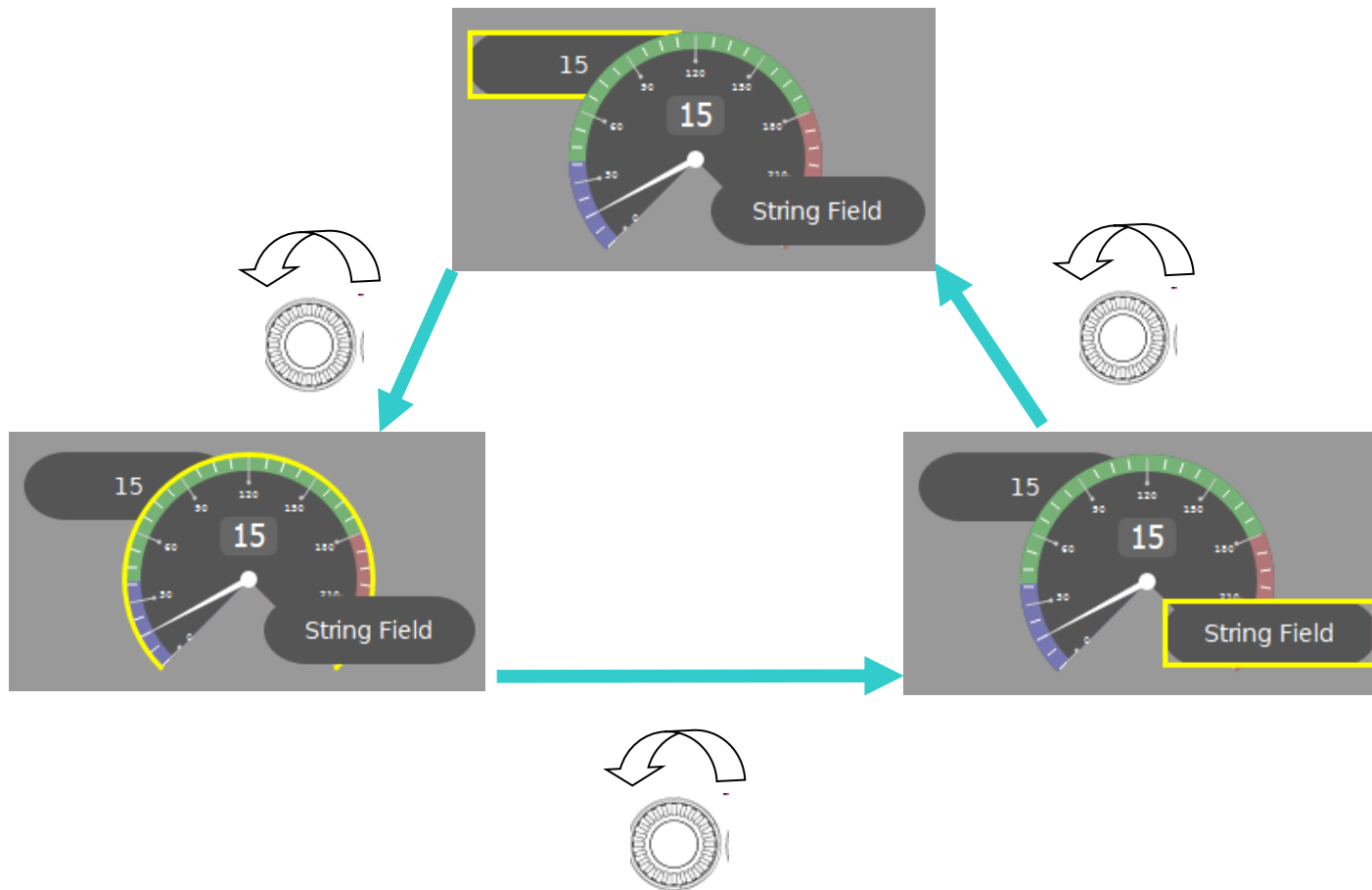


This also determines which object is selected at startup and in which order they are selected by the encoder





## 5.6 The first project – Numeric field – Z-Order – Roll over

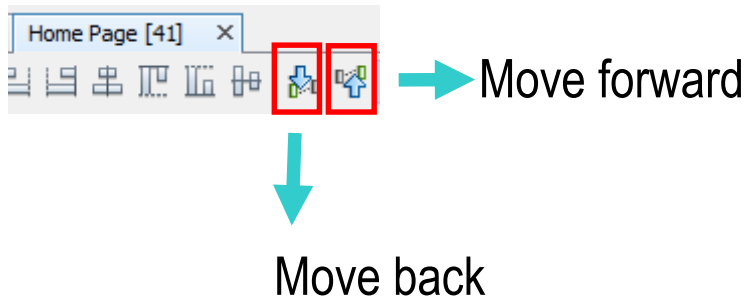


There's a property *Roll over* for frames which will remove the "end" when navigating with the Encoder

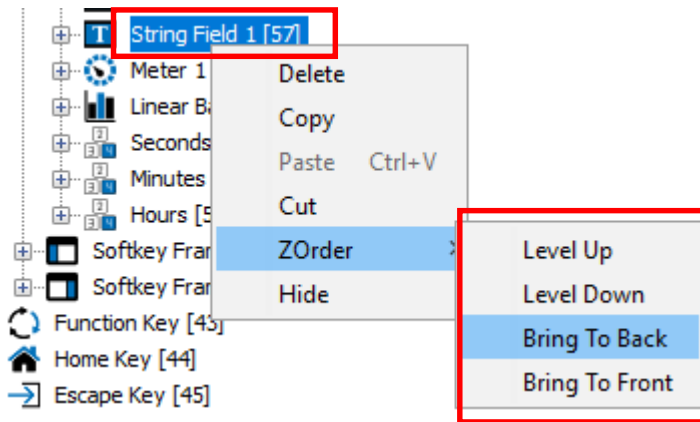
The property can only be set with JavaScript (setProperty function)



## 5.6 The first project – Numeric field – Z-Order



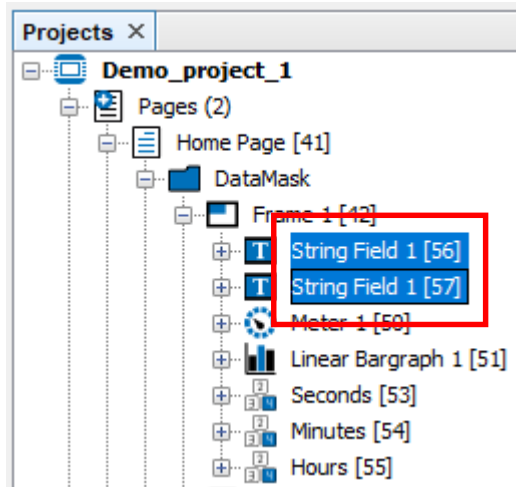
Change the Z-Order by using the buttons in the editor toolbar



or by right clicking the object and choosing ZOrder in the context menu

OR with Drag&Drop directly in the project tree!

## 5.6 The first project – Numeric field – Let's set the clock



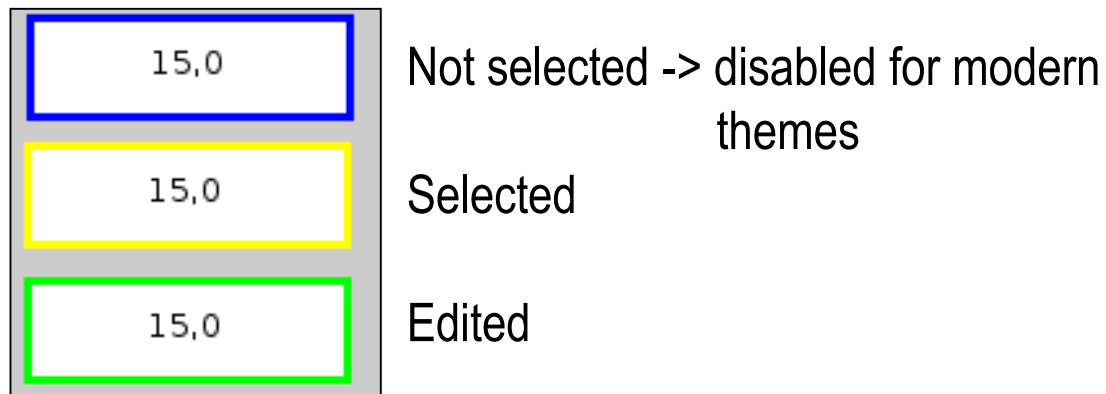
-> Bring the colon string fields to the back so they don't disturb you

Save and download the project to the device and test it





## 5.6 The first project – Numeric field – borders



Borders are used to show which object is currently selected / edited by the encoder

## 5.7 The first project – The second page (Navigation)

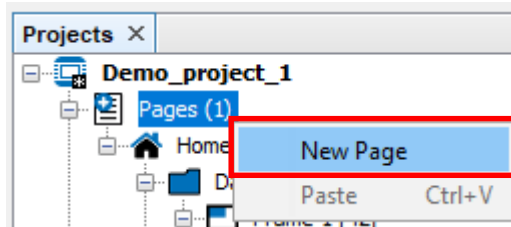
Goals:

Create a second page

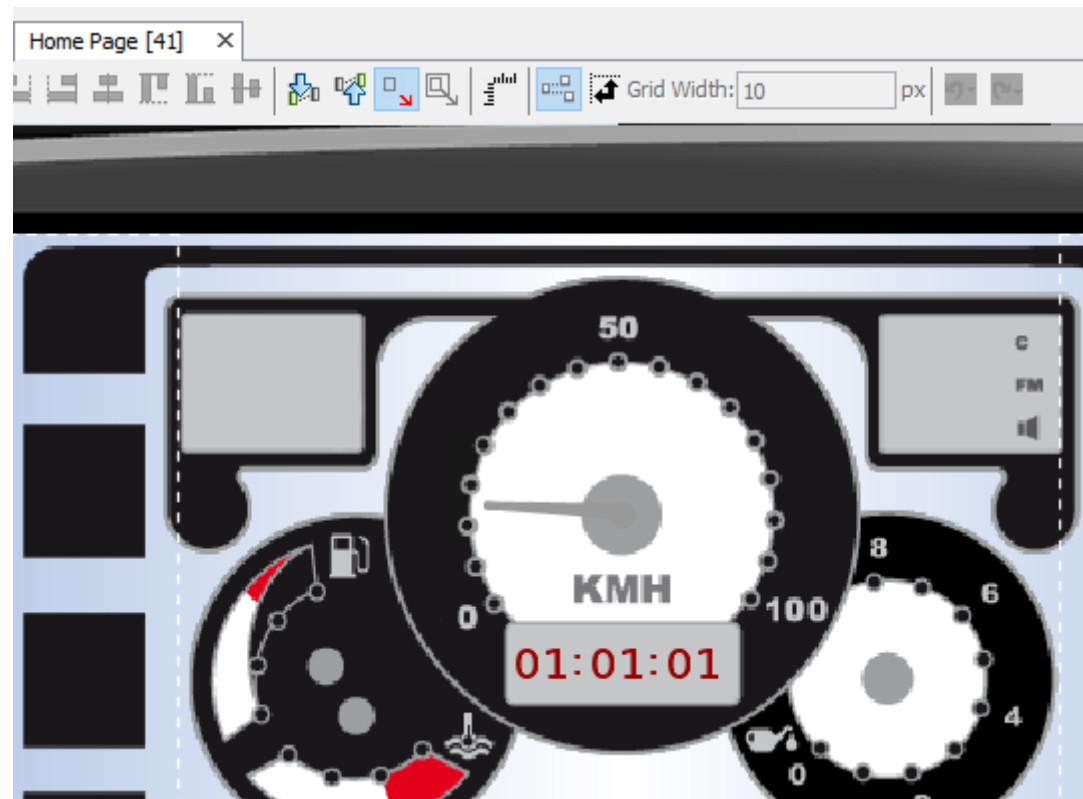
Configure soft keys with images

Configure the navigation between pages

## 5.7 The first project – The second page (Navigation)

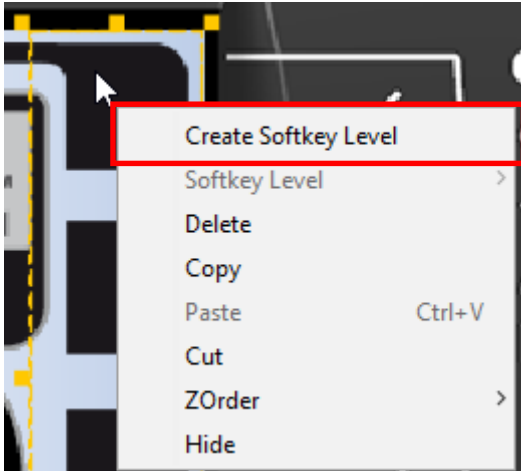


Create a new page by right clicking *Pages* and selecting *New Page*



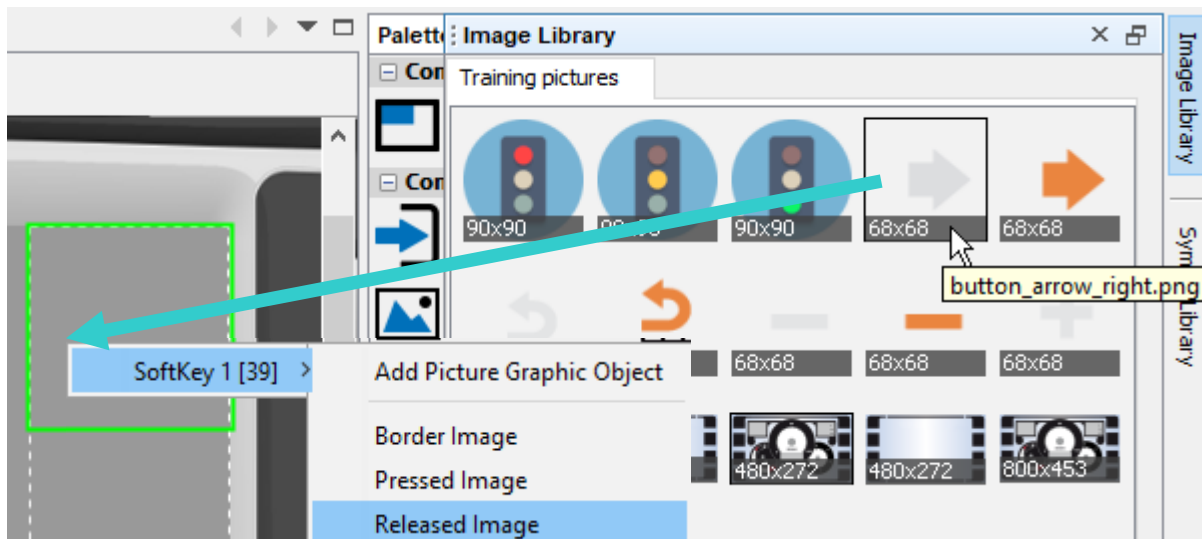
Go back to the Home Page

## 5.7 The first project – The second page (Navigation) – The keys



To use the soft keys, right click on the right soft key frame and select *Create Softkey Level*

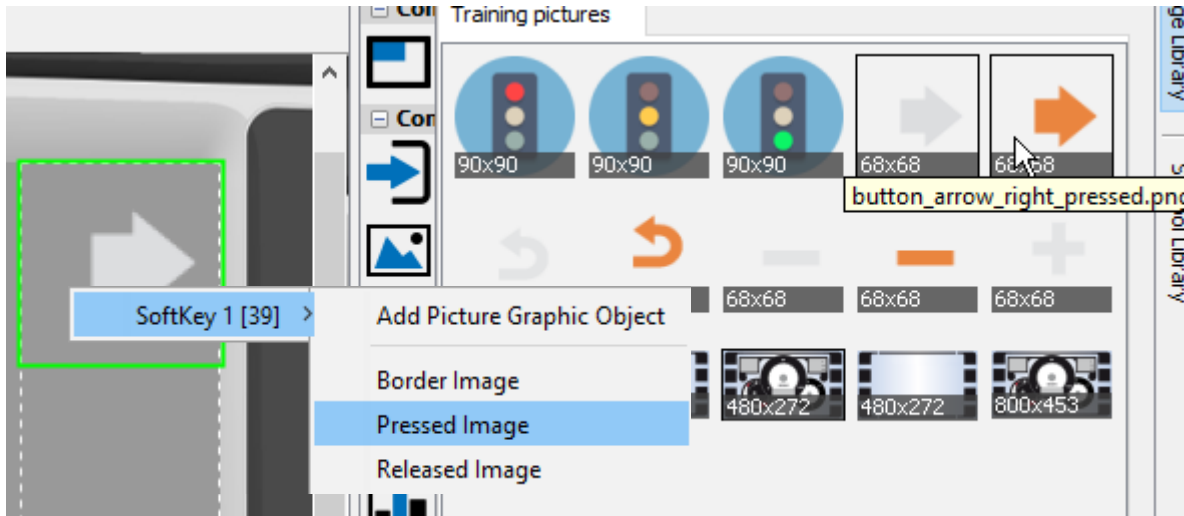
Now you can access the single keys



Drag & drop the gray arrow right from the Image Library onto the softkey

In the context menu, select SoftKey1 and then Released Image

## 5.7 The first project – The second page (Navigation) – The keys



Open the Image Library again and drag & drop The orange arrow right on the SoftKey1 and select Pressed Image in the context menu

SoftKey 1 [70] - Properties	
Properties	Events
Key Number	100
Latching	<input type="checkbox"/>
<b>Pressed State</b>	<input checked="" type="checkbox"/>
Pressed Object ID	-1
Released Object ID	-1
Pressed Image	BUTTON_ARROW_RIGHT_...
Released Image	BUTTON_ARROW_RIGHT_...

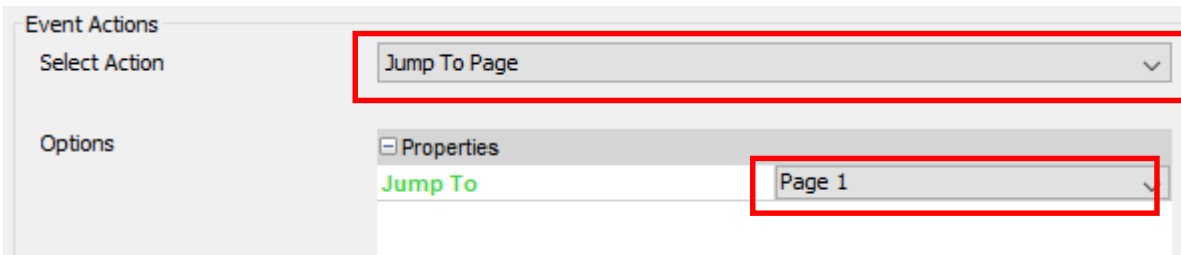
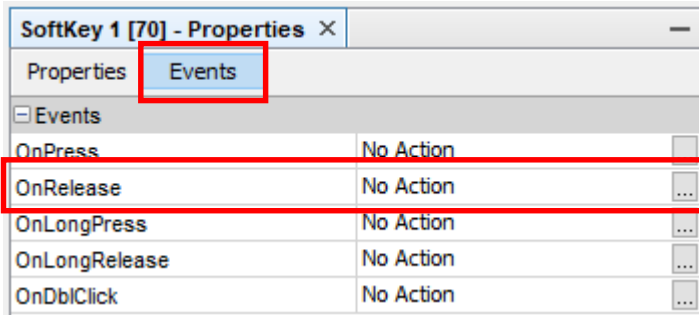
Result:




You can check the results by switching the property *Pressed State*\*

Disable the property when you're done

## 5.7 The first project – The second page (Navigation) – There...



Select the soft key, go to the tab Events and select the  button of the event *OnRelease*

Uncheck *No Action*, choose the action *Jump To Page* and select *Page 1*



## 5.7 The first project – The second page (Navigation) – ...and back again



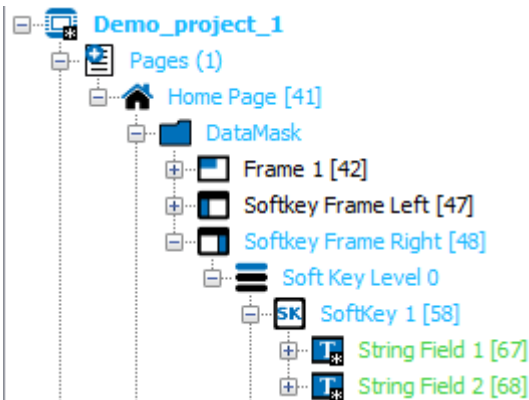
Open the Page 1 and do the following:

- Disable *Draw Border*
- Load the image file *Training\_Background\_2.png* from the Image Library as the background image of the frame
- Create a soft key level in the right soft key frame

## 5.7 The first project – The second page (Navigation) – And back again



String Field 1 [87] - Properties	
Properties	Events
Border Profile	border Profile ...
Transparent Border Color	255,0,0,255 ...
Transparent	<input checked="" type="checkbox"/>
Corner Radius	50
[-] Visibility	
Visibility	<input type="checkbox"/>
Visibility Bit	Value not set
[-] Input Configuration	
Set As Input	<input type="checkbox"/>
Connect To Encoder	<input type="checkbox"/>
Input Character Attribute	None
Inplace Editor	Encoder
[-] Value Related	
Preview Value	Home



This time drag a string field into the first soft key, make it transparent, use the font attribute *Text White*, change the font to *DejaVu Sans* and the Preview Value to *Home*

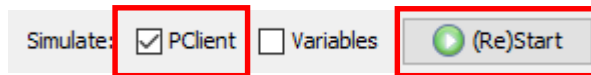
Enable the *Pressed State* and repeat the same, only use the font attribute *Text Bold White*

Use the event *On Release*, choose the action *Jump To Page* and select the home page

Save the project



## 5.7 The first project – The second page (Navigation) – The simulation



To test the navigation, enable the PClient simulation in the toolbar and press the button *(Re)Start*

In the simulation you can test your project easily

Use the mouse wheel for encoder movement

Everything works except hardware specific variables

## 5.8 The first project – Buttons

Goals:

Create some buttons

Configure their functionality

Learn about latching

## 5.8 The first project – Buttons – Push button

Var  Hide pre-defined variables

Group Name	Name	Length (Bits)	Index
Common	push_button	16	0x3000
Common	switch	16	0x3000

Diagram showing a button object and a numeric field object. Context menus are open for both, showing options like 'Variable Reference'. Arrows point from the 'push\_button' variable in the table below to the button and numeric field objects in the diagram.

Name	Length (Bits)	Index	SubIndex	Owner
push_button	16	0x3000	0x01	PClient
switch	16	0x3000	0x02	PClient

In the variable view, hide the pre-defined variables and create two variables

Rename the variables by double clicking the name, change them to *push\_button* and *switch*

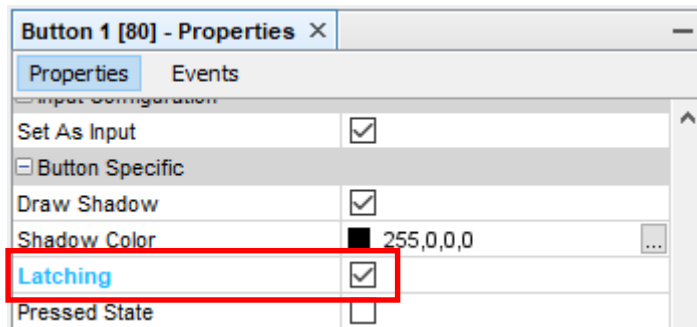
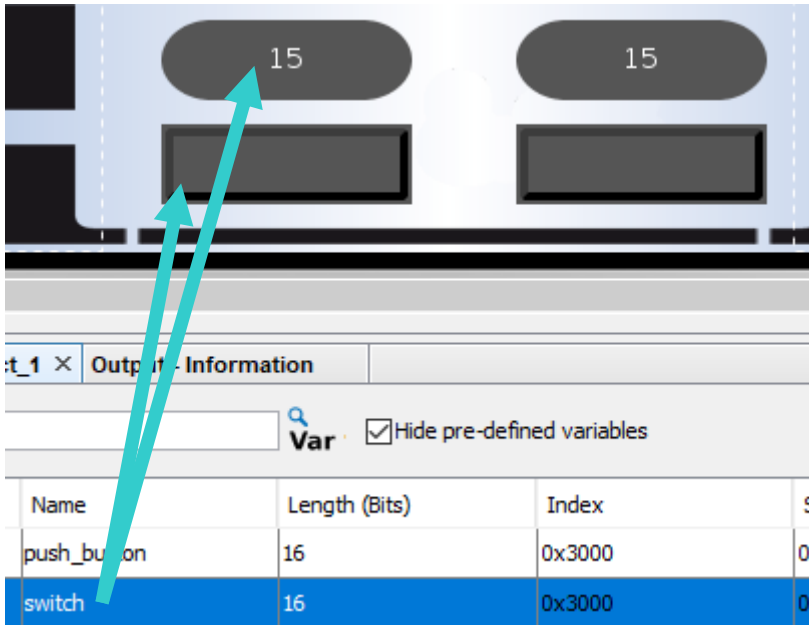
Create a button object and a numeric field

Reference both objects with the variable *push\_button*

Save and download the project to the device and test it



## 5.8 The first project – Buttons – Switch button



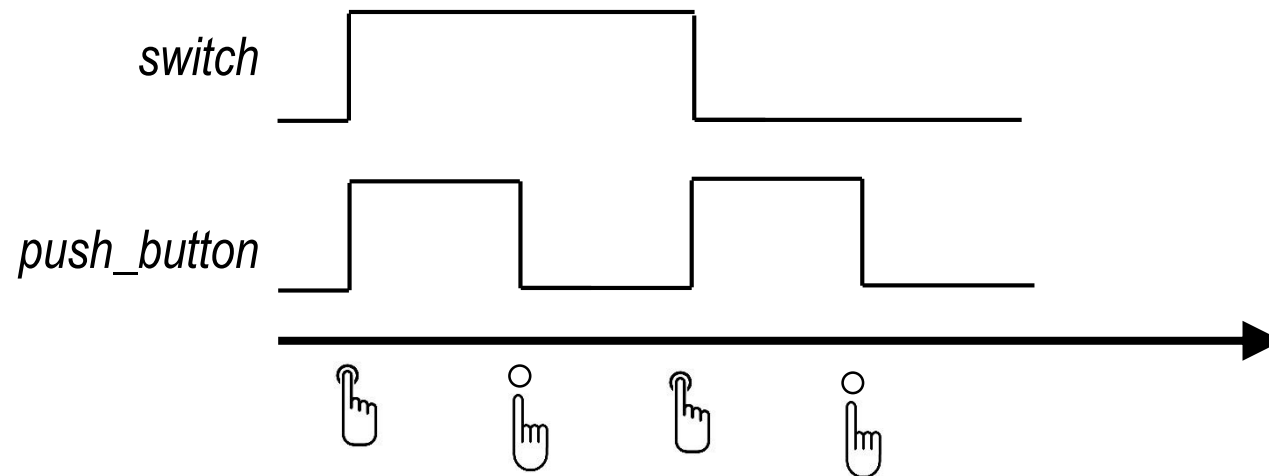
Create another button and numeric field and reference the variable *switch*

In the properties for the button, activate the property *Latching*

Save and download the project to the device and test it



## 5.8 The first project – Buttons – Switch button



Create another button and numeric field and reference the variable *switch*

In the properties for the button, activate the property *Latching*

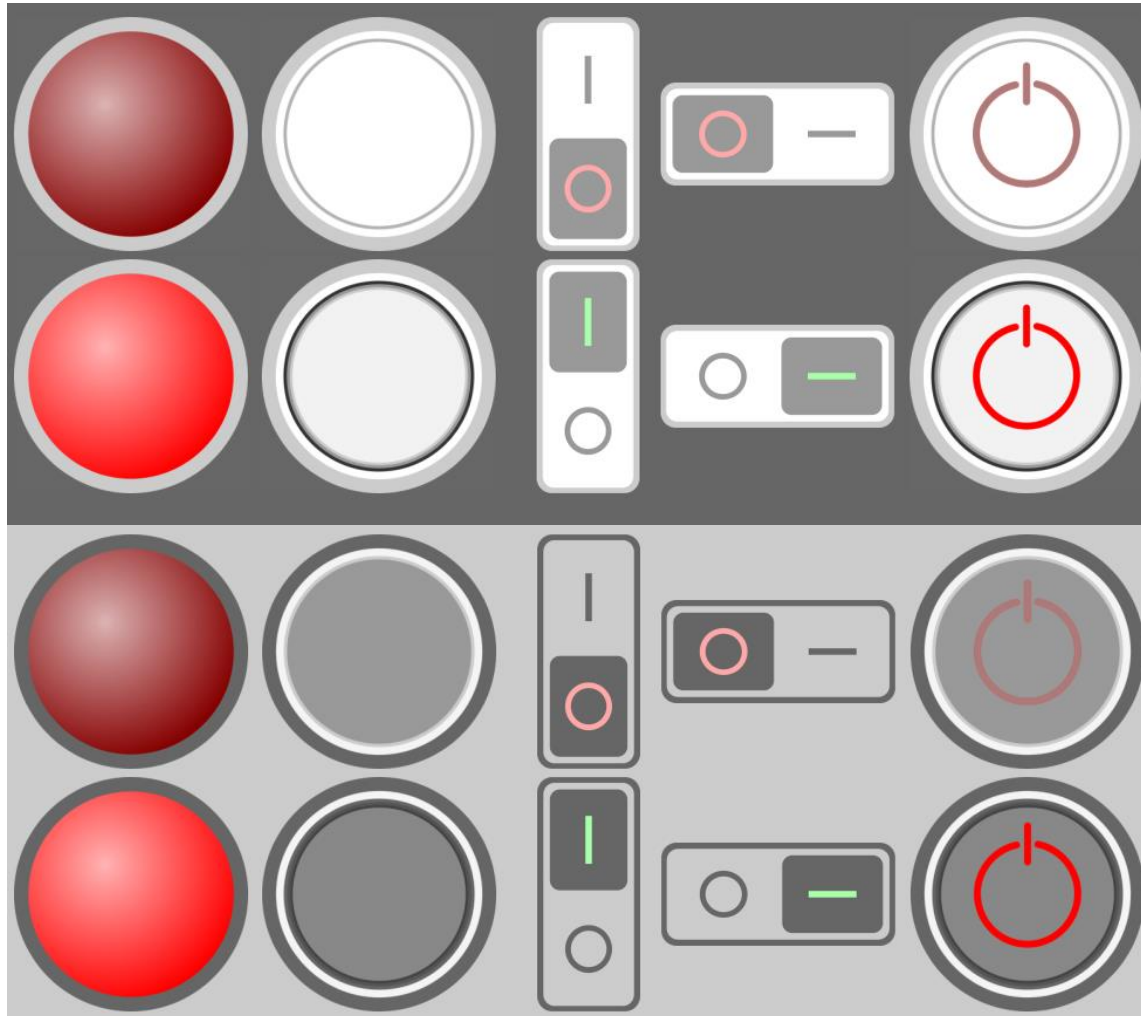
Save and download the project to the device and test it



-> Observe the different behavior

-> Besides connecting buttons with a variable, the events can be configured as well

## 5.8 The first project – Buttons – New objects – Lamps and Switches



New in OPUS Projektor 2018

Lamp

Push Switch

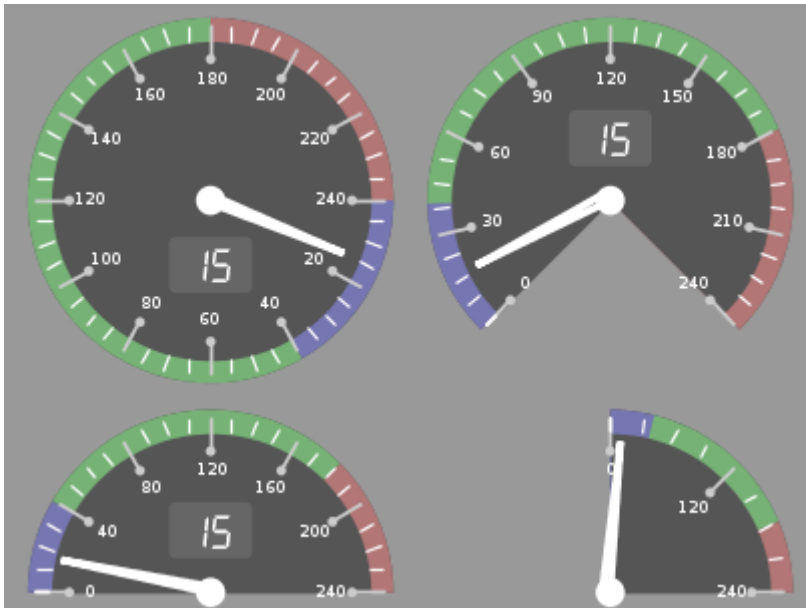
Rocker Switch

Power Switch

The objects behave like buttons, with pre-configured images



## 5.8 The first project – Buttons – New objects – Gauge



New from OPUS Projektor 2018

Gauge

Gauge 270

Gauge 180

Gauge 90

Alternative for the meter object with ticks, tick values, digital value and colored value regions

Tons of properties to modify almost all details of the object

## 5.9 The first project – List objects

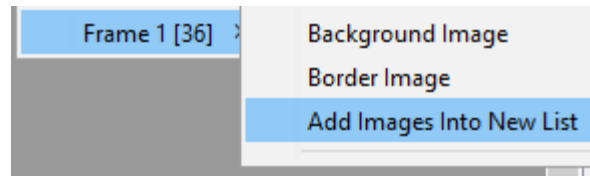
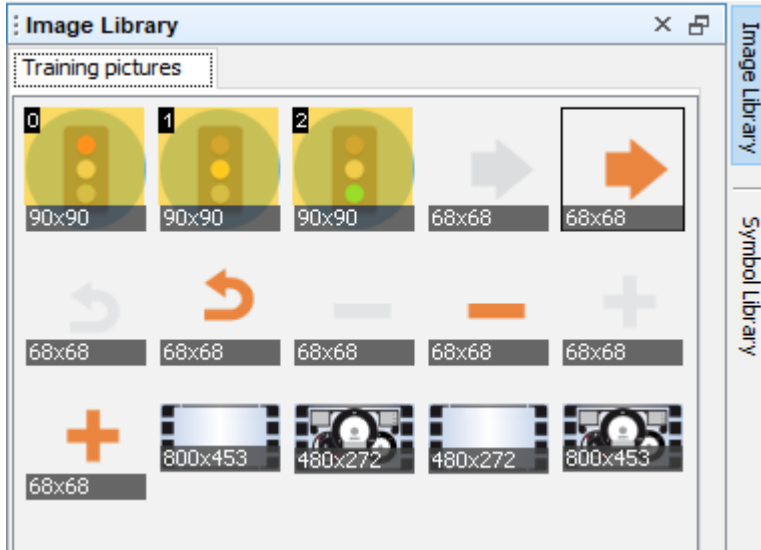
Goals:

Learn what list objects are used for

Load them with objects

Configure which object is seen

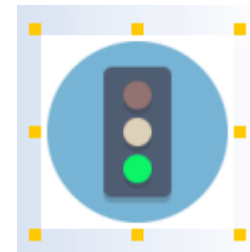
## 5.9 The first project – List objects – Use case 1: Icon folder



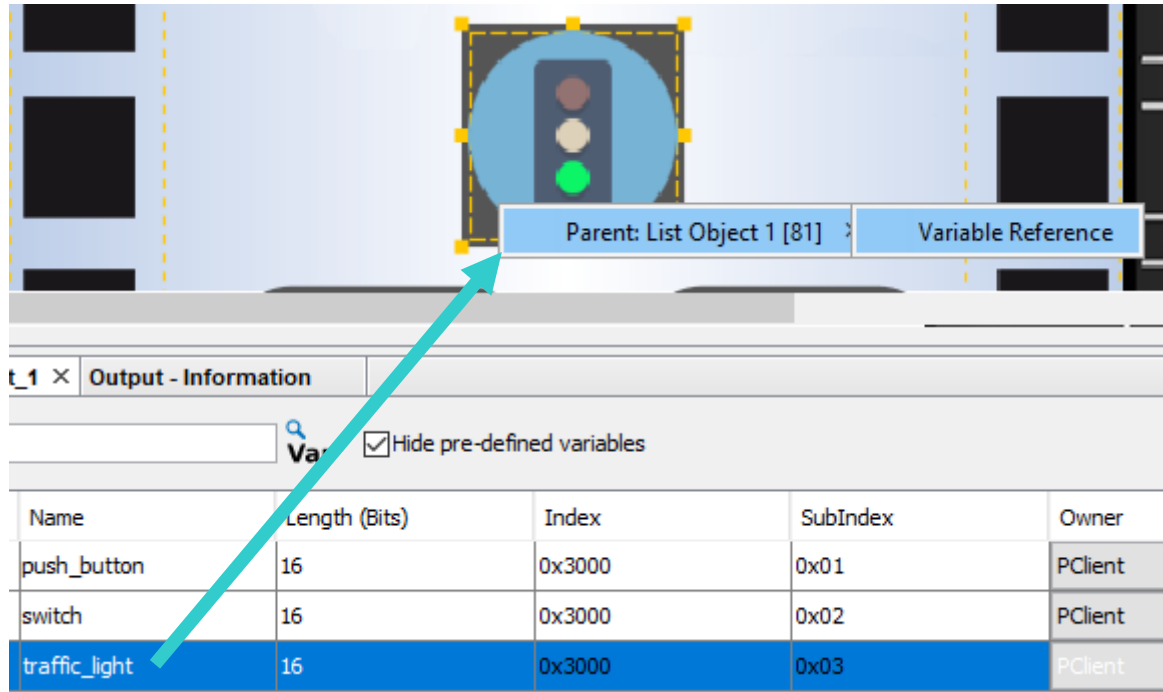
Select the three traffic light images in the Image Library by pressing CTRL and clicking to select

The number shows the order in which they will be used

Drag & drop the images on the frame and select Add Images Into New List in the context menu



## 5.9 The first project – List objects – Use case 1: Icon folder



The screenshot shows a graphical user interface for a traffic light icon. A yellow dashed box highlights the icon, and a blue tooltip below it reads "Parent: List Object 1 [81]" and "Variable Reference". Below the icon is a table with columns: Name, Length (Bits), Index, SubIndex, and Owner. The table contains three rows: "push\_button", "switch", and "traffic\_light". The "traffic\_light" row is highlighted in blue. A red arrow points from the "traffic\_light" row to the tooltip.

Name	Length (Bits)	Index	SubIndex	Owner
push_button	16	0x3000	0x01	PClient
switch	16	0x3000	0x02	PClient
traffic_light	16	0x3000	0x03	PClient

Create a new variable *traffic\_light*

Reference the list object to the new variable *traffic\_light*

Create a Numeric Field next to the list

Connect the variable to the Numeric Field and make it editable

Save, download & test



## 5.9 The first project – List objects – Use case 2: Drop down menu

The image shows the configuration process for a list object. The 'Set List Items' dialog box contains a table with 10 rows, each with a 'String Field' object. The 'Add' button is highlighted with a red box. The 'Inputlist Specific' panel shows 'Items Visible (Editing Mode)' set to 5. The 'Input Configuration' panel shows 'Set As Input' checked. A preview of a dropdown menu with 5 items is shown at the bottom.

S.No.	Object
0	String Field 1 [48]
1	String Field 1 [49]
2	String Field 1 [50]
3	String Field 1 [51]
4	String Field 1 [52]
5	String Field 1 [53]
6	String Field 1 [54]
7	String Field 1 [55]
8	String Field 1 [56]
9	String Field 1 [57]

**Inputlist Specific**

Item Height	34
List Item Mode	Normal
Items Visible (Normal Mode)	1
Items Visible (Editing Mode)	5
List Items	List Items
Number Of List Items	10
Resize Child	Resize children

**Input Configuration**

Set As Input	<input checked="" type="checkbox"/>
Connect To Encoder	<input type="checkbox"/>
Send Value Directly	<input checked="" type="checkbox"/>
Encoder Movement	Linear
Value Change Factor	1000

**Menu 1**

- Menu 1
- Menu 2
- Menu 3
- Menu 4
- Menu 5

List objects can also be used as drop down menus

Create a List object, open the List Items dialog

Select String Field in the dropdown menu and press Add 10 times to add 10 String Fields

Set the list as input

Set Number of visible items in editing mode to 5

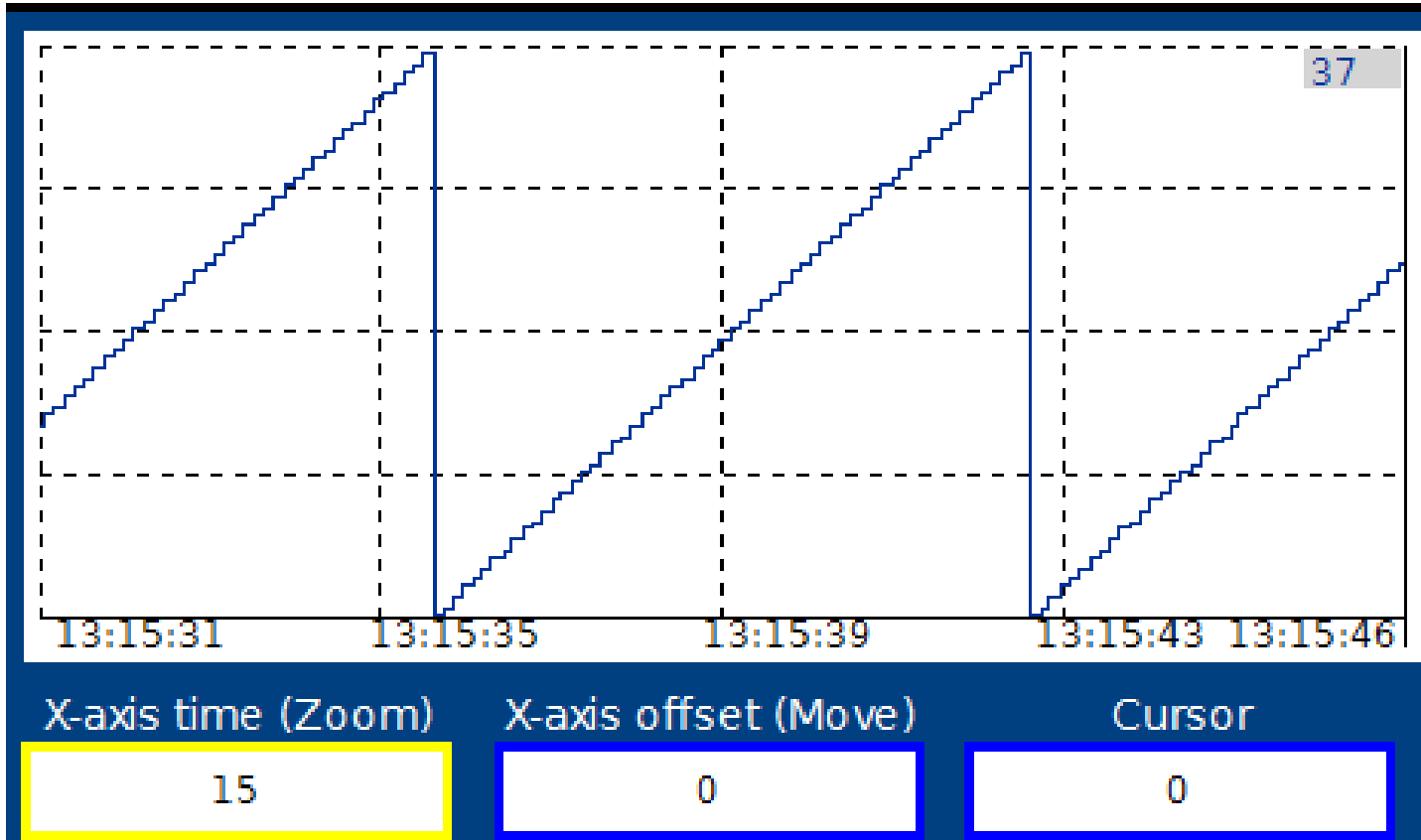
## 5.10 The first project – 2D graph

Goals:

Learn how to configure 2D Graphs

Create input objects for Zoom, Navigation and Cursor

## 5.10 The first project – 2D graph



Create 3 new variables, named

- Zoom (X axis time variable)
- Move (X axis offset variable)
- Cursor (Cursor Position Variable)

Drag & drop a Graph object into the frame

Reference @RTCSeconds as *Variable to Draw* and the 3 new variables for time, offset and cursor

Create 3 editable Numeric Fields and reference them to the 3 variables

Save, download & test



Available as example project



## 5. The first project – Recap

We have learned how to

- update our device software
- create and download projects for our device
- use and configure the objects
  - Meter
  - Picture Graphic
  - Bar Graph
  - Button
  - Numeric Field
  - List Object
  - String Field
  - 2D Graph
- create and use variables and connect them with objects
- use font attributes
- configure editable objects
- integrate images for backgrounds, buttons etc.



## 5. The first project – Recap

We have learned how to

- update our device software
- create and download projects for our device
- use and configure the objects
  - Meter
  - Bar Graph
  - Numeric Field
  - String Field
  - Picture Graphic
  - Button
  - List Object
  - 2D Graph
- create and use variables and connect them with objects
- use font attributes
- configure editable objects
- integrate images for backgrounds, buttons etc.

What did we omit?

- Arched Bargraphs (work just like linear, can be stretched to any angle)
- Symbol Library (comes in Demo mode, over 2400 symbols can be purchased)
- Table (implemented to display J1939 DM1 messages, but can be filled with any JavaScript array)
- Container (just an empty “container” to group objects together)

# Agenda

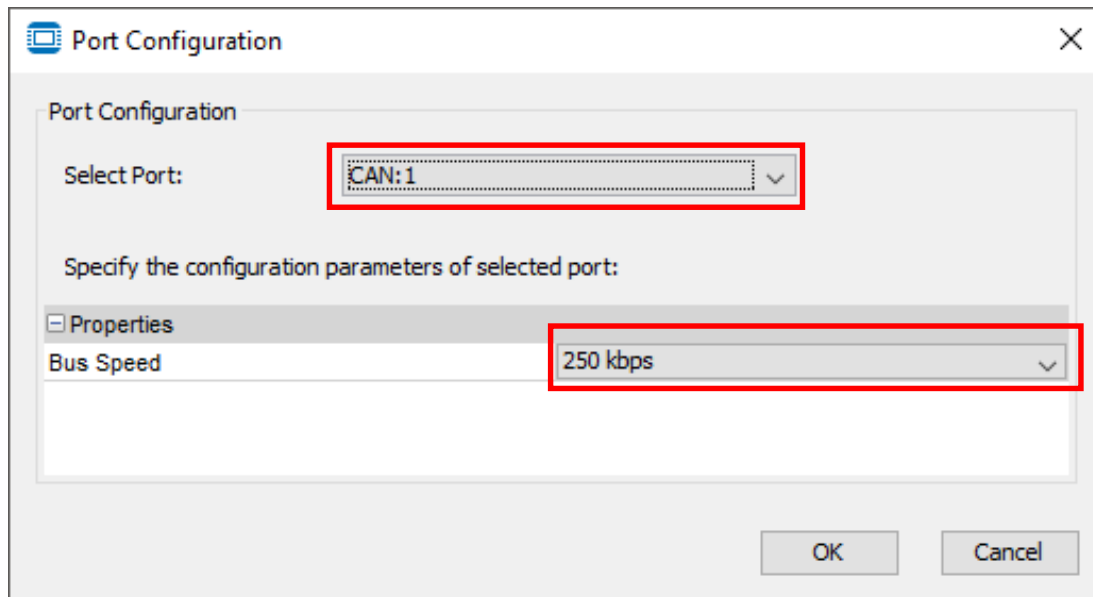
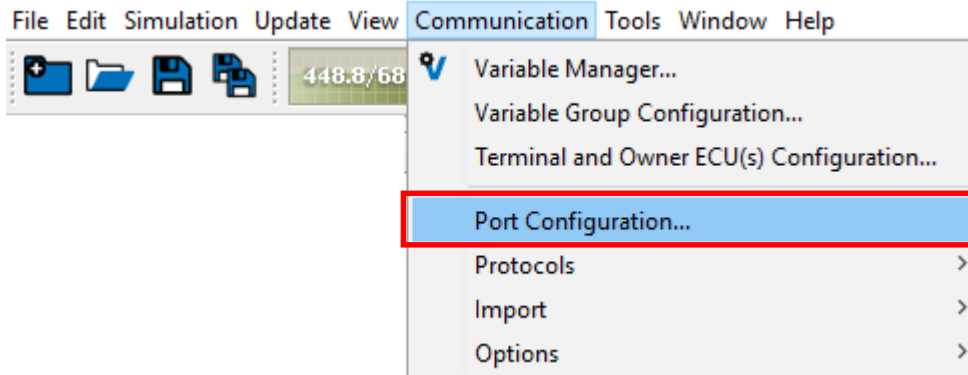
1. Introduction
2. Installation of the Projektor Tool
3. Updating your device
4. Getting to know the Projektor Tool
5. First project
6. CAN communication
7. Extended agenda
8. FAQ

## 6. CAN communication

1. Basic settings
2. Terminal and Owner ECU configuration
3. Creating messages



## 6.1a CAN communication – Basic settings – CAN port

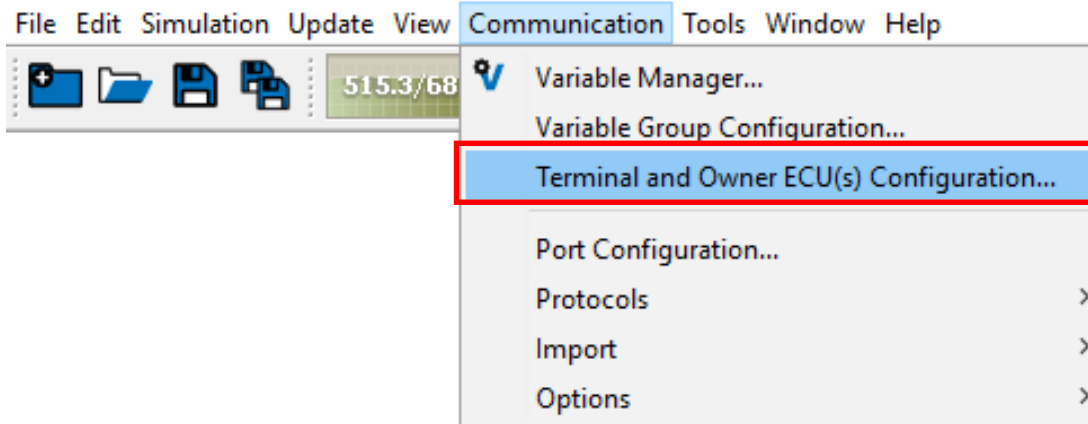


Go to the menu  
*Communication -> Port Configuration...*  
and set the bus speed of the CAN port you  
want to use

Close the dialog

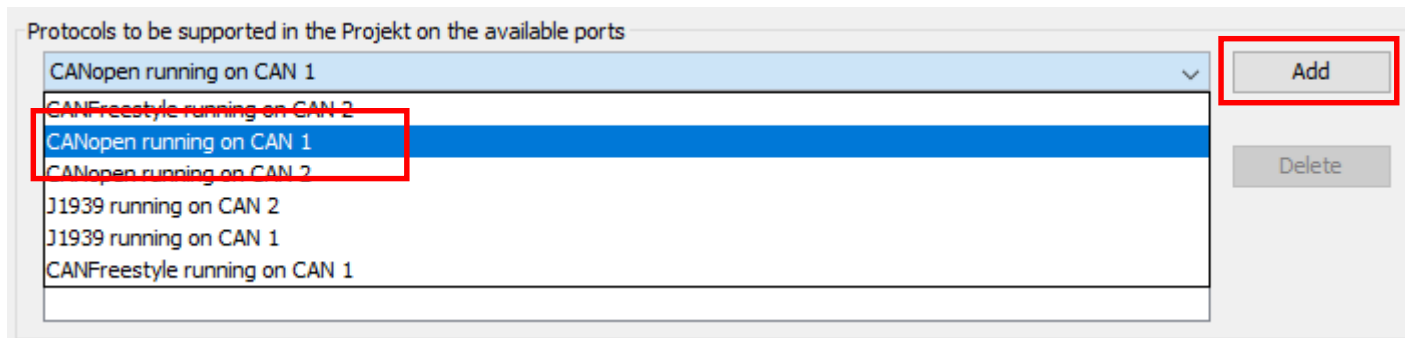
## 6.2a CAN communication – Terminal and Owner ECU

CANopen



Go to the menu  
*Communication -> Terminal and  
Owner ECU(s) Configuration...*

Only three steps until we're ready!



1. Select the CANopen protocol on CAN 1  
and click *Add*

## 6.2a CAN communication – Terminal and Owner ECU



Select protocol and configure Terminal and ECU(s)

CANopen running on CAN 1

Terminal ECU Configuration for CANopen running on CAN1

Properties	
Name	OPUS A8 Eco
Description	
Node ID	0x01
Heartbeat Producer Time (msec)	0
SDO Timeout (msec)	100

Owner ECU(s) network configuration for CANopen running on CAN1

ECU Name: CanOpen\_ECU Add

Select ECU: CanOpen\_ECU Delete

Properties Events

Properties	
Name	CanOpen_ECU
Description	
Heartbeat Time (msec)	0
Node ID	0x02

2. Choose the protocol in the drop-down-list and configure the *Node ID* of the device as needed

Enter a name for an ECU and click *Add*  
configure the *Node ID* of the ECU

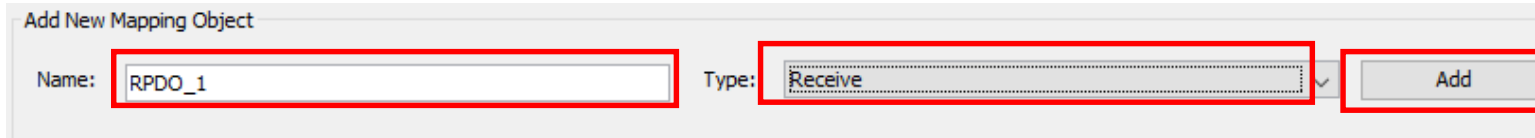
Close the dialog

## 6.3a CAN communication – Creating messages

CANopen



Click on the CANopen shortcut in the tool bar or go to the menu  
*Communication -> Protocols -> CANopen -> Configure Mappings...*



Create a new receive mapping by entering a name, choosing the type *Receive* and clicking *Add*

## 6.3a CAN communication – Creating messages – Receive

Configure Mappings Objects

Select Mapping Object: RPDO\_1 Delete Valid

Properties Configure Mapping Visual CAN Mapping

	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
Byte 1	8	7	6	5	4	3	2	<span style="border: 1px solid red; padding: 2px;">@ProjektCurr 1 L</span>
Byte 2	16	15	14	13	12	11	10	9
Byte 3	24	23	22	21	20	19	18	17
Byte 4	<span style="border: 1px solid red; padding: 2px;">@ProjektCurr 32 MSB</span>	31	30	29	28	27	26	25
Byte 5	32	31	30	29	28	27	26	25

Available Variables

Var page

Communication

@ProjektCurrentPage

For now the properties of the mapping are okay, so go to the tab *Visual CAN Mapping*, search for the variable `@ProjektCurrentPage` and drag & drop it into the mapping matrix to Byte 1, Bit 1

Close the dialog by clicking *OK*

The message has the following parameters:  
 COB-ID – 0x201  
 DLC (message length) – 4 bytes

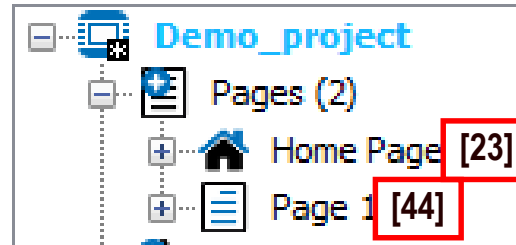
Save and download the project to the device and test it





## 6.3a CAN communication – Creating messages – Receive

Transmit	Message	D...	Data	C	Count	Trigger
	000h	2	01 00	W..	1	Manual
	201h	4	17 00 00 00	W..	8	Manual
	201h	4	2C 00 00 00	W..	8	Manual



Use a program like PCANView and a CAN dongle to test the CAN communication

The first message is needed to enable the CAN communication of the device (as per CANopen standard):

ID – 0x000, DLC – 2, Data – 01 00 h  
This enables all devices on the CAN bus

The second and third messages switch the page.

23 in hex -> 17  
44 in hex -> 2C

# 6.3a CAN communication – Creating messages – Transmit

Add New Mapping Object

Name:  Type:

---

Configure Mappings Objects

Select Mapping Object:

---

Properties  Configure Mapping  Visual CAN Mapping

	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
Byte 1	8	7	6	5	4	3	2	push_button 1 LSB
Byte 2	push_button 16 MSB	15	14	13	12	11	10	9
Byte 3	24	23	22	21	20	19	18	switch 17 LSB
Byte 4	switch 32 MSB	31	30	29	28	27	26	25
Byte 5	40	39	38	37	36	35	34	@ProjektCurr 33 LSB

Available Variables

Var: disp

- Communication
- Display Backlight
  - @DispBacklightAmbientLowP:
  - @DispBacklightFilterFactor
  - @DispBacklightIntensity
  - @DispBacklightIntensity0
  - @DispBacklightIntensity10
  - @DispBacklightIntensity100
  - @DispBacklightIntensity20
  - @DispBacklightIntensity30
  - @DispBacklightIntensity40
  - @DispBacklightIntensity50

Configure Selected Variable Mapping

Byte Bit Position	Length	Group	Variable

Go to the mapping dialog again

Create a Transmit mapping

The mapping automatically gets the COB ID 0x181

In the tab *Visual CAN Mapping* add the variables *push\_button*, *switch* and *@DispBacklightIntensity*



Variables are organized in groups  
All user-created variables go into the group "Common"

## 6.3a CAN communication – Creating messages – Transmit



Properties	
Configure Mapping	
Visual CAN Mapping	
Configure properties of selected Mapping Object	
Properties	Events
General	
ID	62
Name	RPDO_1
Type	Receive
Description	
Receive From	CanOpen_ECU
PDO Number	0
Communication Parameter Index	0x1400
Mapping Parameter Index	0x1600
COB ID	0x201
Object Status	Active
CAN ID Type	11 Bit
Transmission Type	Asynchronous Transmission (255)
Event Timer	0

In the properties, the *Transmission Type* is set to on value change

For cyclic sending, set the time in *Event Timer* in milliseconds

Close the dialog

Save and download the project to the device and test it



## 6.3a CAN communication – Creating messages – Transmit

	Message	DLC	Data
Receive	181h	5	00 00 00 00 64
Transmit	000h	2	01 00
	201h	4	17 00 00 00
	201h	4	51 00 00 00

At first – no messages from the device

->Send the start message again  
ID – 0x000, DLC – 2, Data – 01 00 h

Now press the buttons or change the  
backlight intensity

-> every value change triggers the message

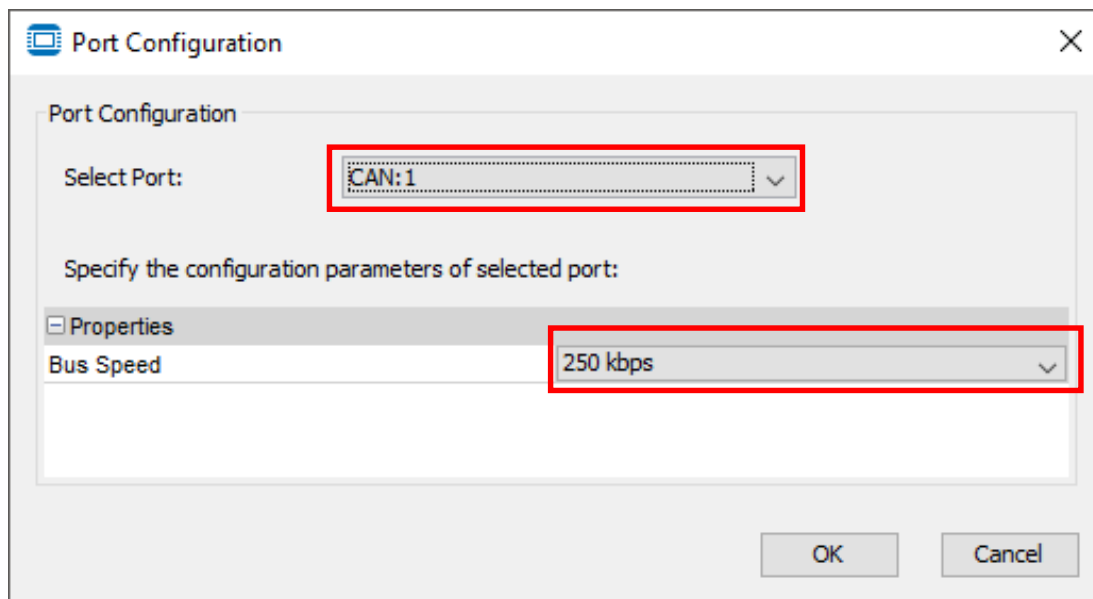
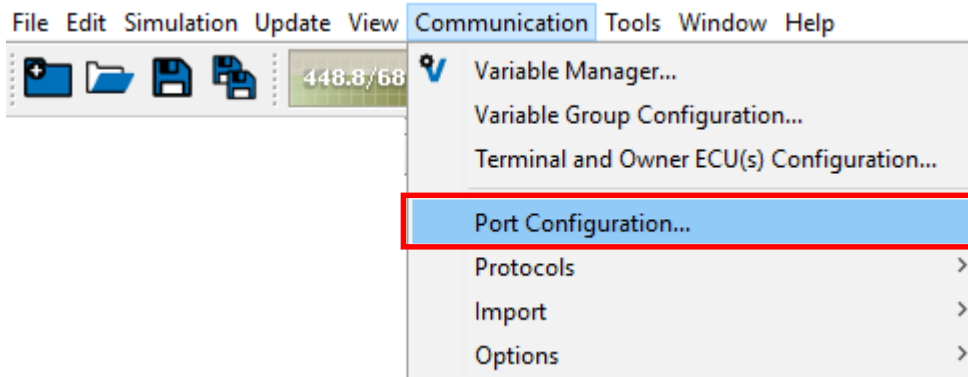
## 6.3a CAN communication – Recap

We have learned how to:

- configure the basic CAN settings for CANopen
- configure the OPUS device and ECU settings
- create CAN mappings for receive and transmit PDO messages
- put variables on the CAN bus

Continue with chapter 6

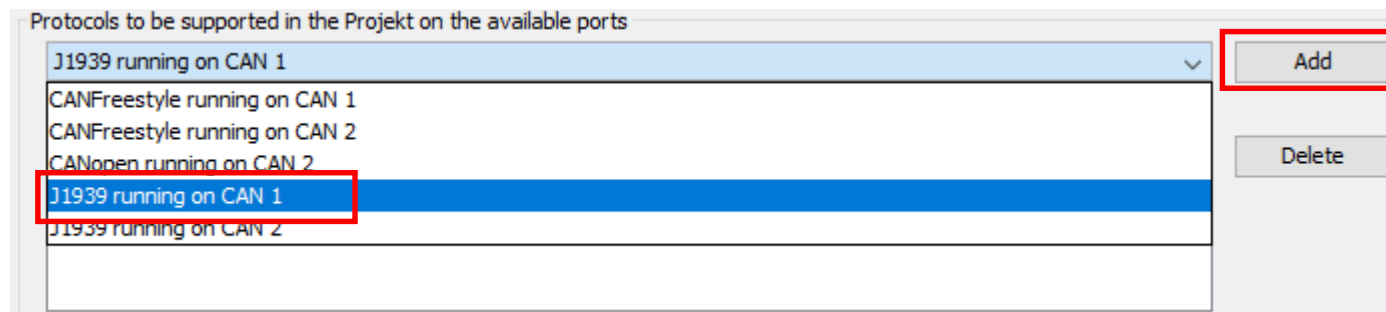
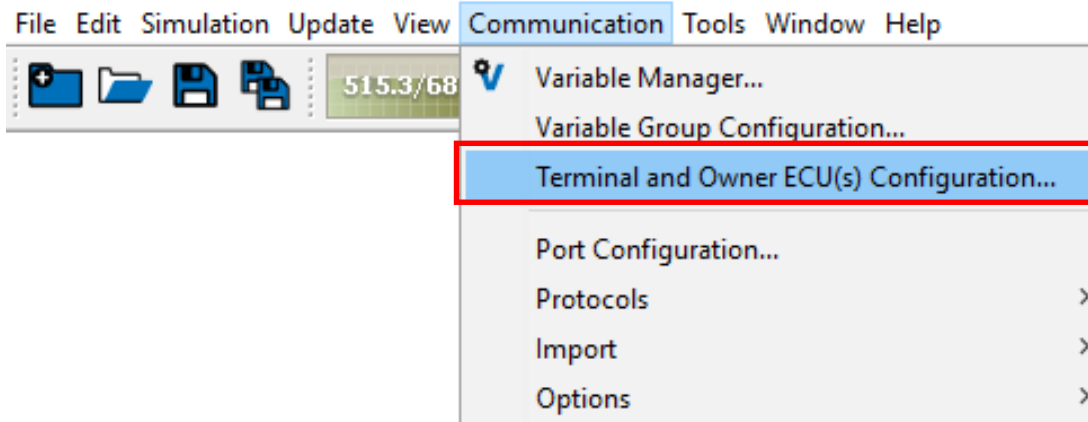
## 6.1b CAN communication – Basic settings – CAN port



Go to the menu  
*Communication -> Port Configuration...*  
and set the bus speed of the CAN port you  
want to use

Close the dialog

## 6.2b CAN communication – Terminal and Owner ECU



Go to the menu  
*Communication -> Terminal and  
Owner ECU(s) Configuration...*

Only three steps until we're ready!

1. Select the J1939 protocol on CAN 1  
and click *Add*

## 6.2b CAN communication – Terminal and Owner ECU

Select protocol and configure Terminal and ECU(s)

J1939 running on CAN 1

Terminal ECU Configuration for J1939 running on CAN1

Properties	
Name	OPUS A8 Eco
Description	
Address Claim	Static
Preferred Source Address	0x54
Terminal Name	00-00-3C-00-15-60-00-00

Owner ECU(s) network configuration for J1939 running on CAN1

ECU Name: J1939\_ECU Add

Select ECU: J1939\_ECU Delete

Properties	
Name	J1939_ECU
Description	
Address Claim	Static
Source Address	0x01
J1939 ECU Name	FF-FE-00-FF-FF-E0-00-00

2. Choose the protocol in the drop-down-list and configure the *Preferred Source Address* as needed

3. Enter a name for an ECU and click *Add* configure the *Source Address* from the ECU

Close the dialog



## 6.3b CAN communication – Creating messages



Click the J1939 shortcut in the toolbar or go to the menu *Communication -> Protocols -> J1939 -> Configure Mappings...*

Add New Mapping Object

Name:  Type:

Create a new receive mapping by entering a name and clicking *Add*

## 6.3b CAN communication – Creating messages



Properties Configure Mapping Visual CAN Mapping

Configure properties of selected Mapping Object

Properties Events

General

ID	70
Name	PGN_R1
Type	Receive
Description	
Mapping Variable Type	Length
Delimiter	*
Receive From	J1939_ECU
Little Endian	<input checked="" type="checkbox"/>
Mapping Length (in Byte)	8
Object Status	Active
PGN	0xFABC
PGN Priority	6

Request Settings

Enable Request	<input type="checkbox"/>
Use Standard Request	<input checked="" type="checkbox"/>

In the Properties, change *Receive From* to your ECU (e.g. J1939\_ECU)

Enter *0xFABC* as PGN

## 6.3b CAN communication – Creating messages – Receive

Configure Mappings Objects

Select Mapping Object: PGN\_R1

Properties | Configure Mapping | **Visual CAN Mapping**

	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
Byte 1	8	7	6	5	4	3	2	@ProjektCurr 1
Byte 2	16	15	14	13	12	11	10	9
Byte 3	24	23	22	21	20	19	18	17
Byte 4	@ProjektCurr 32 MSB	31	30	29	28	27	26	25
Byte 5	40	39	38	37	36	35	34	33

Available Variables

Var page

@ProjektCurrentPage

Configure Selected Variable Mapping

Byte ...	Length	Is Co...	Cons...	Group	Variable	Mask	Offset1	Shift ...	Scale	Offset2	Signed	Forw...

Go to the tab *Visual CAN Mapping*, search for the variable `@ProjektCurrentPage` and drag & drop it into the mapping matrix to Byte 1, Bit 1

Close the dialog by clicking *OK*

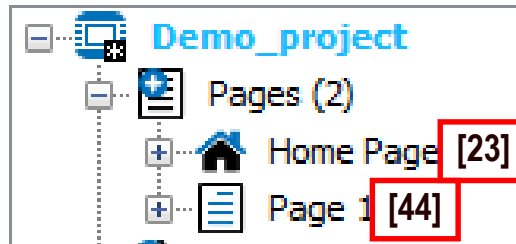
The message has the following parameters:  
 PGN – 0x18FABC01  
 DLC (message length) – 8 bytes

Save and download the project to the device and test it



## 6.3b CAN communication – Creating messages – Receive

<input type="checkbox"/>	Message	DLC	Data
Transmit	18FABC01h	4	17 00 00 00
	18FABC01h	4	2C 00 00 00



Use a program like PCANView and a CAN dongle to test the CAN communication

The two messages switch the page.

23 dec in hex is 17

44 dec in hex is 2C

## 6.3b CAN communication – Creating messages – Transmit

Add New Mapping Object

Name:  Type:

Properties **Configure Mapping** Visual CAN Mapping

Configure properties of selected Mapping Object

Name	PGN_T1
Type	Transmit
Description	
Mapping Variable Type	Length
Delimiter	*
<b>Transmit To</b>	J1939_ECU
Little Endian	<input checked="" type="checkbox"/>
Mapping Length (in Byte)	8
Object Status	Active
<b>PGN</b>	0x0FABD
PGN Priority	6
<input type="checkbox"/> Request Settings	
Enable Request	<input type="checkbox"/>
Request Message	0x00 Data Bytes ( 0-0-0-0-0-0-0-0 )
<input type="checkbox"/> Receive Settings	
<input type="checkbox"/> Transmit Settings	
Use as Write Request	<input checked="" type="checkbox"/>
<b>Send Value On</b>	Any Variable Change
Select Variable(1x)	None
Transmission Period (in ms)	0

Go to the mapping dialog again

Create a Transmit mapping

Set the PGN to 0xFABD, *Transmit To* to your ECU, *Send Value On* to Any Variable Change

## 6.3b CAN communication – Creating messages – Transmit

Configure Mappings Objects

Select Mapping Object: PGN\_T1

Properties Configure Mapping **Visual CAN Mapping**

	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
Byte 1	8	7	6	5	4	3	2	push_button 1 LSB
Byte 2	push_button 16 MSB	15	14	13	12	11	10	9
Byte 3	24	23	22	21	20	19	18	switch 17 LSB
Byte 4	switch 32 MSB	31	30	29	28	27	26	25
Byte 5	@DispBacklig 40 MSB	39	38	37	36	35	34	@DispBacklig 33 LSB
Byte 6	48	47	46	45	44	43	42	41

In the tab *Visual CAN Mapping* add the variables *push\_button*, *switch* and *@DispBacklightIntensity*

Close the dialog

Save and download the project to the device and test it



Variables are organized in groups  
All user-created variables go into the group “Common” by default

## 6.3b CAN communication – Creating messages – Transmit



<input type="checkbox"/>	Message	DLC	Data
Receive	18FABD54h	5	00 00 01 00 4C

Press the buttons or change the  
backlight intensity  
-> every value change triggers the message

## 6.4b CAN communication – Recap



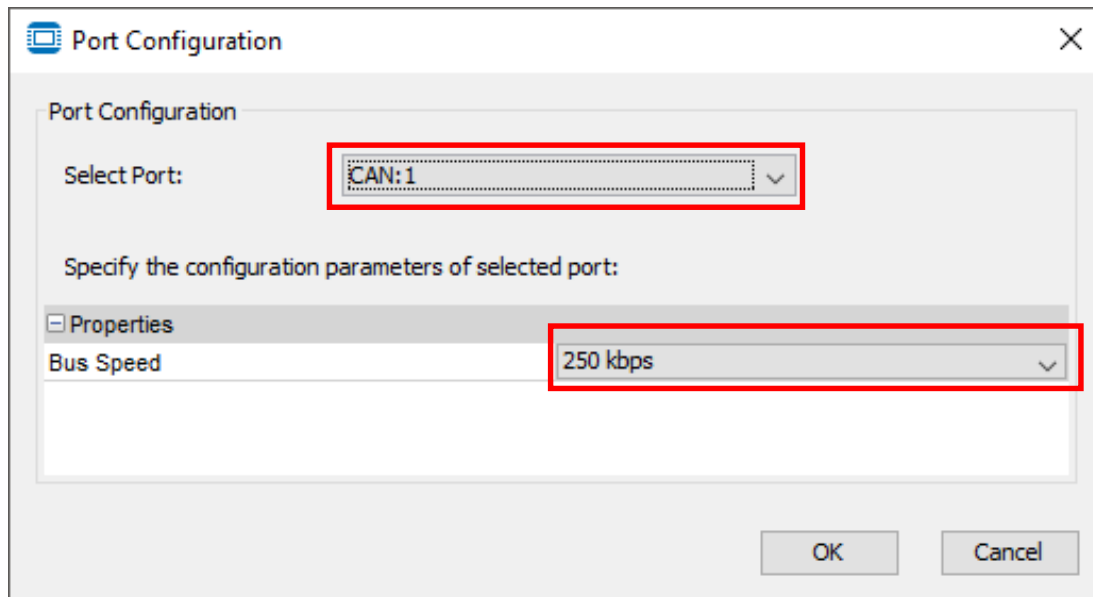
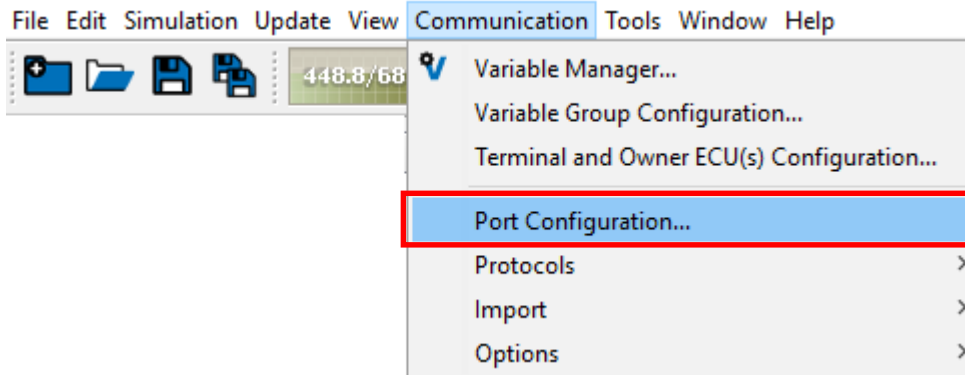
We have learned how to:

- configure the basic CAN settings for J1939
- configure the OPUS device and ECU settings
- create CAN mappings for receive and transmit PGN messages
- put variables on the CAN bus

Continue with chapter 6



## 6.1c CAN communication – Basic settings – CAN port

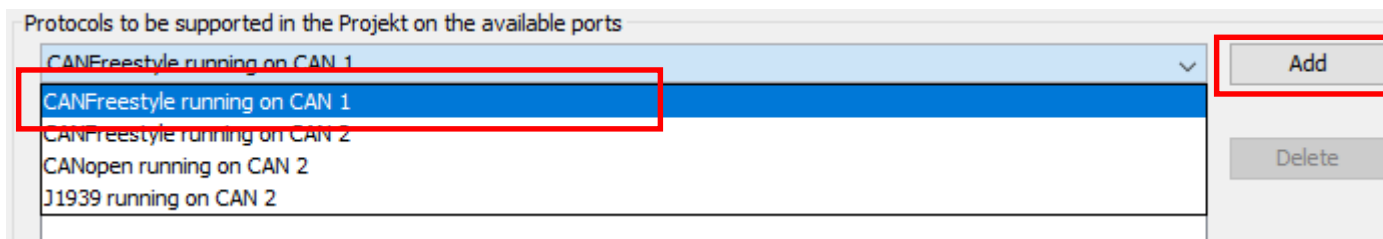
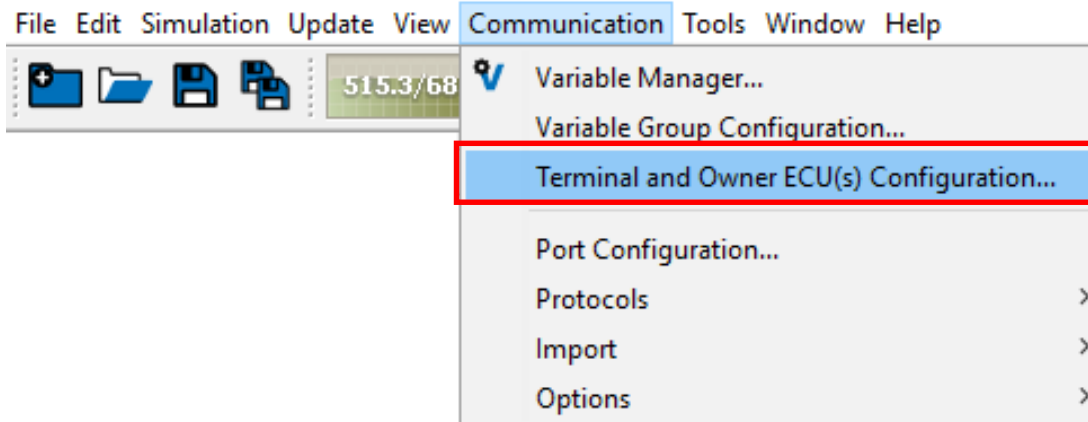


Go to the menu  
*Communication -> Port Configuration...*  
and set the bus speed of the CAN port you  
want to use

Close the dialog

## 6.2c CAN communication – Terminal and Owner ECU

# CANFreestyle



Go to the menu  
*Communication -> Terminal and  
Owner ECU(s) Configuration...*

Only three steps until we're ready!

1. Select the CANFreestyle protocol on  
CAN 1 and click *Add*

## 6.2c CAN communication – Terminal and Owner ECU

# CANFreestyle

Select protocol and configure Terminal and ECU(s)

CANFreestyle running on CAN 1

Terminal ECU Configuration for CANFreestyle running on CAN1

[-] Properties

Name	OPUS A8 Eco
Description	

Owner ECU(s) network configuration for CANFreestyle running on CAN1

ECU Name: CAN\_Freestyle\_ECU Add

Select ECU: CAN\_Freestyle\_ECU Delete

[-] Properties

ID	72
Name	CAN_Freestyle_ECU

In the terminal configuration there are no settings necessary

Enter a name for an ECU and click *Add*

The ECU doesn't need any additional configuration

Close the dialog

## 6.3c CAN communication – Creating messages

# CANFreestyle



Click the CANFreestyle shortcut in the toolbar or go to the menu  
*Communication -> Protocols -> CANFreestyle -> Configure Mappings...*

The image shows a dialog box titled 'Add New Mapping Object'. It has two input fields: 'Name:' with the text 'Message\_R1' and 'Type:' with a dropdown menu showing 'Receive'. To the right of the 'Type:' dropdown is an 'Add' button. Red boxes highlight the 'Name' field, the 'Type' dropdown, and the 'Add' button.

Create a new receive mapping by entering a name, choosing Receive and clicking *Add*

## 6.3c CAN communication – Creating messages – Receive

# CANFreestyle

Properties | Configure Mapping | Visual CAN Mapping

Configure properties of selected Mapping Object

Properties | Events

General

ID	73
Name	Message_R1
Type	Receive
Description	
Receive From	CAN_Freestyle_ECU
Little Endian	<input checked="" type="checkbox"/>
Mapping Length (in Byte)	8
Object Status	Active
CAN ID Type	11 Bit
CAN ID(0x)	0x123
CAN ID Mask	Full 11 bit ID 1 (0x7ff)

Request Settings

In the properties, enter the *CAN ID* 0x123

## 6.3c CAN communication – Creating messages – Receive

Configure Mappings Objects

Select Mapping Object: Message\_R1

Properties | Configure Mapping | Visual | **CAN Mapping**

	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
Byte 1	8	7	6	5	4	3	2	@ProjektCurrentPage 1 LSB
Byte 2	16	15	14	13	12	11	10	9
Byte 3	24	23	22	21	20	19	18	17
Byte 4	@ProjektCurrentPage 32 MSB	31	30	29	28	27	26	25
Byte 5								

Available Variables

Var page

Communication

Projekt

@ProjektCurrentPage

Go to the tab *Visual CAN Mapping*, search for the variable `@ProjektCurrentPage` and drag & drop it into the mapping matrix to Byte 1, Bit 1

Close the dialog by clicking *OK*

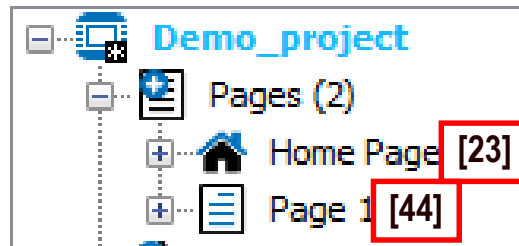
The message has the following parameters:  
 CAN-ID – 0x123  
 DLC (message length) – 8 bytes

Save and download the project to the device and test it



## 6.3c CAN communication – Creating messages – Receive

<input type="checkbox"/>	Message	DLC	Data
Transmit	123h	8	17 00 00 00 00 00 00 00
	123h	8	2C 00 00 00 00 00 00 00



Use a program like PCANView and a CAN dongle to test the CAN communication

The two messages switch the page.

23 dec in hex is 17

44 dec in hex is 2C

# 6.3c CAN communication – Creating messages – Transmit



Add New Mapping Object

Name:  Type:

Configure properties of selected Mapping Object	
General	
ID	74
Name	Message_T1
Type	Transmit
Description	
Transmit To	CAN_Freestyle_ECU
Little Endian	<input checked="" type="checkbox"/>
Mapping Length (in Byte)	8
Object Status	Active
CAN ID Type	11 Bit
CAN ID(0x)	0x124
CAN ID Mask	Full 11 bit ID 1 (0x7ff)
Request Settings	
Enable Request	<input type="checkbox"/>
Request Message	0x00 Data Bytes ( 0-0-0-0-0-0-0-0 )
Receive Settings	
Transmit Settings	
Use as Write Request	<input checked="" type="checkbox"/>
Send Value On	Any Variable Change
Select Variable(Tx)	None
Transmission Period (in ms)	0

Go to the mapping dialog again

Create a Transmit mapping

Set the *CAN ID* for the mapping to 0x124

When should the message be sent?

Set *Send Value On* to *Any Variable Change* and/or set a *Transmission Period*



## 6.3c CAN communication – Creating messages – Transmit

# CANFreestyle

Add New Mapping Object

Name:  Type:

Configure Mappings Objects

Select Mapping Object:

Properties

	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
Byte 1	8	7	6	5	4	3	2	push_button 1 LSB
Byte 2	push_button 16 MSB	15	14	13	12	11	10	9
Byte 3	24	23	22	21	20	19	18	switch 17 LSB
Byte 4	switch 32 MSB	31	30	29	28	27	26	25
Byte 5	@DispBacklig 40 MSB	39	38	37	36	35	34	@DispBacklig 33 LSB

Go to the mapping dialog again

Create a Transmit mapping

Set the *CAN ID* for the mapping to 0x124

In the tab *Visual CAN Mapping* add the variables *push\_button*, *switch* and *@DispBacklightIntensity*

Close the dialog

Save and download the project to the device and test it



## 6.3c CAN communication – Creating messages – Transmit

**CAN**Freestyle

<input type="checkbox"/>	Message	DLC	Data
Receive	18FABD54h	5	00 00 01 00 4C

Press the buttons or change the  
backlight intensity  
-> every value change triggers the message

## 6.4c CAN communication – Recap

**CAN**Freestyle

We have learned how to:

- configure the basic CAN settings for CAN Freestyle
- configure the OPUS device and ECU settings
- create CAN mappings for receive and transmit CAN messages
- put variables on the CAN bus

# Agenda

1. Introduction
2. Installation of the Projektor Tool
3. Updating your device
4. Getting to know the Projektor Tool
5. First project
6. CAN communication
7. Extended agenda
8. FAQ

## 7. Extended agenda

1. **Power management**
2. **Visibility**
3. **JavaScript**
4. Alarms
5. Language dependency
6. Variable Logging

## 7.1. Extended agenda – Power management

OPUS devices work just like computers, they need to be shut down properly before being disconnected from power.

Disconnecting clamp 30 while the device is running can damage files and even the filesystem.

It can also result in longer startup times, because the filesystem has to run something like checkdisk, which takes time and slows the device down.

Always take measures so that the device can be shut down properly.

# 7.1. Extended agenda – Power management

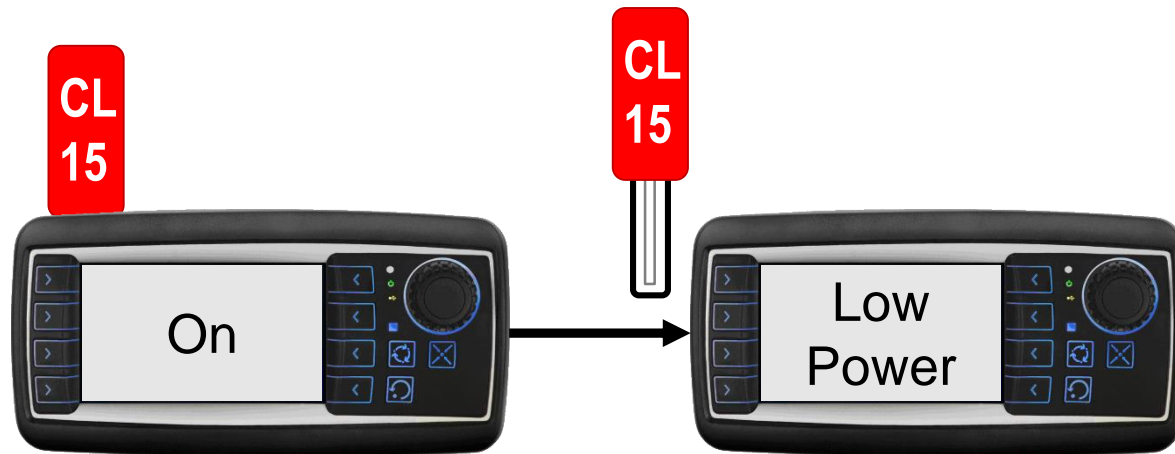
CL  
15



mA @13.5 V

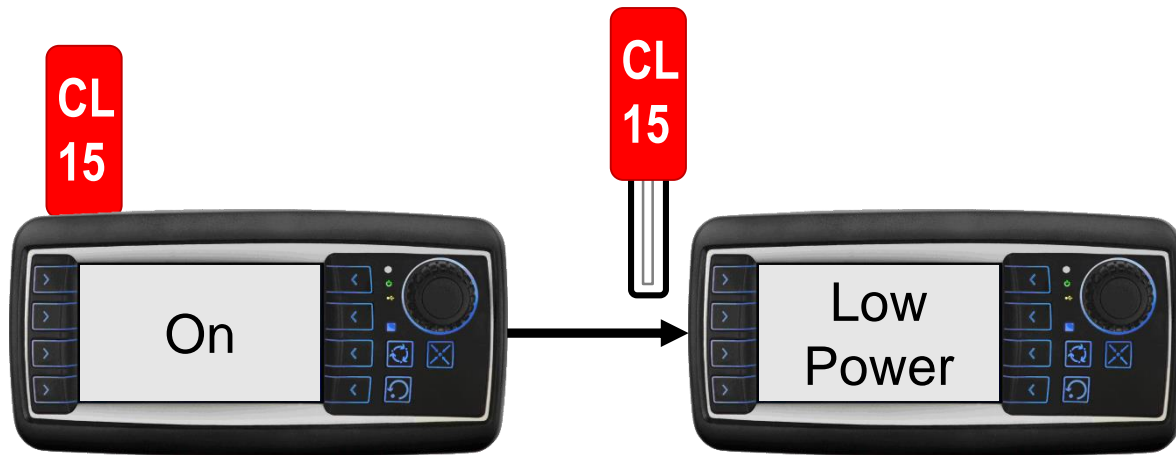
	A3	A6	A6G2	A8
On	430	900	1000	1600

## 7.1. Extended agenda – Power management





# 7.1. Extended agenda – Power management



PClient, JavaScripts, CAN are working

on/off

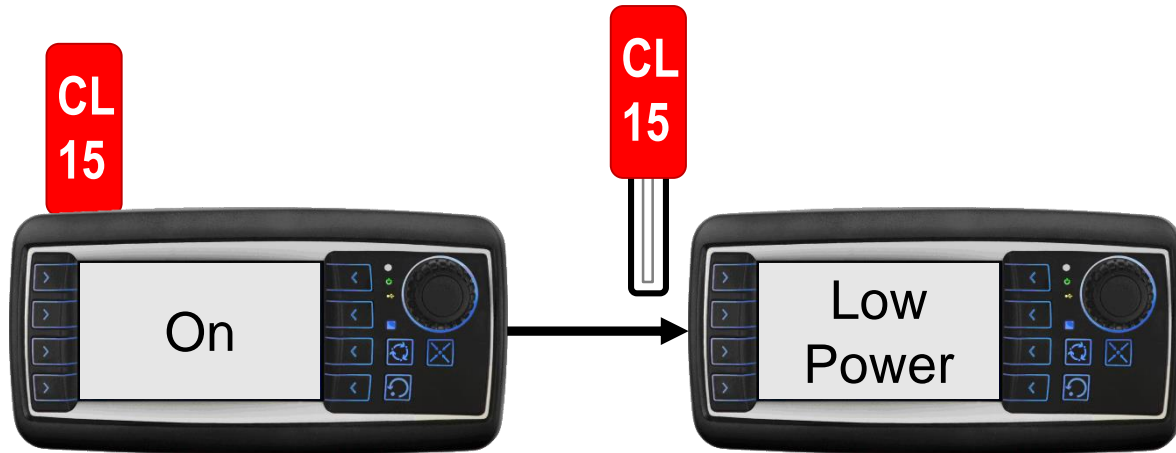
- /  Display function
- /  Display backlight
- /  Key backlight
- /  Key function
- /  Encoder function
- /  Outputs

on/off

- /  Multicolor LED
- /  Beeper / Speaker
- /  Touchscreen
- /  Console (RS232)
- /  Video processing
- /  Camera output

(@PWR\_LowPower\_\_\_\_\_)

# 7.1. Extended agenda – Power management



mA @13.5 V

	A3	A6	A6G2	A8
On	430	900	1000	1600
Low Power*	160	200	200	300

PClient, JavaScripts, CAN are working

- /  Display function
- /  Display backlight
- /  Key backlight
- /  Key function
- /  Encoder function
- /  Outputs
- /  Multicolor LED
- /  Beeper / Speaker
- /  Touchscreen
- /  Console (RS232)
- /  Video processing
- /  Camera output

(@PWR\_LowPower\_\_\_\_\_)

\*With minimal configuration

## 7.1. Extended agenda – Power management



# 7.1. Extended agenda – Power management



## 7.1. Extended agenda – Power management



Nothing is working, no functionality!

# 7.1. Extended agenda – Power management

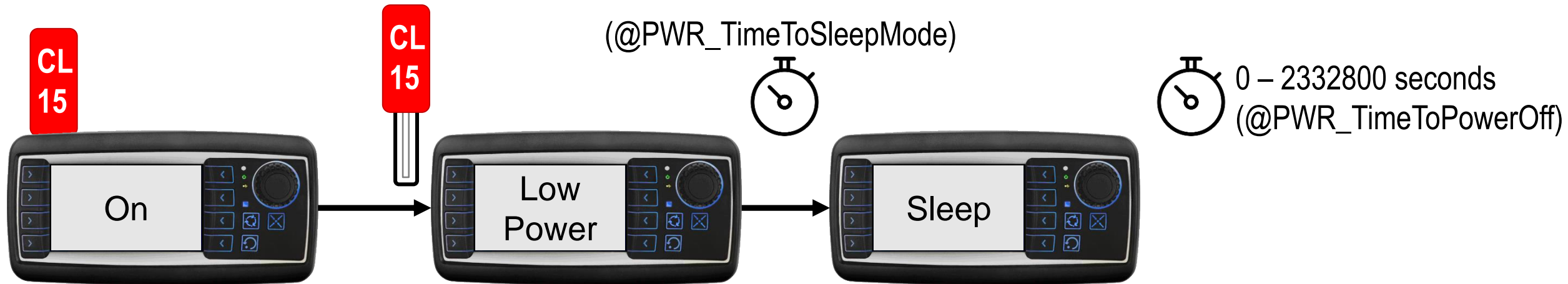


mA @13.5 V

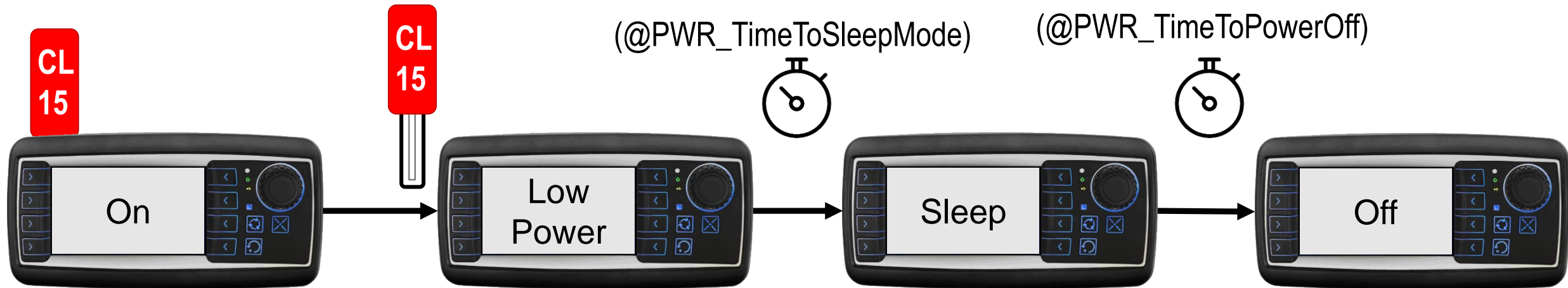
	A3	A6	A6G2	A8
On	430	900	1000	1600
Low Power	160	200	200	300
Sleep	≤ 90	≤ 100	≤ 100	≤ 200

Nothing is working, no functionality!

# 7.1. Extended agenda – Power management

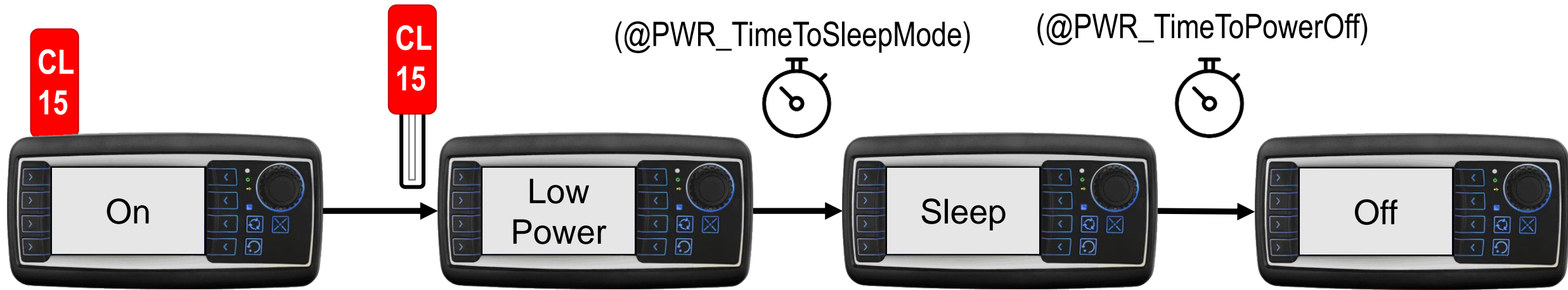


# 7.1. Extended agenda – Power management





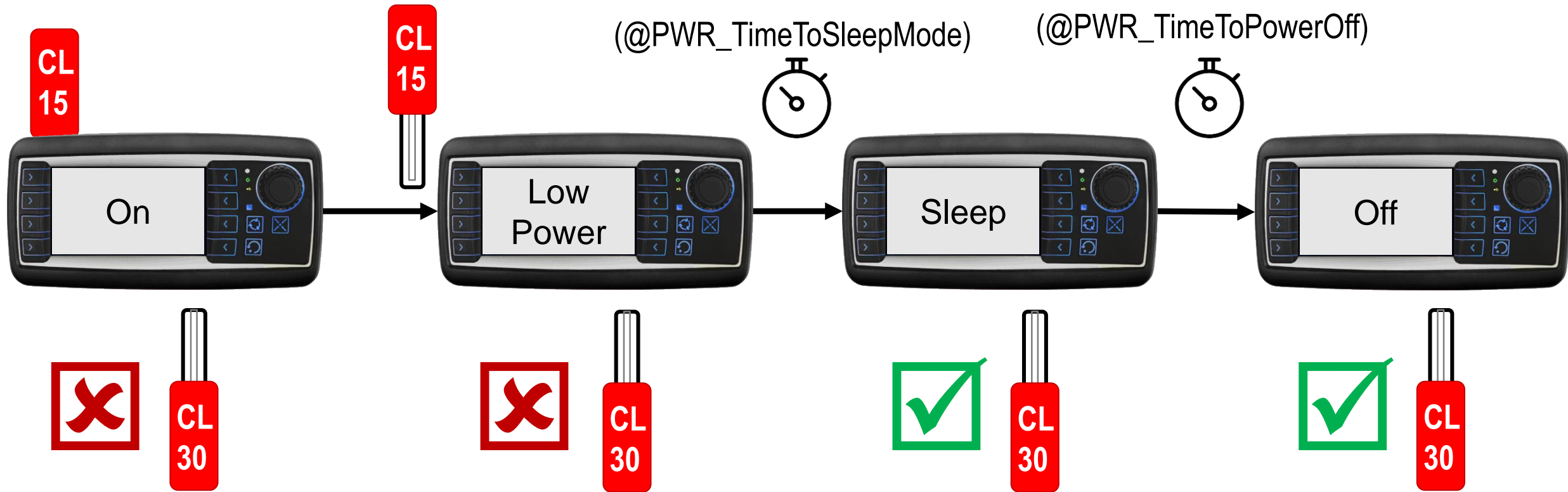
# 7.1. Extended agenda – Power management



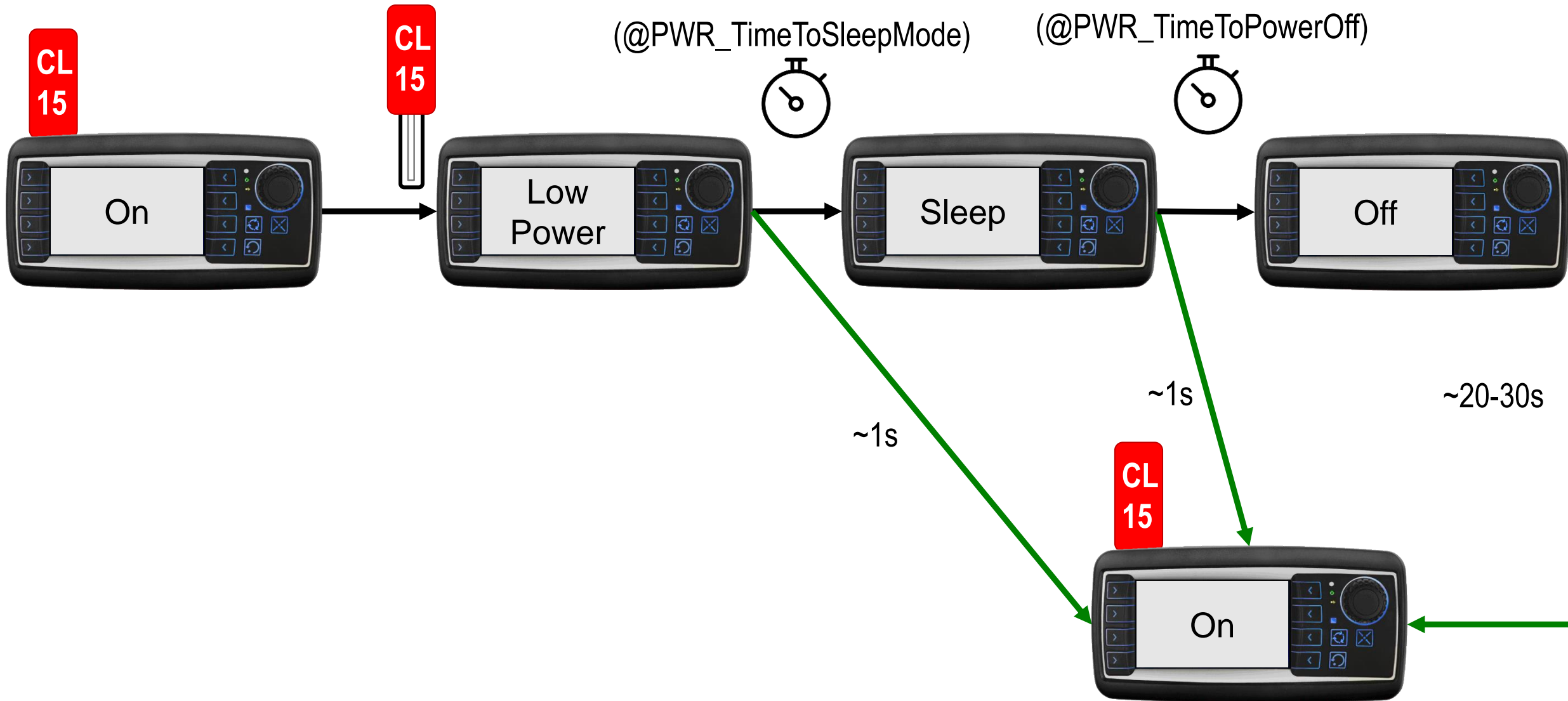
mA @13.5 V

	A3	A6	A6G2	A8
On	430	900	1000	1600
Low Power	160	200	200	300
Sleep	≤ 90	≤ 100	≤ 100	≤ 200
Off	<3	<3	<3	<3

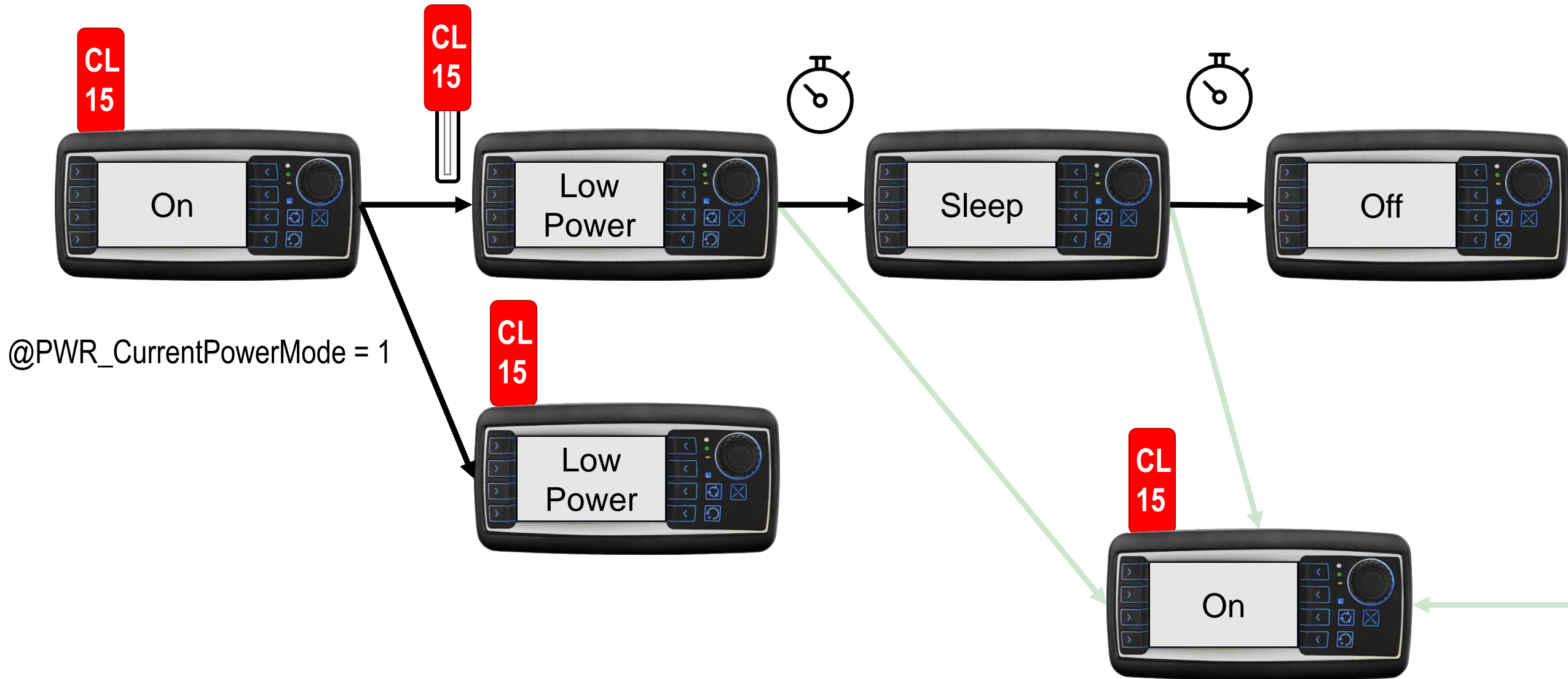
# 7.1. Extended agenda – Power management



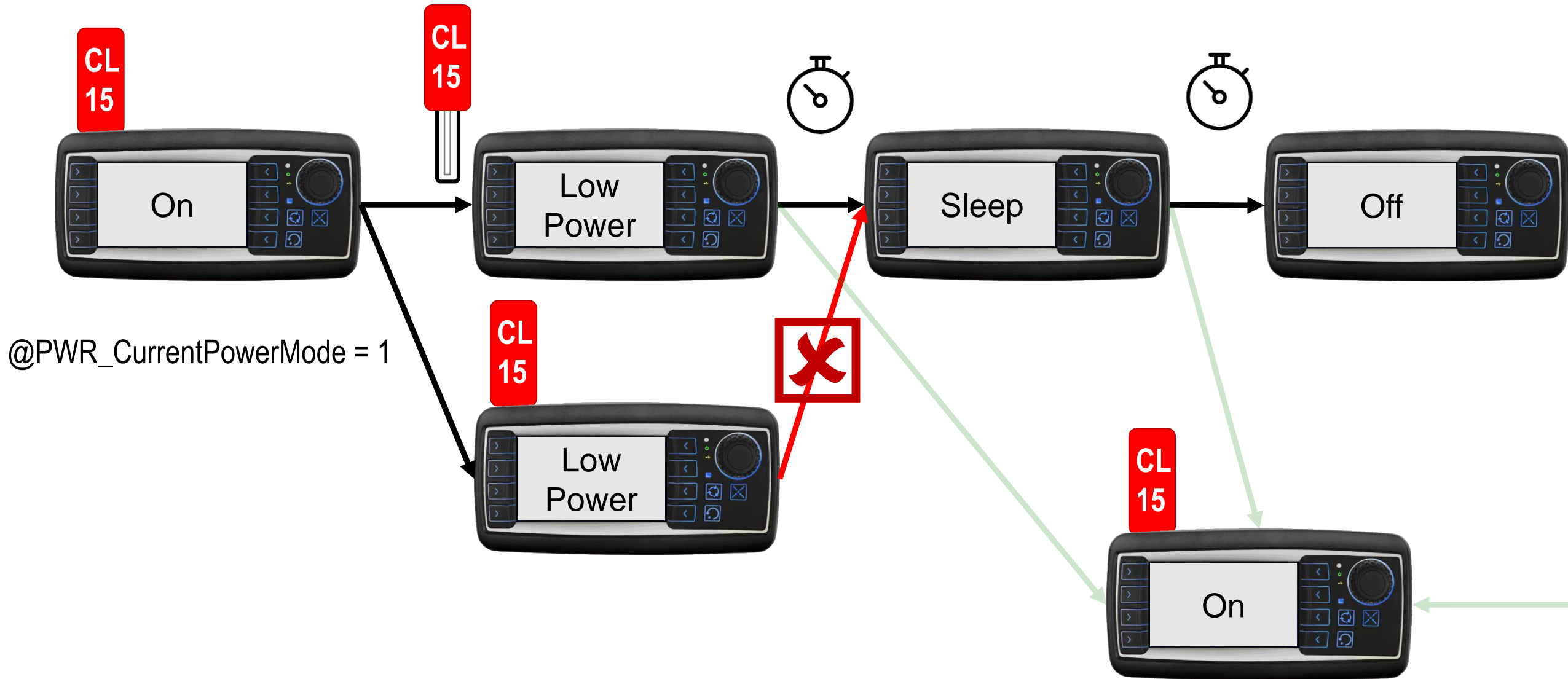
# 7.1. Extended agenda – Power management



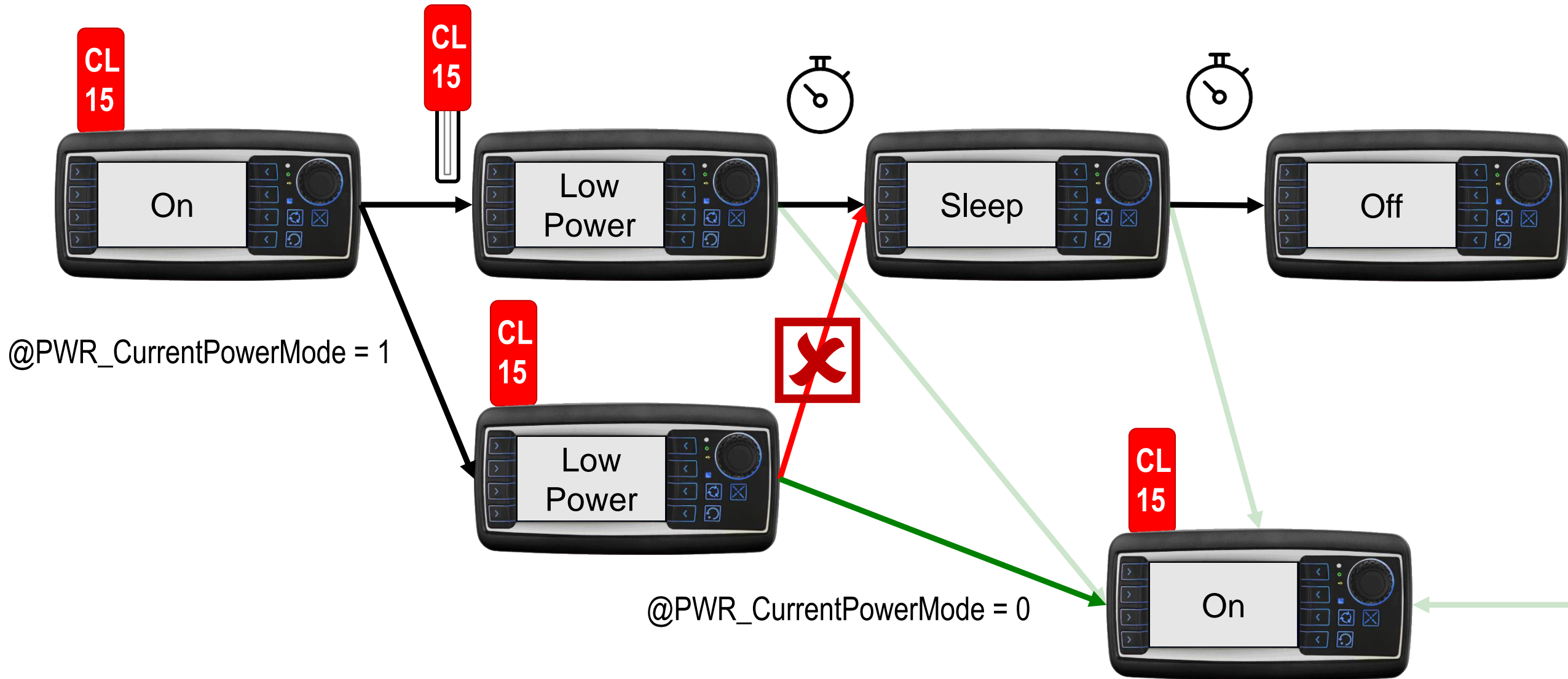
# 7.1. Extended agenda – Power management



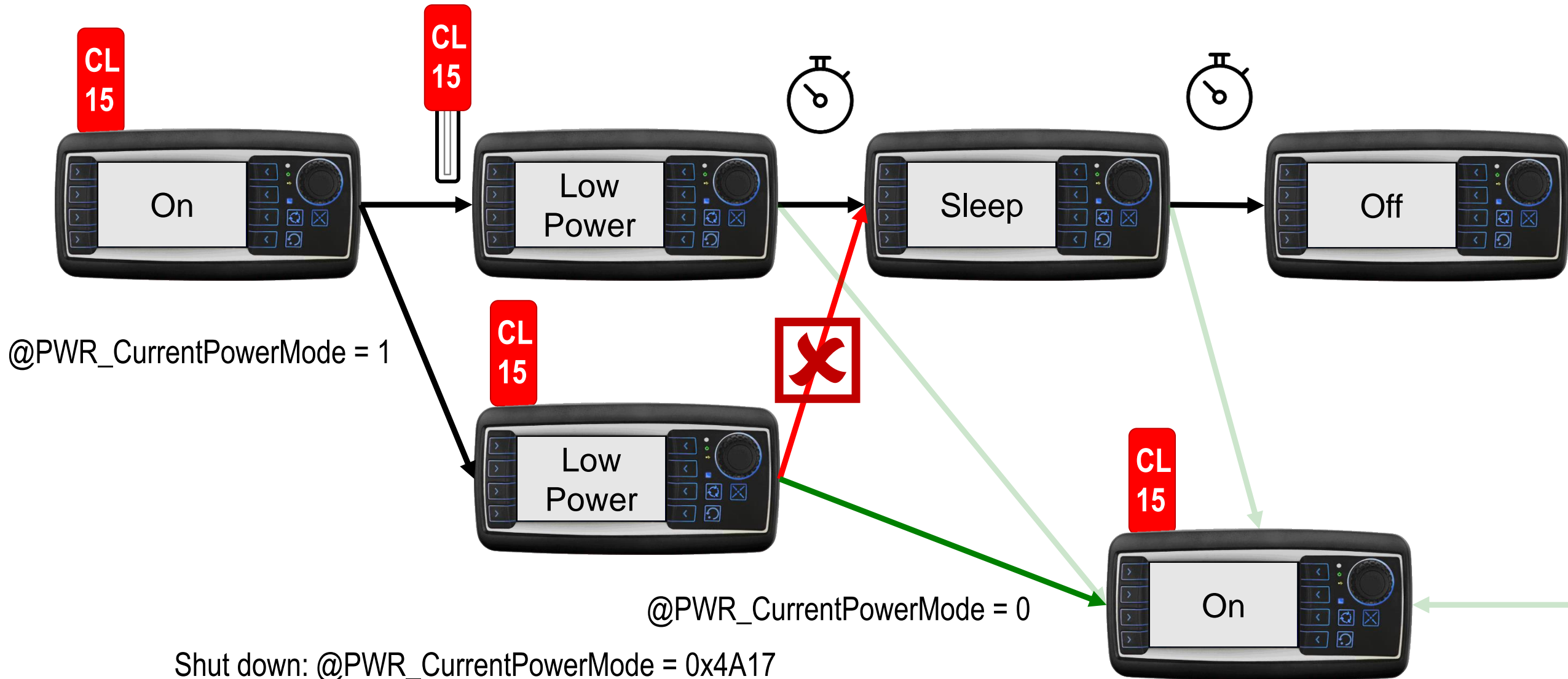
# 7.1. Extended agenda – Power management



# 7.1. Extended agenda – Power management



# 7.1. Extended agenda – Power management



Shut down:  $@PWR\_CurrentPowerMode = 0x4A17$   
 Reboot:  $@PWR\_CurrentPowerMode = 0xb007$

## 7. Extended agenda

1. **Power management**
2. **Visibility**
3. **JavaScript**
4. Alarms
5. Language dependency
6. Variable Logging

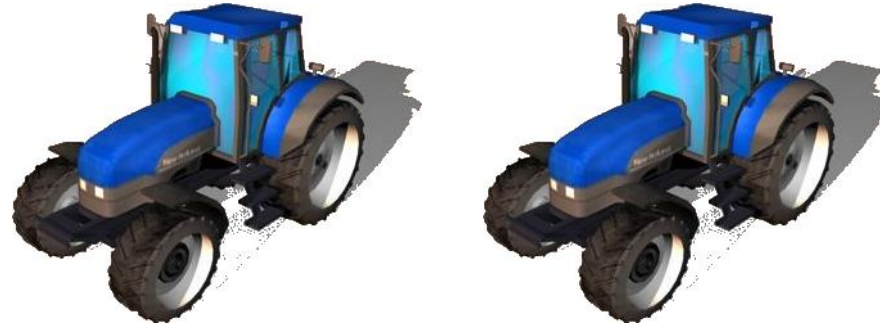


## 7.2. Extended agenda – Visibility



You can use visibility to make any object(s) invisible / visible as you need

## 7.2. Extended agenda – Visibility



You can use visibility to make any object(s) invisible / visible as you need

## 7.2. Extended agenda – Visibility



You can use visibility to make any object(s) invisible / visible as you need

## 7.2. Extended agenda – Visibility



You can use visibility to make any object(s) invisible / visible as you need

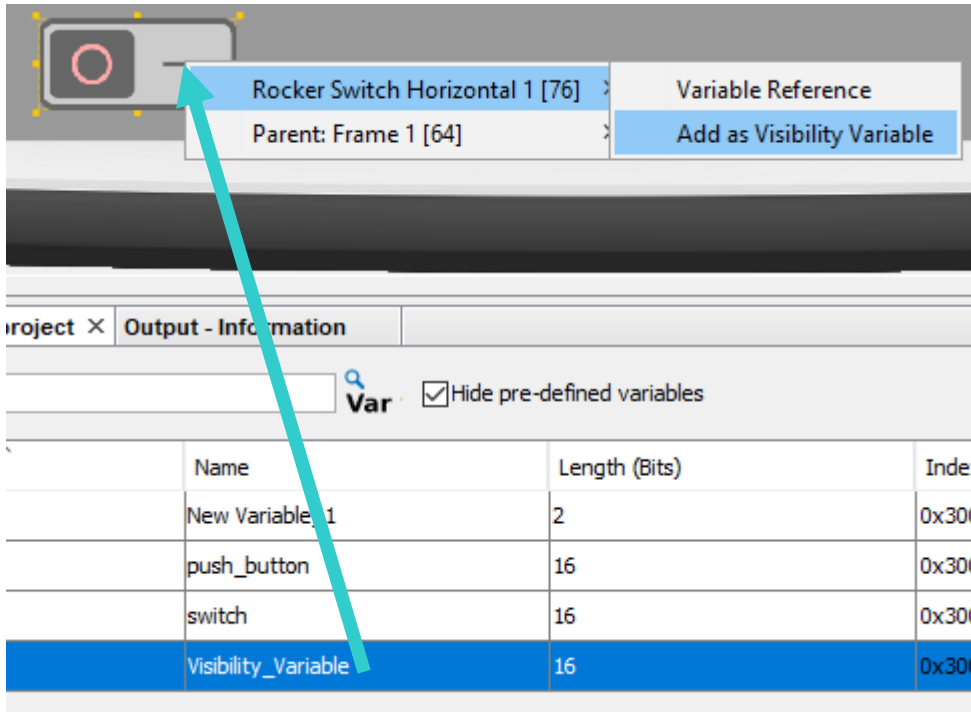
## 7.2. Extended agenda – Visibility



There are 2 ways now to make objects visible / invisible

The first is described on the next slide.  
The second in the next 10 slides – you decide.

## 7.2. Extended agenda – Visibility



Visibility\_Variable == 0



Visibility\_Variable != 0



Use any variable as a visibility variable

Drag & drop the variable on the object and choose *Add as Visibility Variable*

Variable == 0 -> object invisible

Variable != 0 -> object visible

Continue with next chapter

## 7.2. Extended agenda – Visibility



You can use visibility to make any object(s) invisible / visible as you need

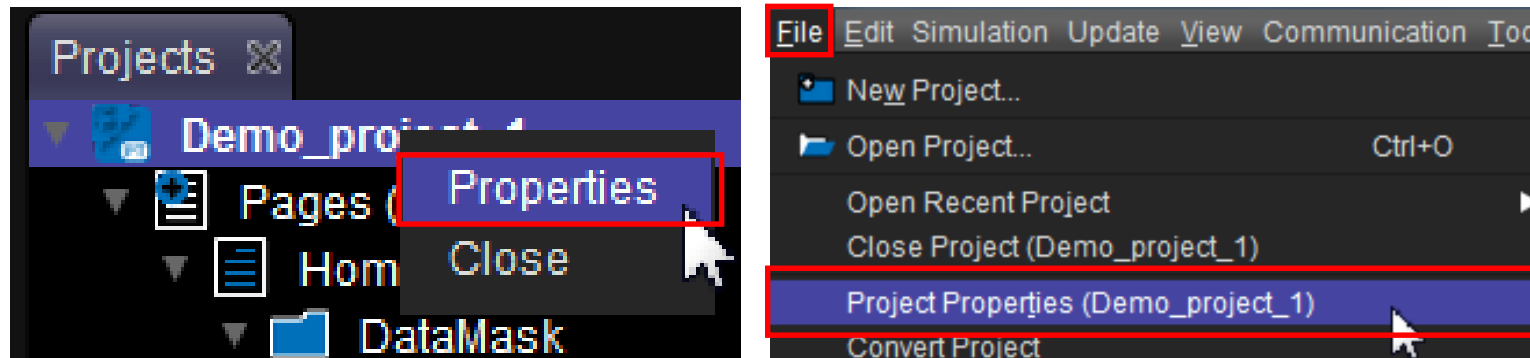
There are 64 visibility variables, @Visibility00 to @Visibility63

Each one of them has 16 bits.

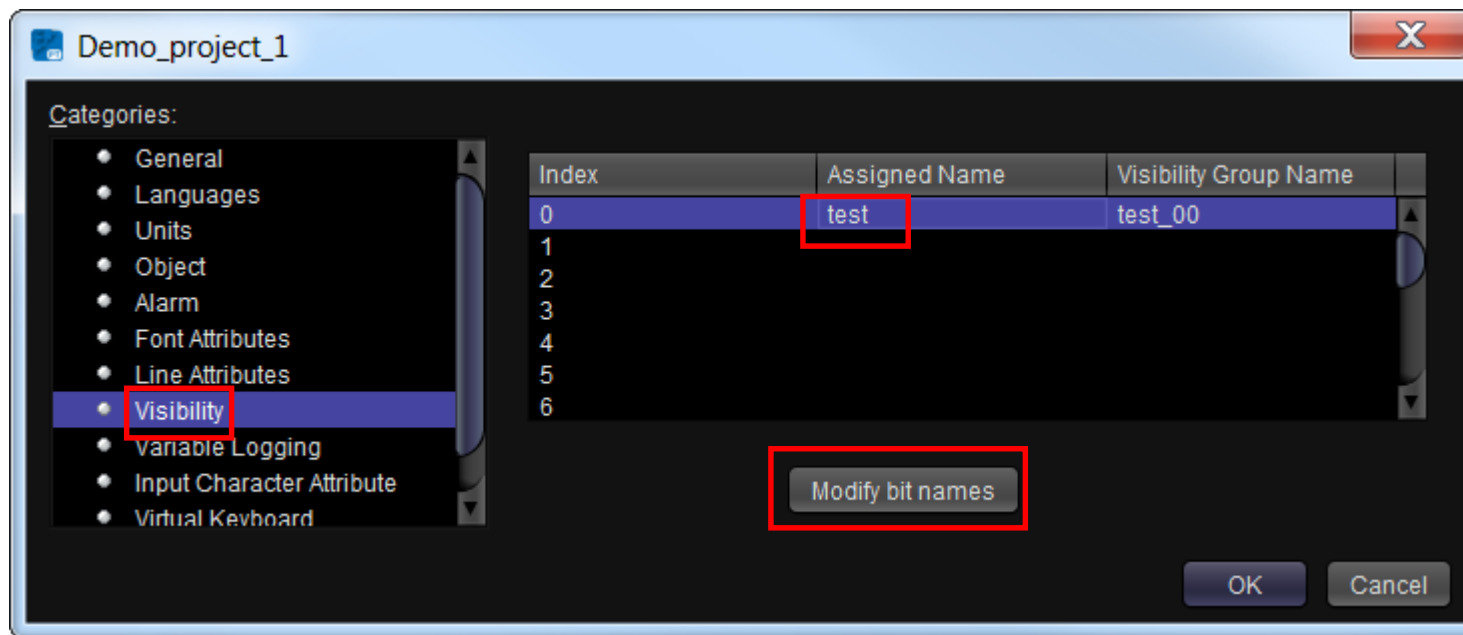
You can use each bit of every variable to switch the visibility of one or several objects

That's 1024 separate visibility switches

## 7.2. Extended agenda – Visibility – Setup



In the project properties, go to the tab *Visibility*

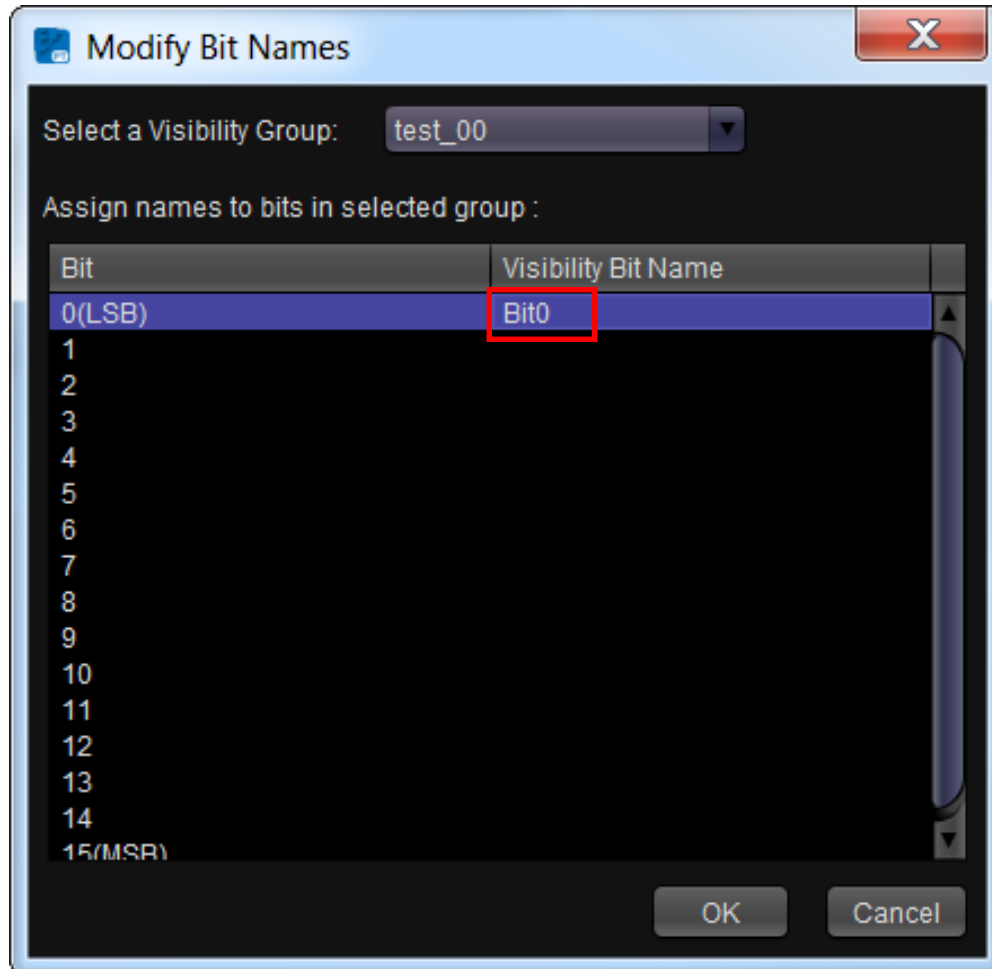


You can enable a visibility variable by giving it an internal *Assigned Name*.

*Click on Modify bit names*



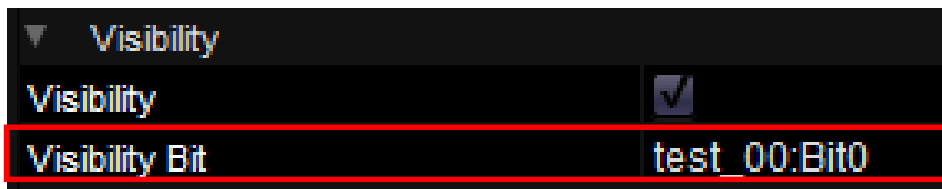
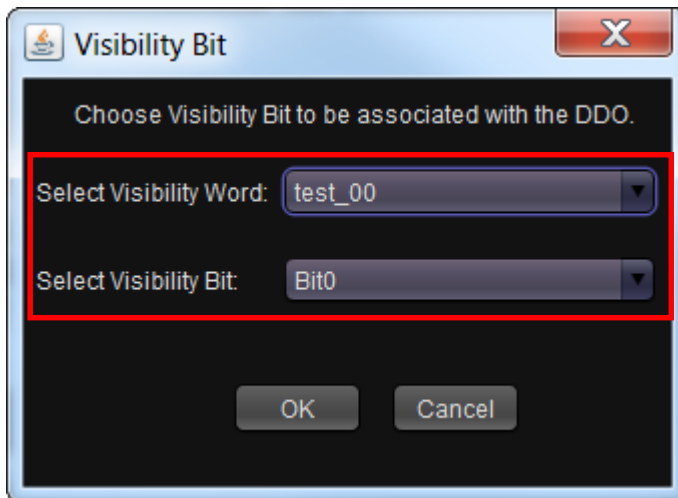
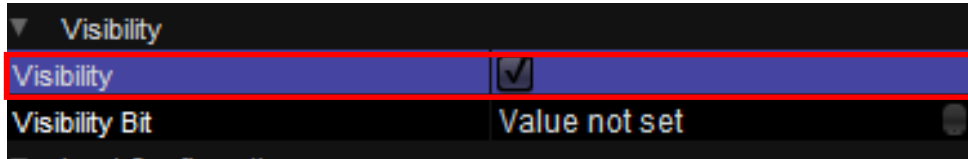
## 7.2. Extended agenda – Visibility – Setup



Press the button *Modify bit names* to give specific names to the single bits of that variable.

These names are only for you so you can connect the objects and the bits easily

## 7.2. Extended agenda – Visibility – Configuring the objects



To use visibility for any object, check the property *Visibility* and choose a visibility bit in the dialog for *Visibility Bit*

If a bit is set to 1, the object is visible, if it is set to 0, the object is invisible.

To set the visibility for those objects, just change the value of the corresponding *@Visibilityxy* variable accordingly

## 7.2. Extended agenda – Visibility – Using the variables

Example: Three objects with bits 0, 1 and 2 of the visibility variable

@Visibility value (decimal)	Visibility of tractor		
	1	2	3
0	-	-	-
1	✓	-	-
2	-	✓	-
3	✓	✓	-
4	-	-	✓
5	✓	-	✓
6	-	✓	✓
7	✓	✓	✓
8	-	-	-
...		...	
15	✓	✓	✓
...		...	
65535	✓	✓	✓

Tractor 1 (Bit 0)



Tractor 2 (Bit 1)



Tractor 3 (Bit 2)



To use visibility for any object, check the property *Visibility* and choose a visibility bit in the dialog for *Visibility Bit*

If a bit is set to 1, the object is visible, if it is set to 0, the object is invisible.

To set the visibility for those objects, just change the value of the corresponding *@Visibilityxy* variable accordingly

-> Start by using only Bit0 and use different variables. This makes switching easy. With several bits used, bit operations are needed to switch correctly

## 7.2. Extended agenda – Visibility – Using the variables

Example: Six tractors with the following visibilities



Bit 0



Bit 1



Bit 2



Bit 3



Bit 4



Bit 5

Bit 0, 2 and 5 are set to 1 initially  
-> Which value does the visibility variable have?

## 7.2. Extended agenda – Visibility – Using the variables

Example: Six tractors with the following visibilities



Bit 0



Bit 1



Bit 2



Bit 3



Bit 4



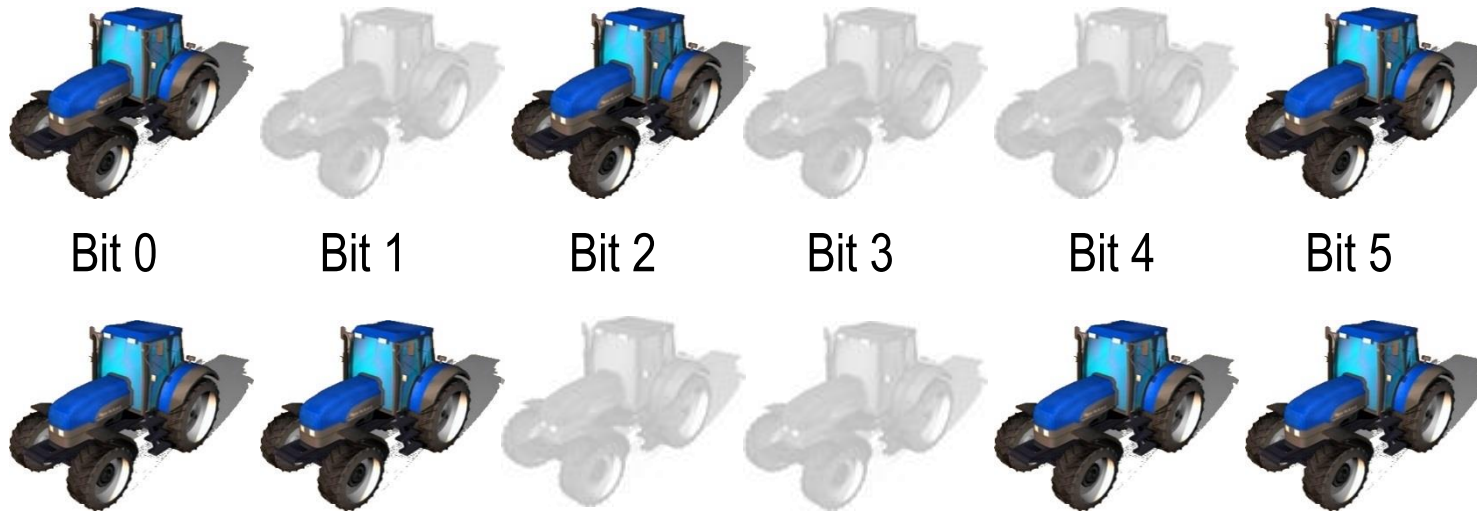
Bit 5

Bit 0, 2 and 5 are set to 1 initially  
-> Which value does the visibility variable have?

37 (in binary 100101)

## 7.2. Extended agenda – Visibility – Using the variables

Example: Six tractors with the following visibilities



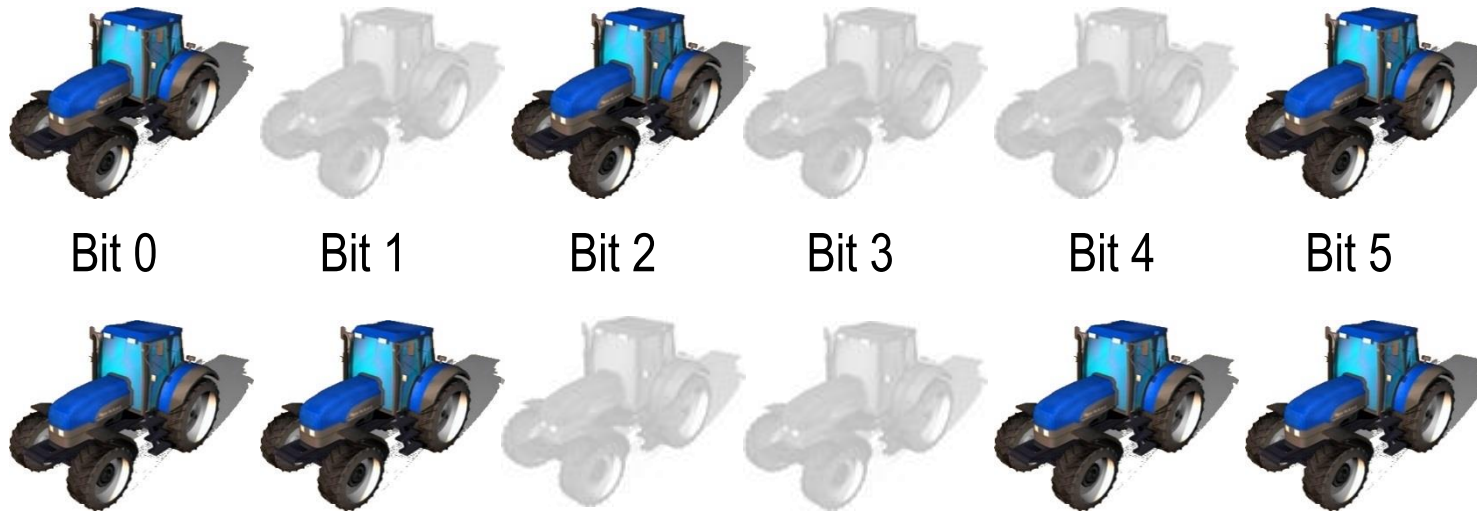
Bit 0, 2 and 5 are set to 1 initially  
-> Which value does the visibility variable have?

37 (in binary 100101)

-> Which value must the variable have if I want to make it look like this?

## 7.2. Extended agenda – Visibility – Using the variables

Example: Six tractors with the following visibilities



Bit 0, 2 and 5 are set to 1 initially  
-> Which value does the visibility variable have?

37 (in binary 100101)

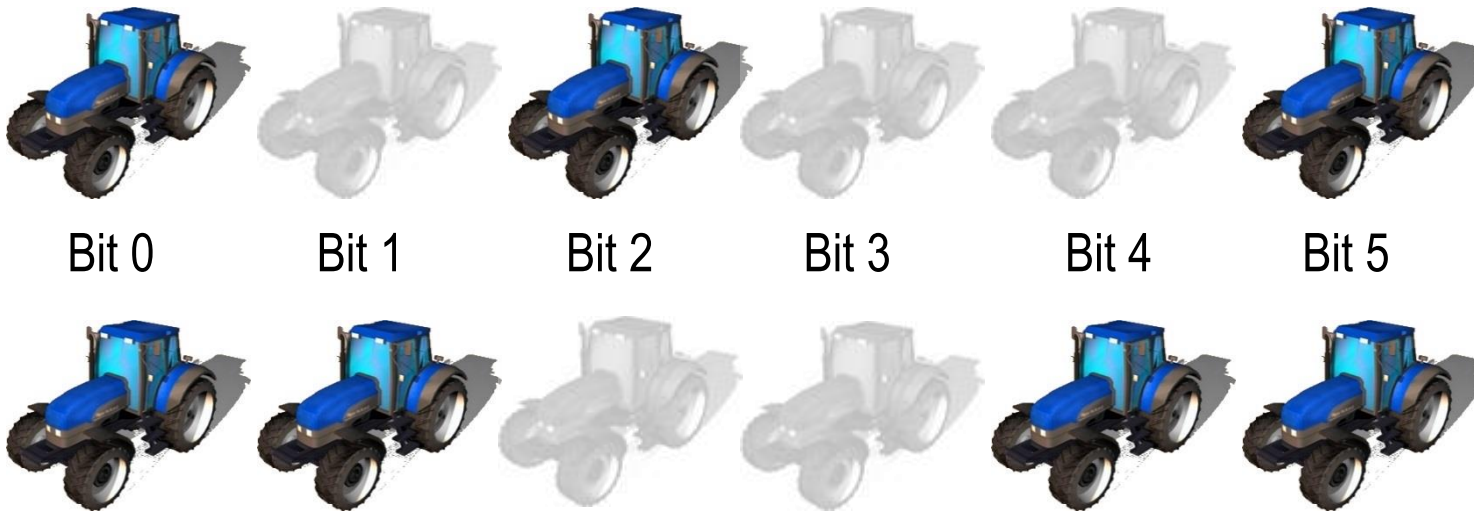
-> Which value must the variable have if I want to make it look like this?

51 (in binary 110011)

-> What does the JavaScript code look like to make this change?

## 7.2. Extended agenda – Visibility – Using the variables

Example: Six tractors with the following visibilities



```
vis = vis | 2;
vis = vis & ~ 4;
vis = vis | 16;
```

Bit 0, 2 and 5 are set to 1 initially  
-> Which value does the visibility variable have?

37 (in binary 100101)

-> Which value must the variable have if I want to make it look like this?

51 (in binary 110011)

-> What does the JavaScript code look like to make this change?

-> Stick with only Bit 0 in the beginning



## 7. Extended agenda

1. **Power management**
2. **Visibility**
3. **JavaScript**
4. Alarms
5. Language dependency
6. Variable Logging

## 7.3. Extended agenda – JavaScript – Why?

Why JavaScripts?

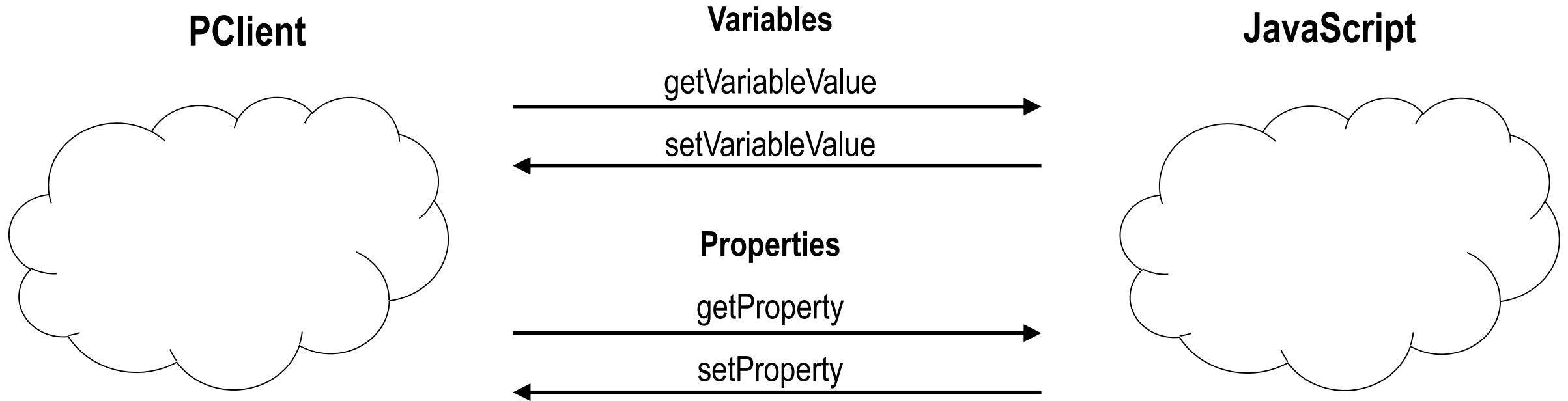
- value calculations (that extend the scaling operations)
- combination of several functions in one event
- change object properties
- logic programming

## 7.3. Extended agenda – JavaScript – Events

Events that can trigger JavaScripts

- Project initialization
- Button / soft key actions
- Variable value changes
- cyclic execution (OnProjektRepeat / OnPageRepeat)

## 7.3. Extended agenda – JavaScript – Variables



## 7.3. Extended agenda – JavaScript – Infos and tips

JavaScripts offer many possibilities, but:

- JavaScripts don't work in real-time (max. every 100 ms)
- Every JavaScript uses resources -> As few scripts as possible, as short as possible
- Put the right code to the right event (if you want to react on a variable value change, don't use a cyclic script, use the event `OnValueChangeByOwner`)
- Don't combine commands (not `setVariableValue("test",(getVariableValue("test")+1);)`)
- Variables from the Hardware Daemon are not refreshed instantly!  
-> `setVariableValue("@DispBacklightIntensity",new_value);` and then `getVariableValue("@DispBacklightIntensity");` in the same script will return the old value!

## 7.3. Extended agenda – JavaScript – More infos and tips

- There is no “time” and no “waiting” within one JavaScript! Everything will be executed instantly

```
for (var x = 0; x == 100; x++)  
{  
    setVariableValue (“test”,x);  
}
```

will set the PClient variable “test” not to 0, then 1, then 2..., but only directly to 100.

Check out our examples to get some ideas

For a detailed syntax description:

Our HTML help for functions specific to our device, and <http://www.w3schools.com/js/> for general syntax

## 7.3. Extended agenda – JavaScript – First program

We want to create the following functionality:

Use the two buttons and create “plus 5” and “minus 5” functionality for the display backlight

## 7.3. Extended agenda – JavaScript – Example 1

We want to create the following functionality:

When Digital Input 1 is active, the LED shall glow green, when it is inactive it shall not glow at all

When the Digital Input 1 is active for the first time, the project should jump from the Homepage to Page 1



## 7.3. Extended agenda – JavaScript – Example 1

We want to create the following functionality:

When Digital Input 1 is active, the LED shall glow green, when it is inactive it shall not glow at all

Some tips:

- Use the event `OnValueChangeByOwner` of the variable `@DigitalInput01`
- Use the `getVariableValue` and `setVariableValue` functions to read and write the PClient variables
- The LED is controlled by the variable `@LED_Multicolor`
- You only need one script for this example
- Green is `0x00FF00`
- If statements need to be put in brackets -> `if (<statement>)`

Solution: check the project `Example_JavaScript_Lesson_1`

## 7.3. Extended agenda – JavaScript – Example 2

We want to extend the functionality of the first example:

When Digital Input 1 is active, the LED shall glow green.

When Digital Input 2 is active, it shall glow red.

When both are active, it shall glow yellow.

When the Digital Input 1 is active for the first time, the project should jump from the Homepage to Page 1

The list object with the traffic light shall switch according to the LED color (DI1 -> green light, DI2 -> red light, DI1 + 2 -> yellow light).

If no input is connected, the list object shall be made invisible with the use of visibility. This should also be the initial state when the project starts

## 7.3. Extended agenda – JavaScript – Example 2

When Digital Input 1 is active, the LED shall glow green.

When Digital Input 2 is active, it shall glow red.

When both are active, it shall glow yellow.

When the DI1 is active for the first time, the project should jump from the Homepage to Page 1

The list object with the traffic light shall switch according to the LED color

(DI1 -> green light, DI2 -> red light, DI1 + 2 -> yellow light).

If no input is connected, the list object shall be made invisible with the use of visibility. This should also be the initial state when the project starts

Hints:

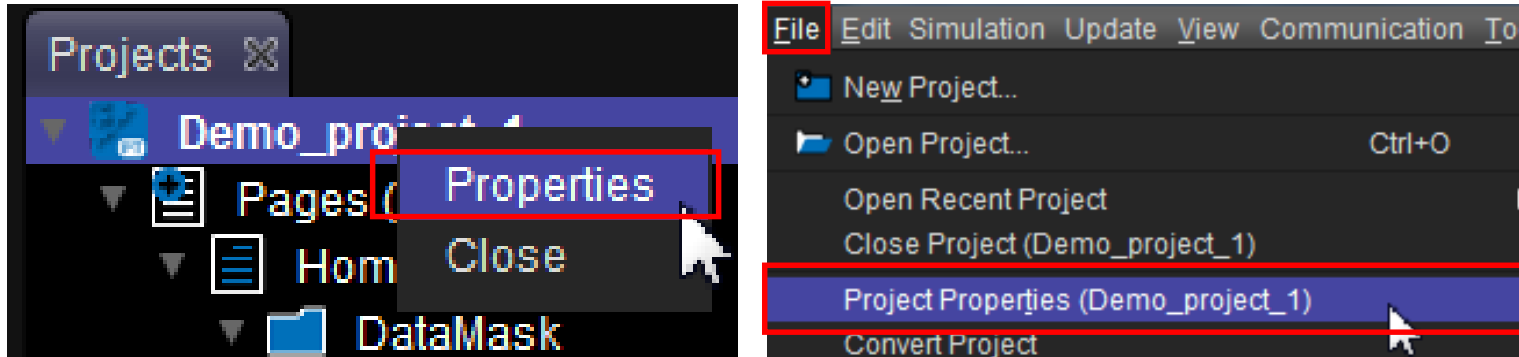
- you have two events to use now, but try using still only one script
- check if you have considered all possible cases
- the initial setup at project start can be programmed with the event OnProjektInit
- remember to set up the visibility variable before using it in the project properties
- Red is 0xFF0000, Green is 0x00FF00 and Yellow is 0xFFFF00

Solution: check the project `Example_JavaScript_Lesson_2`

## 7. Extended agenda

1. **Power management**
2. **Visibility**
3. **JavaScript**
4. Alarms
5. Language dependency
6. Variable Logging

## 7.4. Extended agenda – Alarms – Alarm templates



Alarms are pages or smaller popup frames that can be opened and closed by setting a single variable.

First the alarm templates can be configured in the project properties

Alarm types

Priority	Alarm	Type	Timeout	Behaviour	Mini...	X	Y	Width	Height	Sound p...	...
1	Emergency	Page	0	Emergency	-1	0	0	480	272	e739c	
2	Med	Page	0	Emergency	-1	0	0	480	272	f73dd	
3	Low	Page	0	Emergency	-1	0	0	480	272	c7298	
4	Warning	Frame	0	Warning	0	90	36	300	200		
5	Informational	Frame	0	Informational	0	90	36	300	200		

Type *Page* means a full screen alarm, with *Frame* you can set size and position of the alarm popup in the X, Y, Width and Height column

## 7.4. Extended agenda – Alarms – Alarm templates

Alarm types

Priority	Alarm	Type	Timeout	Behaviour	Mini...	X	Y	Width	Height	Sound p...	...
1	Emergency	Page	0	Emergency	-1	0	0	480	272	e739c	<input type="checkbox"/>
2	Med	Page	0	Emergency	-1	0	0	480	272	f73dd	<input type="checkbox"/>
3	Low	Page	0	Emergency	-1	0	0	480	272	c7298	<input type="checkbox"/>
4	Warning	Frame	0	Warning	0	90	36	300	200		<input type="checkbox"/>
5	Informational	Frame	0	Informational	0	90	36	300	200		<input type="checkbox"/>

The *Behaviour* determines how the alarm can be closed:

*Emergency:*

- by setting the alarm variable

*Warning:*

- by setting the alarm variable

- by pressing the Escape key

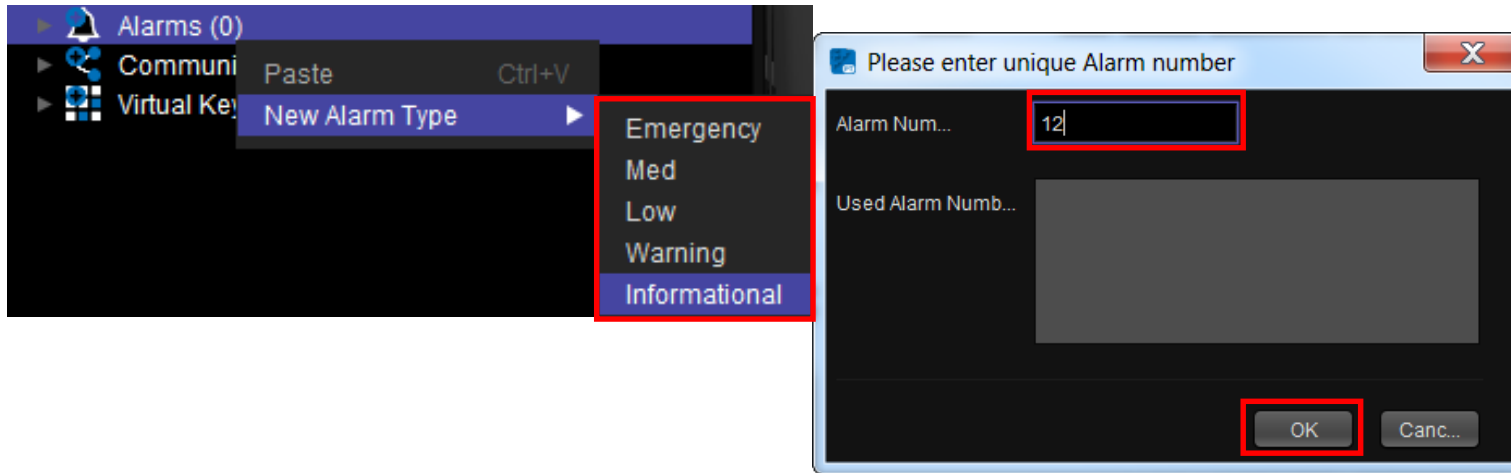
*Informational:*

- by setting the alarm variable

- by pressing the Escape key

- automatically after a timeout

## 7.4. Extended agenda – Alarms – A new alarm



To create an alarm, right click on *Alarms (x)* and select *New Alarm Type* and then the alarm template of your choice

In the dialog enter a unique alarm number. This will be used to open/close the alarm.

The actual alarm page can be configured just as a normal project page with all the objects.

The blue area is the alarm area, the white area will be transparent, i.e. you see the page below it

## 7.4. Extended agenda – Alarms – Alarm variable

Event Actions

No Action

Select Action: Set Value

Options

Properties

Variable	@AlarmShow
Value	0x800C

Event Actions

No Action

Select Action: Set Value

Options

Properties

Variable	@AlarmShow
Value	0xC

To show the alarm with the number 12,  
set the variable  
`@AlarmShow` to `0x800C` (or 32780 decimal)

To remove the alarm with the number 12,  
set the variable  
`@AlarmShow` to `0x000C` (or 12 decimal)

-> To set an alarm, set the variable to  
`0x8000 || alarm number` (in hexadecimal)

-> To remove an alarm, set the variable to  
just the alarm number (in hexadecimal)



## 7.4. Extended agenda – Alarms – Setting the alarm

```
Script Content:
var value = getVariableValue("mph");
if (value > 100)
{
    setVariableValue("@AlarmShow", 0x800C;
}
else
{
    setVariableValue("@AlarmShow", 0xC;
```

Alarms can be set

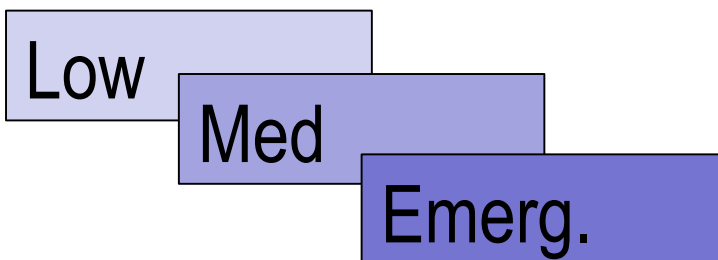
- over CAN (put the variable `@AlarmShow` into a mapping)
- over JavaScript (by setting the variable)
- with a button / soft key action (Set Value)
- with any other event

## 7.4. Extended agenda – Alarms – Alarm priorities

Alarm types

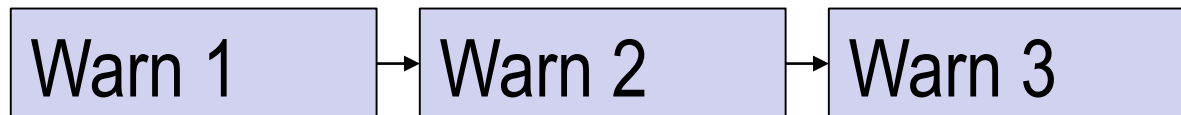
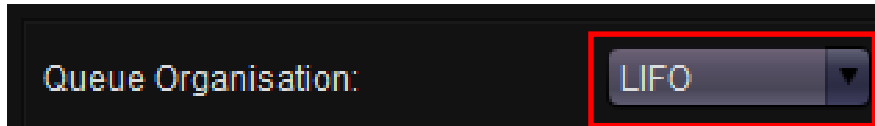
Priority	Alarm	Type	Timeout	Behaviour	Mini...	X	Y	Width	Height	Sound p...	...
1	Emergency	Page	0	Emergency	-1	0	0	480	272	e739c	
2	Med	Page	0	Emergency	-1	0	0	480	272	f73dd	
3	Low	Page	0	Emergency	-1	0	0	480	272	c7298	
4	Warning	Frame	0	Warning	0	90	36	300	200		
5	Informational	Frame	0	Informational	0	90	36	300	200		

Alarm types also have a priority



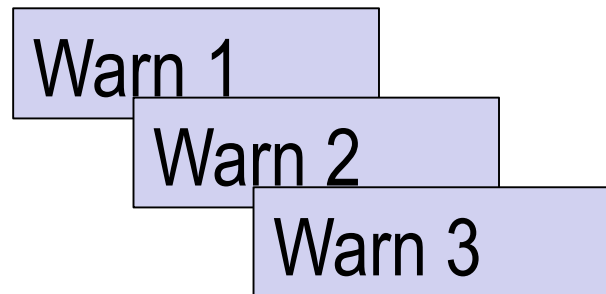
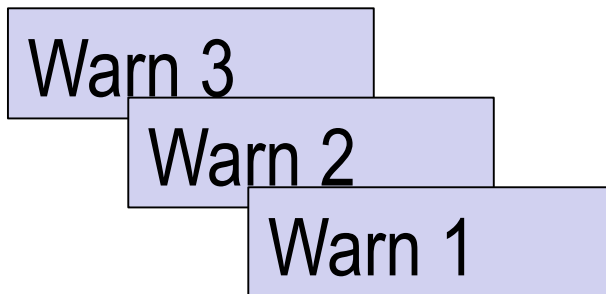
This determines which alarm is shown on top when more than 1 are active

## 7.4. Extended agenda – Alarms – Alarm priorities



**FIFO**

**LIFO**

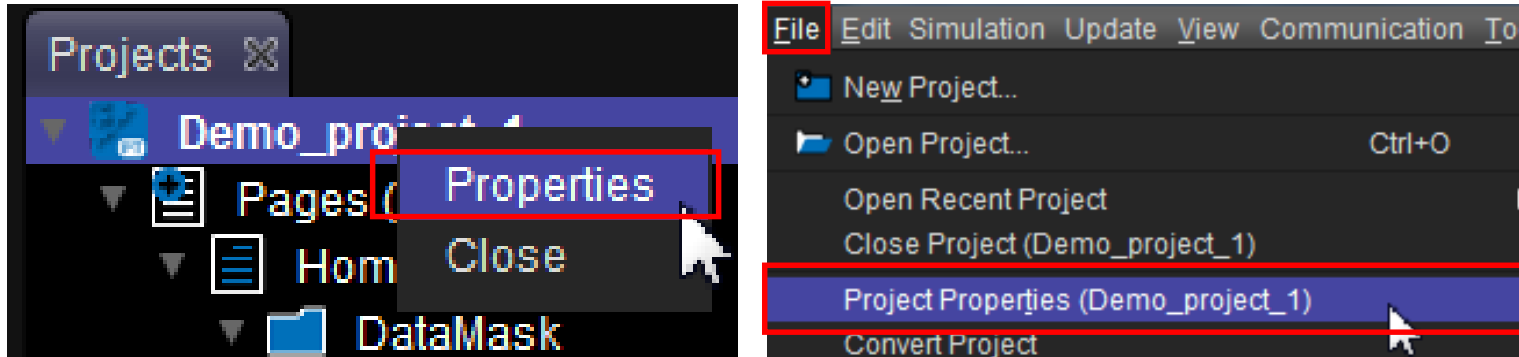


For alarms with the same priority the behavior can be set to LIFO (last in – first out) or FIFO (first in – first out)

## 7. Extended agenda

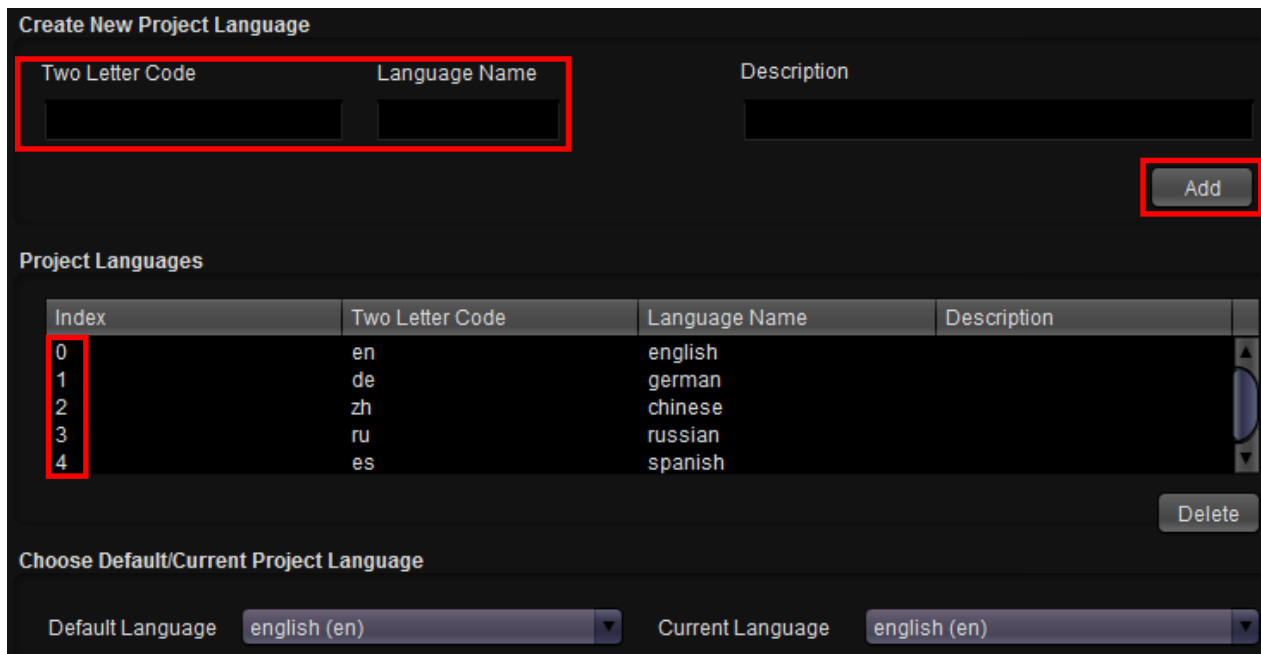
1. **Power management**
2. **Visibility**
3. **JavaScript**
4. Alarms
5. Language dependency
6. Variable Logging

## 7.5. Extended agenda – Language dependency – Setup



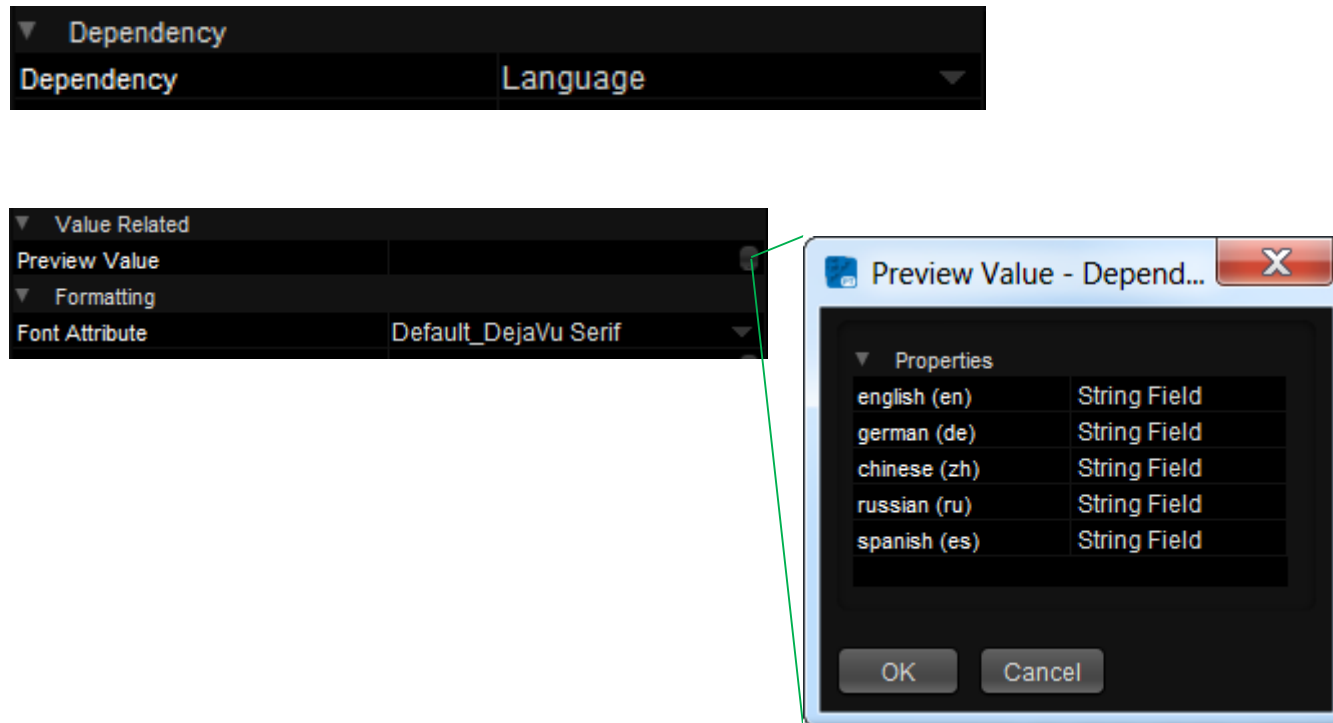
You can add new languages in the project properties in the tab *Languages*

Create a new language by writing a two letter code and a language name, then click add.



The most important element for changing the language in the project is the index

## 7.5. Extended agenda – Language dependency – Objects



To prepare objects for language dependency, set their property *Dependency* to *Language*

For the string fields, the text (*Preview Value*) can be changed manually for all the languages. You can also change the alignment and the font file for each language

## 7.5. Extended agenda – Language dependency – Import / export

The screenshot shows the Projektor Tool interface. The 'Tools' menu is open, highlighting 'Import Export Language CSV'. Below it, the 'Select Projekt Language to export the C...' dialog is shown with 'Source Language' set to 'english (en)' and 'Target Language' set to 'german (de)'. The 'Export' button is highlighted. Below that, the 'Import Language CSV File' dialog is shown with the 'CSV File Path' set to 'C:\Users\lcst\Desktop\test.csv' and the 'Preview' button highlighted. The 'Preview' table is as follows:

Object ID	Object Name	Property Name	Source Lang...	Target Langu...	Max Length	Default Text	Translated Text
88	String Field 1	Preview Value	en	de	255	String Field	German Text

The 'Target Language' is set to 'german' and the 'Import' button is highlighted.

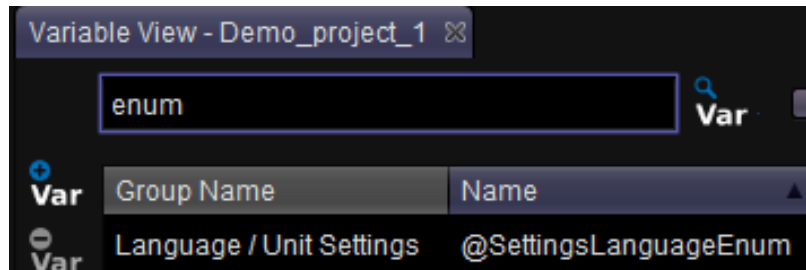
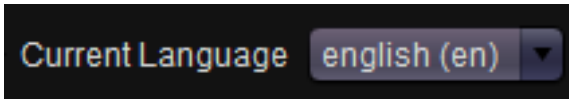
You can export a table for each source – target language pair into a csv file to translate everything at once

Go to *Tools* -> *Import Export Language CSV*

Select a source and a target language and the file will be created.

Once the translation is done, import the file back into the Projektor Tool. You can preview the changes, then click on *Import* to import the texts.

## 7.5. Extended agenda – Language dependency – Changing the language



For a preview you can change the language in the toolbar in the Projektor Tool

To change the language on the device, set the variable `@SettingsLanguageEnum` to the index number of the desired language

The current language can be set

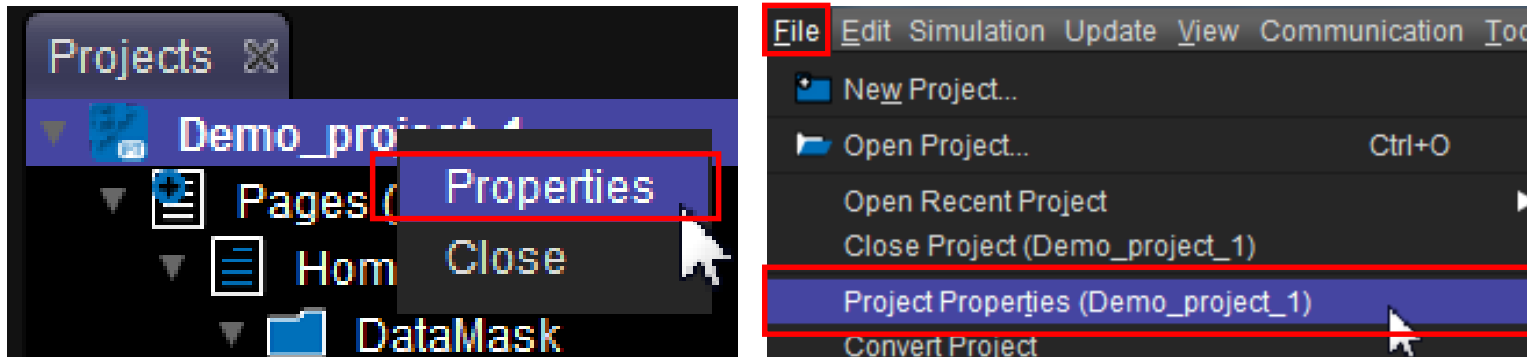
- over CAN (put the variable `@SettingsLanguageEnum` into a mapping)
- over JavaScript (by setting the variable)
- with a button / soft key action (Set Value)
- with any other event



## 7. Extended agenda

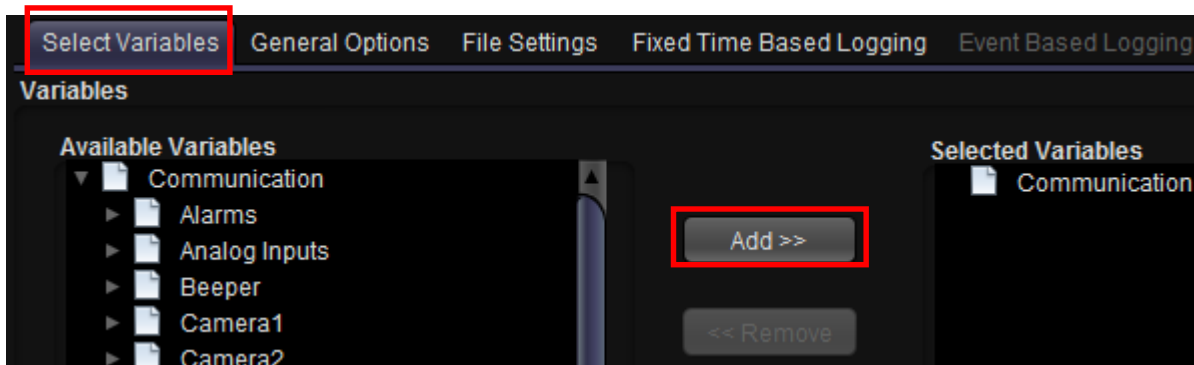
1. **Power management**
2. **Visibility**
3. **JavaScript**
4. Alarms
5. Language dependency
6. Variable Logging

## 7.6. Extended agenda – Variable logging



You can log the values of any variable on the file system of the device

Setup variable logging in the project properties in the tab *Variable Logging*



In the first tab you can select any variable that you want to log

To add a variable for logging, select it in the list on the left and click *Add>>*

## 7.6. Extended agenda – Variable logging

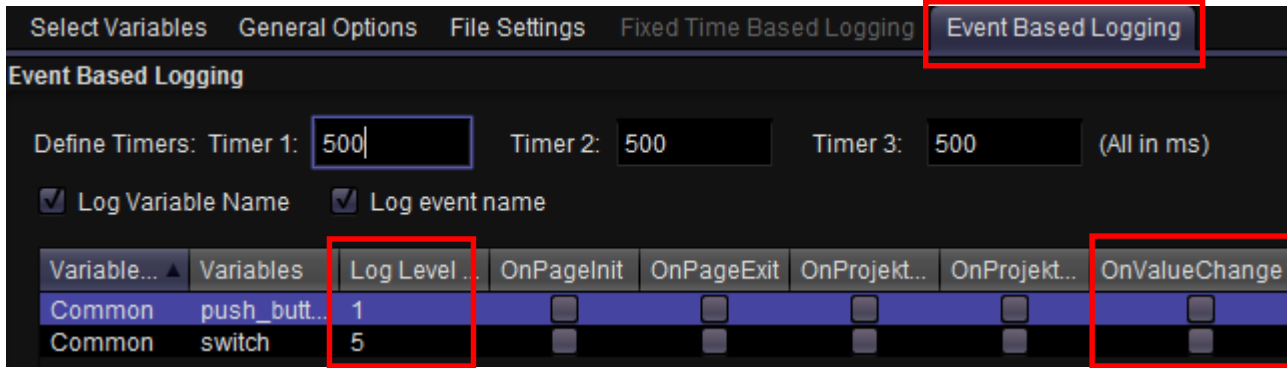
The screenshot shows the 'General Options' tab selected. The 'Event Based Logging' radio button is highlighted with a red box. The interface includes a text input for the log file name (data\_log\_file.log), a 'Show Log File Example' button, a date-time format input (<YYYY>-<MM>-<DD>\_<hh>-<mm>-<ss>.<f>), a 'Define Log File Header' button, a 'Delimiter Character' dropdown (set to ';'), and a 'Reset Line Prefix To Default' button. Below these are two radio buttons: 'Fixed Time Based Logging' and 'Event Based Logging'.

The screenshot shows the 'File Settings' tab selected. The 'Continuous Logging' radio button is highlighted with a red box. The interface includes a 'Please choose:' section with two radio buttons: 'Continuous Logging' and 'New File at restart'. Below this are three input fields: 'Maximum File Size' (1000 KiloByte), 'Number Of Files' (10), and 'Current Disk Space Usage' (2000 Kilobyte). Each input field has a corresponding unit and range description.

In the general options, select *Event Based Logging*

In the file settings you can switch between only one backup file (*Continuous Logging*) or 4 to 50 backup files. In both cases you have a ringbuffer functionality

## 7.6. Extended agenda – Variable logging



Example: Three variables with set up log level 1,5 and 9

Project log level	Variables will be logged		
	1	5	9
1	✓	-	-
2,3	✓	-	-
4	✓	-	-
5	✓	✓	-
6,7	✓	✓	-
8	✓	✓	-
9	✓	✓	✓

In the last tab you can select, which variable should be logged when.

The best event to choose is *OnValueChanged*

You can use the log level to set different verbose levels

Set the variable `@LogLevel` in the project to decide which variables should be logged.

To copy/move the log files on a USB stick, set the variable `@LogCopyAllNow` to  $\frac{1}{2}$  (also see example project)

## 7. Extended agenda – Recap

We have learned how to:

- use power management to save battery power
- use visibility to manage object visibility
- use JavaScripts and when
- much more

## Agenda

1. Introduction
2. Installation of the Projektor Tool
3. Updating your device
4. Getting to know the Projektor Tool
5. First project
6. CAN communication
7. Extended agenda
8. FAQ

Thank you for your attention –  
last call for questions!\*

\*but we're still there if you think of them  
later (which you will)

## 8. FAQ

1. Can I integrate the project into the PClient update?
2. Which variables can I change how?
3. How can I use the beeper?
4. How can I change the boot logo?
5. What can I do with the serial console
6. Why are the loading times of my project increasing over time?
7. How can I use the virtual keyboard?

## 8.1 FAQ – Can I integrate the project into the PClient update?

Yes.

Save your project, then go to your project folder, there is a folder called `\terminal_files`.

Now open the application update file `user.tar.gz` with a program like 7zip. In it, browse to the folder `user.tar.gz\user.tar\pclient\projekte\default_prj\`

Delete the folder `\terminal_files` in the `user.tar.gz` file and drag & drop your project folder into the file at the above location.

Remember that you need to re-create the checksum file for the `user.tar.gz`



## 8.2 FAQ – Which variables can I change how?

Any variable in the project that isn't configured as read-only, both pre-defined and user created, can be changed

- by using the action Set Value
- through user input, by connecting the variable to an editable object
- over CAN
- within a JavaScript file

## 8.3 FAQ – How can I use the beeper?

The beeper can be used with alarms and on touch / keypress events or with variables.  
Please check the example project for beeper to see examples.

## 8.4 FAQ – How can I change the boot logo?

Please read the  
OPUS devices update manual for  
step-by-step instructions

## 8.5 FAQ – What can I do with the serial console?

- df – list mounted devices (check if USB stick is mounted)
- top – executes a process manager (check if Pclient is running)
- dmesg – lists all diagnostic messages from the system
- ls -la – list files and folders in the current folder
- cd – change directory
- cp – copy a file (-rf to also copy folders)
- mv – move a file (-rf to also move folders)
- ifconfig – check Ethernet port configuration
- reboot – performs a restart of the device
- /opt/etc/init.d/02\_pclient start / stop / restart – starts, stops or restarts the pclient

Please also check the document [Console\\_Commands.pdf](#)

Preinstalled BusyBox with lots of standard Linux commands

## 8.6 FAQ – Why are the loading times of my project increasing over time?

The file system (JFFS2) runs something like checkdisk sometimes to ensure all files are in good condition.

With an increasing amount of file operations (like installing a lot of new project versions) this will happen more often, and especially at startup. This can slow down the loading process significantly.

The solution would be a clean reinstallation of the PCLient, as then the /opt partition will be deleted and created freshly.

## 8.7 FAQ – How can I use the virtual keyboard?

The screenshot shows a project tree on the left with 'Virtual Keyboards' highlighted. A context menu is open over it, showing options: 'Create VKB Layout', 'Layered Keyboard', 'Layered Portrait Keyboard', and 'Numeric Keyboard'. Two arrows point from the 'Layered Keyboard' and 'Numeric Keyboard' options to two property windows below.

**Numeric Field 1 [33] - Properties**

Properties	Events
Input Configuration	
Set As Input	<input checked="" type="checkbox"/>
Connect To Encoder	<input type="checkbox"/>
Send Value Directly	<input checked="" type="checkbox"/>
Encoder Movement	Linear
Value Change Factor	1000
Enabled	<input checked="" type="checkbox"/>
Inplace Editor	Virtual Keyboard

**String Field 1 [34] - Properties**

Properties	Events
Set As Input	<input checked="" type="checkbox"/>
Connect To Encoder	<input type="checkbox"/>
Input Character Attribute	None
Inplace Editor	Virtual Keyboard
Value Related	Encoder
Preview Value	Virtual Keyboard

Right-click on *Virtual Keyboards* in the project tree, choose *Layered* for text, *Numeric* for numbers

In the object properties, activate *Set As Input*, then choose *Virtual Keyboard* as *Inplace Editor*

Now when the object is edited, the virtual keyboard will pop up

That's it!

If you're still reading, you are cordially invited to read on in the OPUS Projektor manual. It contains a tutorial, an F.A.Q. and information about almost everything.

You can access the help by pressing F1 or going to the menu Help -> Help