

Welcome to the training of the Opus Projector Tool

PROJEKTOR TOOL TRAINING

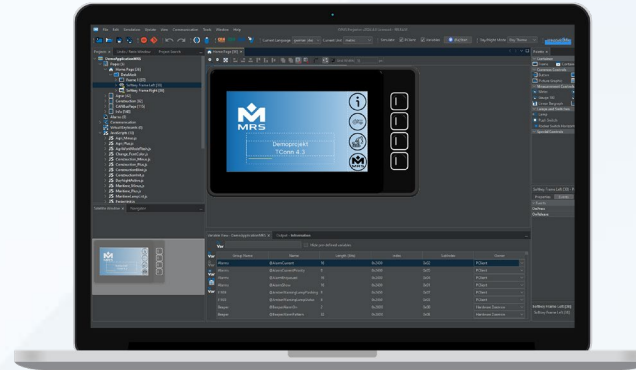
Table of contents

- Introduction
- Projector tool overview
- First project
- CAN communication
- Module update

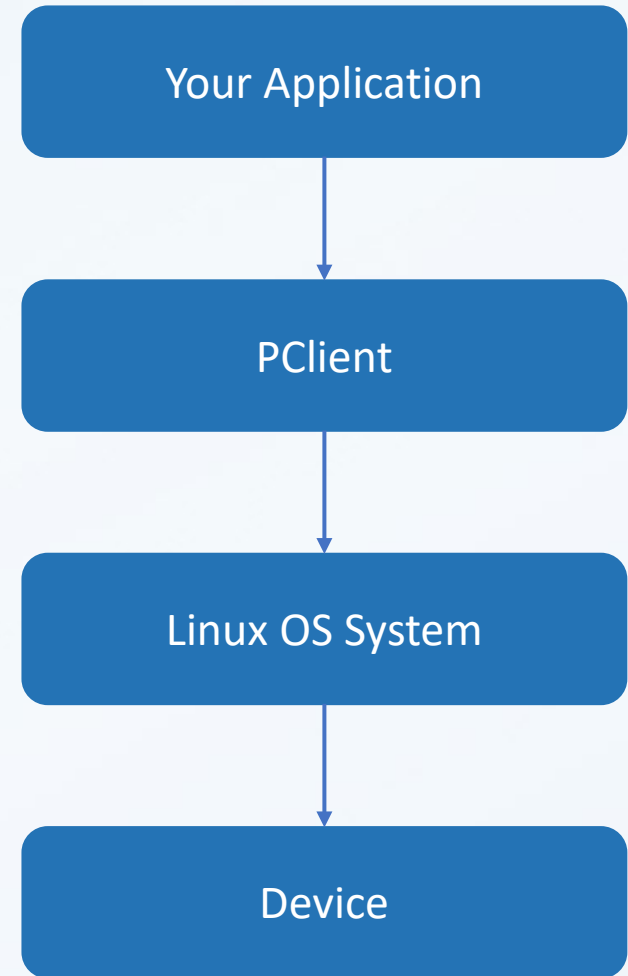
PROJEKTOR TOOL TRAINING

Introduction

Projector tool on the PC



PClient (Projector Client) and Linux OS on the device



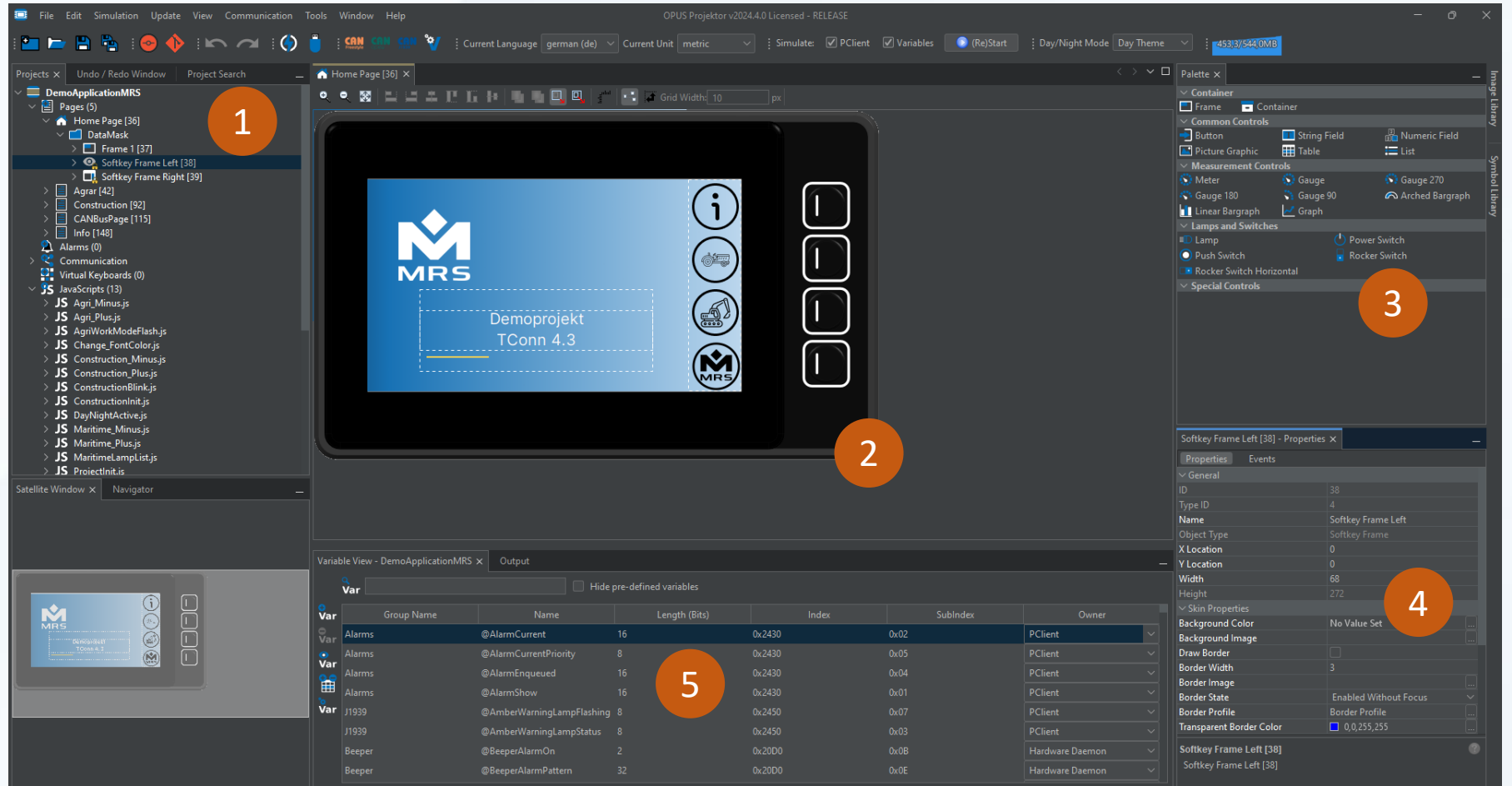
PROJEKTOR TOOL TRAINING



PROJEKTOR TOOL TRAINING

Overview

- 1: Project tree
- 2: Main window
- 3: Object palette
- 4: Settings and events of the selected object
- 5: Variables window



The screenshot displays the OPUS Projektor v2024.4.0 interface. The main window shows a simulation of a control panel with the MRS logo and text 'Demoprojekt TConn 4.3'. The interface is annotated with five numbered callouts:

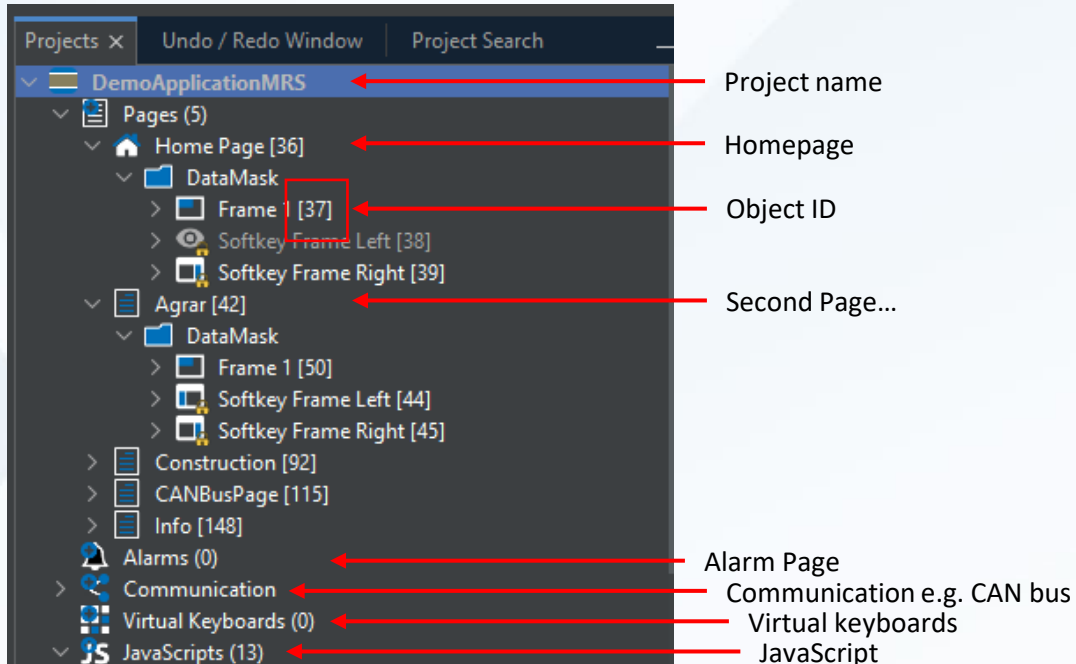
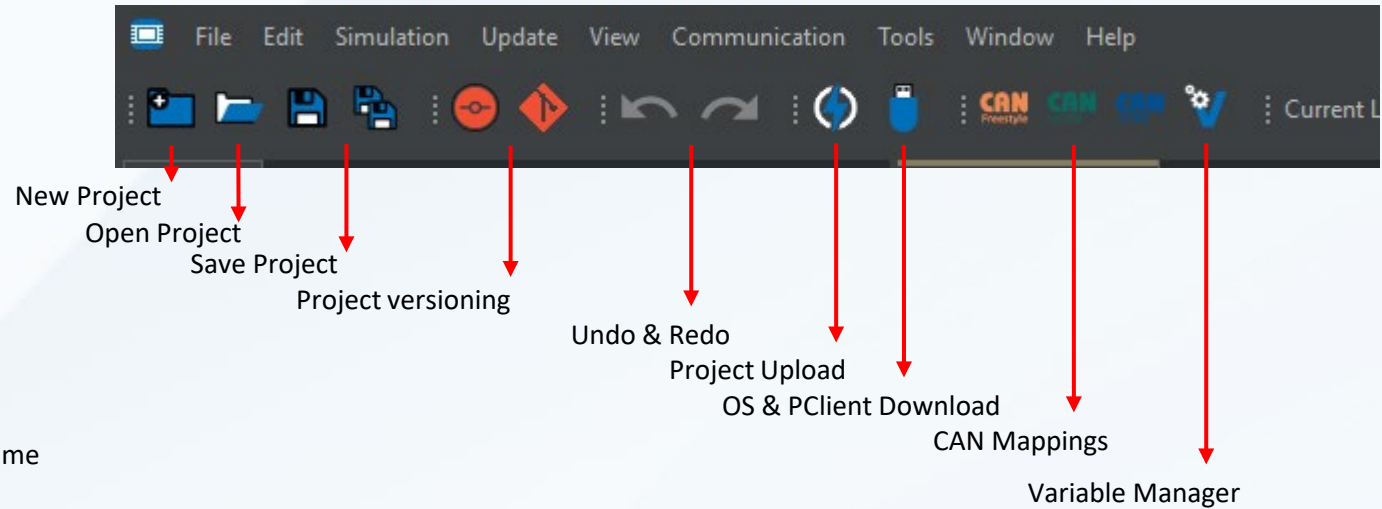
- 1:** Project tree (left sidebar)
- 2:** Main window (central simulation area)
- 3:** Object palette (right sidebar)
- 4:** Settings and events of the selected object (bottom right panel)
- 5:** Variables window (bottom left panel)

The Variables window shows the following data:

Var	Group Name	Name	Length (Bits)	Index	SubIndex	Owner
Var	Alarms	@AlarmCurrent	16	0x2430	0x02	PClient
Var	Alarms	@AlarmCurrentPriority	8	0x2430	0x05	PClient
Var	Alarms	@AlarmEnqueued	16	0x2430	0x04	PClient
Var	Alarms	@AlarmShow	16	0x2430	0x01	PClient
Var	J1939	@AmberWarningLampFlashing	8	0x2450	0x07	PClient
Var	J1939	@AmberWarningLampStatus	8	0x2450	0x03	PClient
Var	Beeper	@BeeperAlarmOn	2	0x20D0	0x08	Hardware Daemon
Var	Beeper	@BeeperAlarmPattern	32	0x20D0	0x0E	Hardware Daemon

PROJEKTOR TOOL TRAINING

Overview



PROJEKTOR TOOL TRAINING

New Project

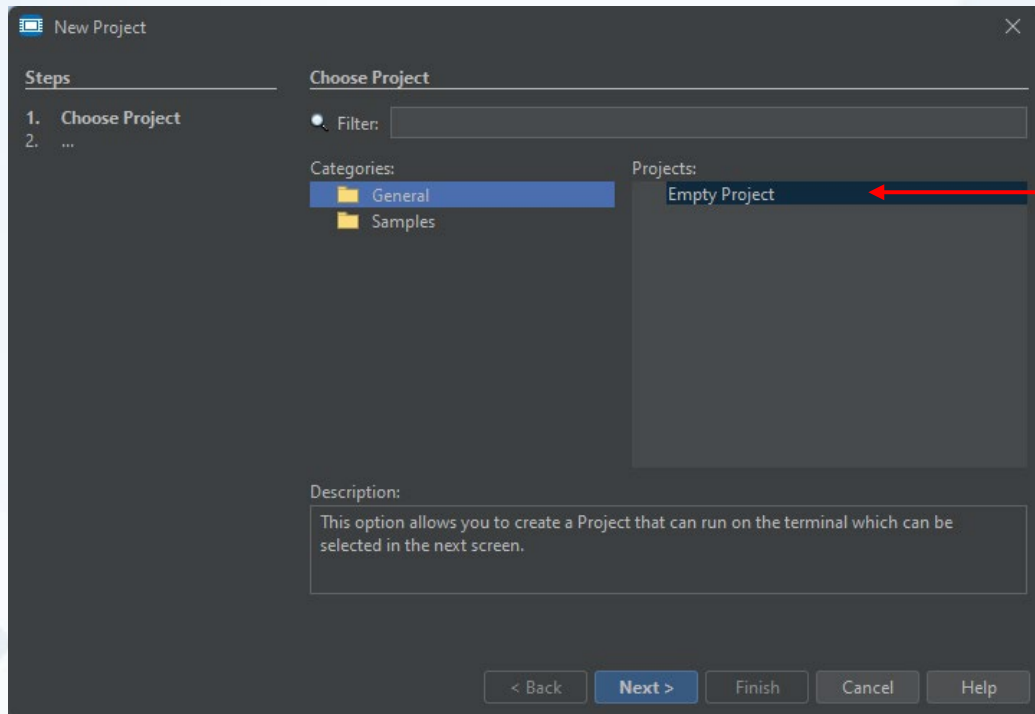
- Project creation
- Project background
- Objects & Buttons
- Variables
- Flashing software
- Extension:
 - CAN communication

Project description

In this project, we want to create a simulated dashboard that displays various parameters such as CPU load, temperature and the backlight. We use a pointer diagram to visualize the lighting. It is set using the physical buttons or via CANBUS

PROJEKTOR TOOL TRAINING

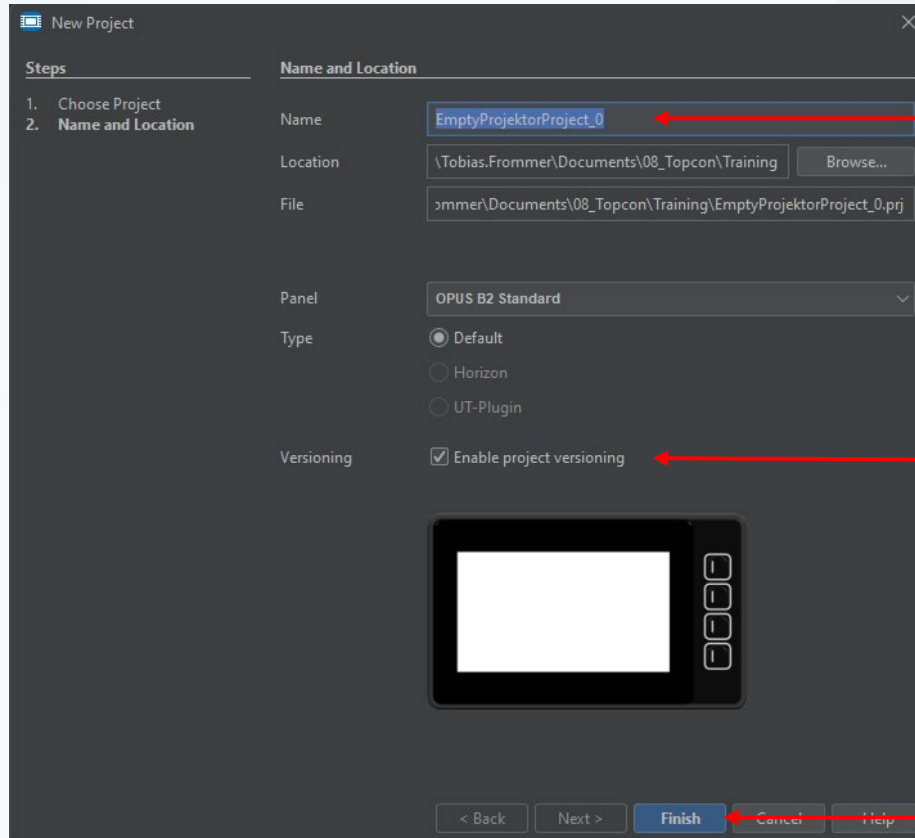
Project creation



A new project can be created using the “New Project” button, after which defined templates or an empty project (Empty Project) can be selected

PROJEKTOR TOOL TRAINING

Project creation



Project name

Location

Device name

- TConn 4.3 = Opus B2 Standard
- TConn 7 = Opus B2 Plus Standard

Activate if project versioning (local Git) is to be activated

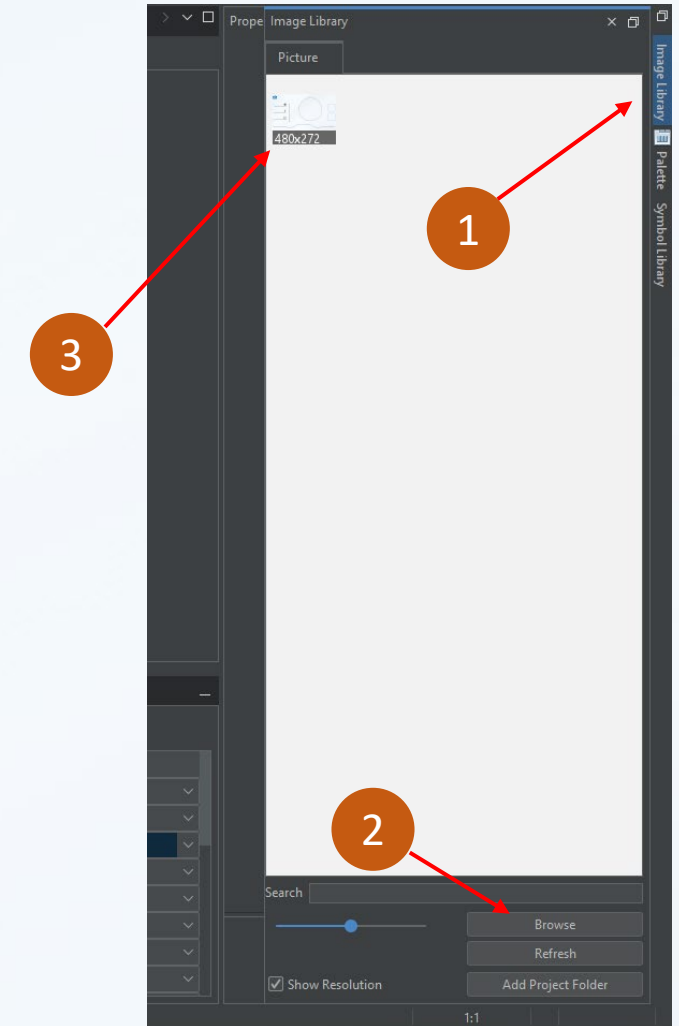
Create a project with Finish

PROJEKTOR TOOL TRAINING

Add background

Step 1: Add training images to the project

1. Open the image library on the right-hand side
2. Use the “Browse” button to select the training folder and click on Open
3. The images are then loaded from the folder

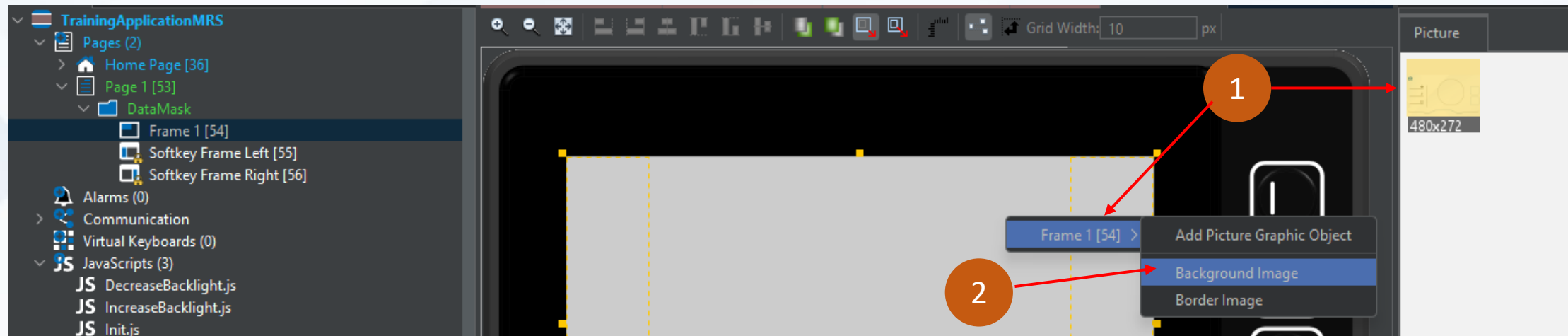
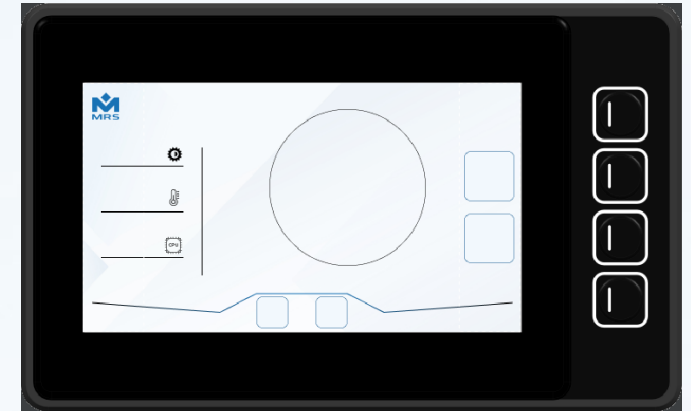


PROJEKTOR TOOL TRAINING

Add background

Step 2: Set background image as background

1. Drag and drop the image onto the gray area (frame)
2. 2 ways to add images:
 - Background Image = Background
 - Add Picture Graphic Object = normal placement of the picture



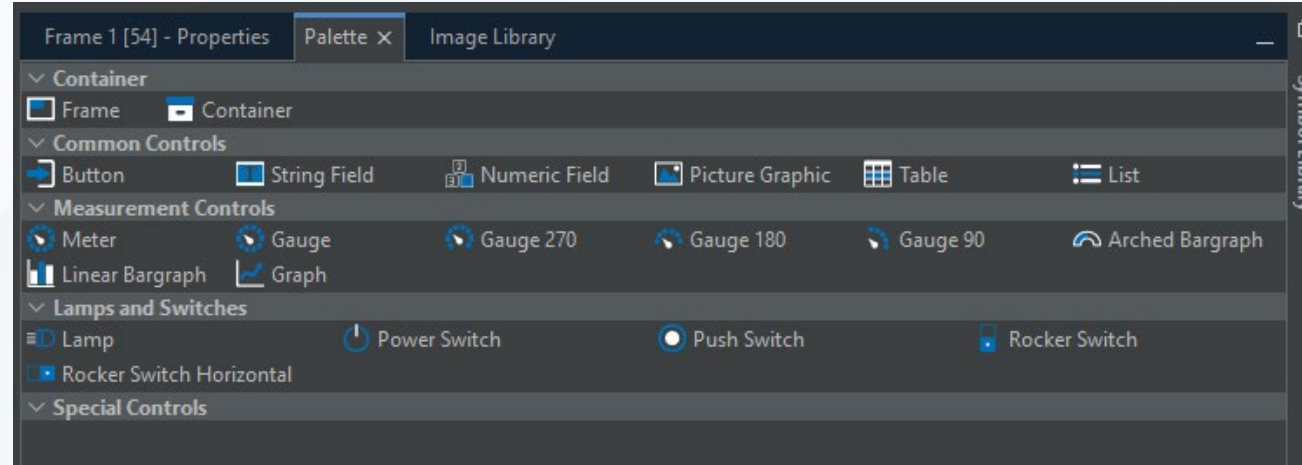
PROJEKTOR TOOL TRAINING

Objects & Buttons

An object is all content that can be added to the UI via drag & drop.

Examples of this are:

- Buttons
- String
- Numeric Fields
- Switch
- Meter
- Bargraph
- Gauge



PROJEKTOR TOOL TRAINING

Objects & Buttons

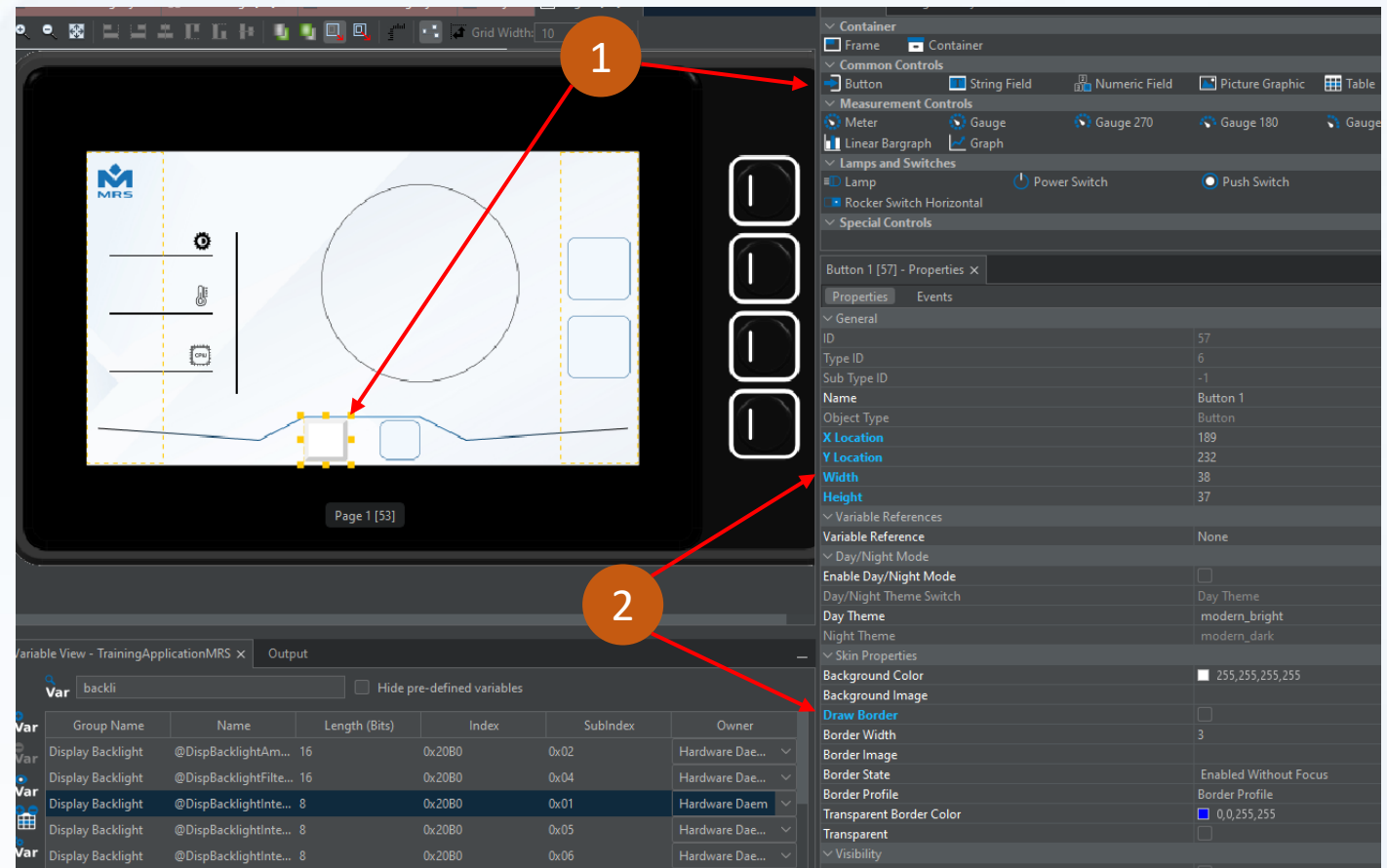
Once the background has been set, the buttons can be set. The buttons are for setting the lighting.

1. The button can be dragged and dropped onto the UI. The button is created automatically.
2. The properties of the button can then be set.

Make the following settings:

Height & width = 38

The button can then be copied to the right-hand side. (CTRL + C / CTRL + V)



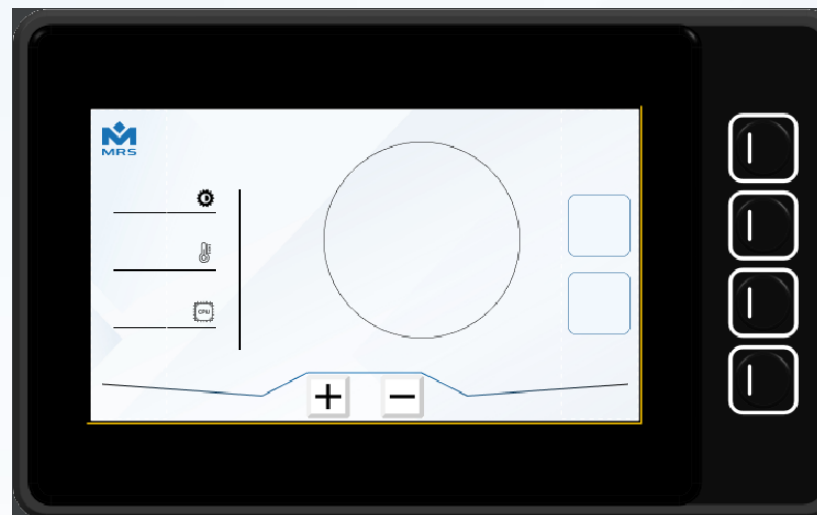
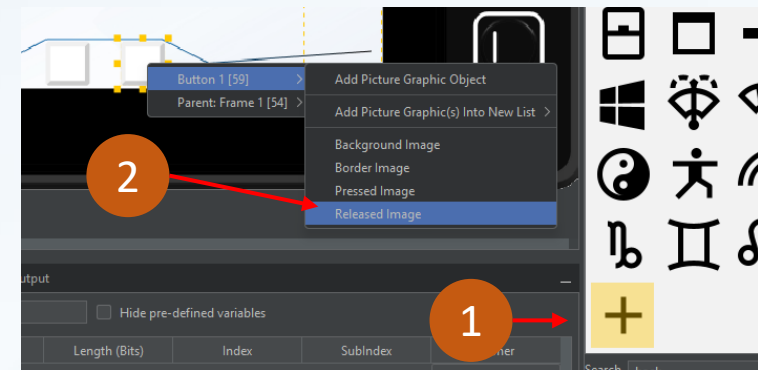
PROJEKTOR TOOL TRAINING

Objects & Buttons

After the buttons have been placed, symbols still need to be added to the buttons and the logic behind them created

1. In addition to the image library, there is also a symbol library. These symbols can also be added via drag & drop.
2. Find a plus and a minus sign in the library and add them to the buttons as a released image.

There are two different states of the buttons, pressed and unpressed. Depending on the state, different images and functions can be stored. Example: When the button is pressed, a “red plus” is displayed and when the button is not pressed, a “black plus” is displayed.

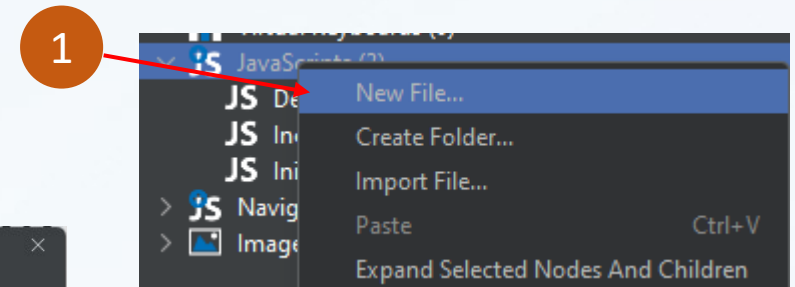
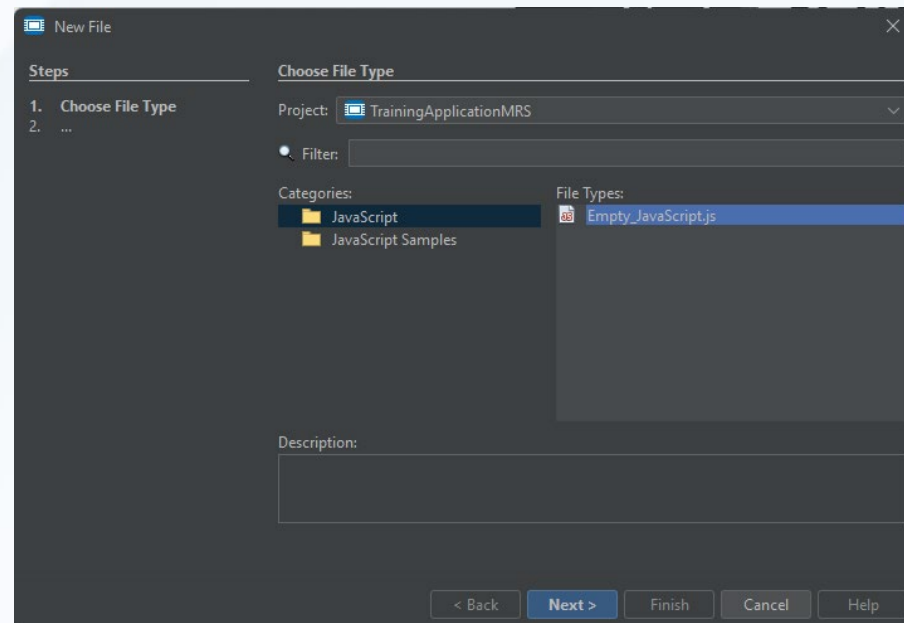


PROJEKTOR TOOL TRAINING

Objects & Buttons

The logic of the buttons can be created with so-called Java Script files. We want the logic to increase or decrease the backlighting by +5 or -5 per button press.

1. To do this, two JavaScript files must first be created.
2. In the project tree on the left-hand side, right-click to create a new JavaScript. We have the choice of creating a new JavaScript or using a sample. We create a new script and assign a name.



PROJEKTOR TOOL TRAINING

Objects & Buttons

We create a new JavaScript and insert the following content:

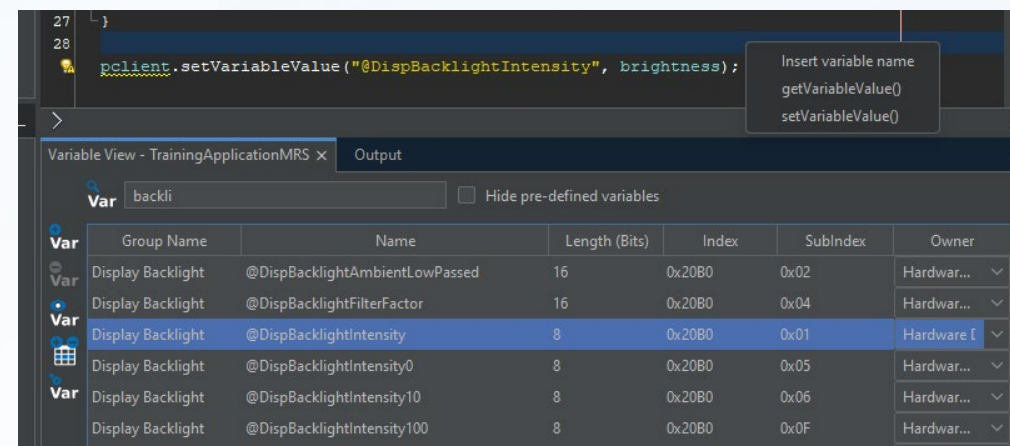
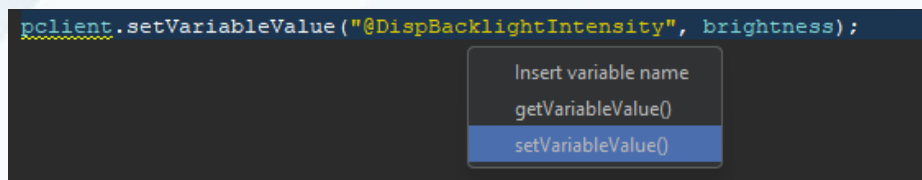
```
// Script to Decrease the Backlight with Touch or Button

var brightness = pclient.getVariableValue("@DispBacklightIntensity");
if (brightness > 14)
{
    brightness = brightness - 5;
}
else
{
    brightness = 10;
}
pclient.setVariableValue("@DispBacklightIntensity", brightness);
```

```
// Script to Increase the Backlight with Touch or Button

var brightness = pclient.getVariableValue("@DispBacklightIntensity");
if (brightness < 96)
{
    brightness = brightness +5;
}
else
{
    brightness = 100;
}
pclient.setVariableValue("@DispBacklightIntensity", brightness);
```

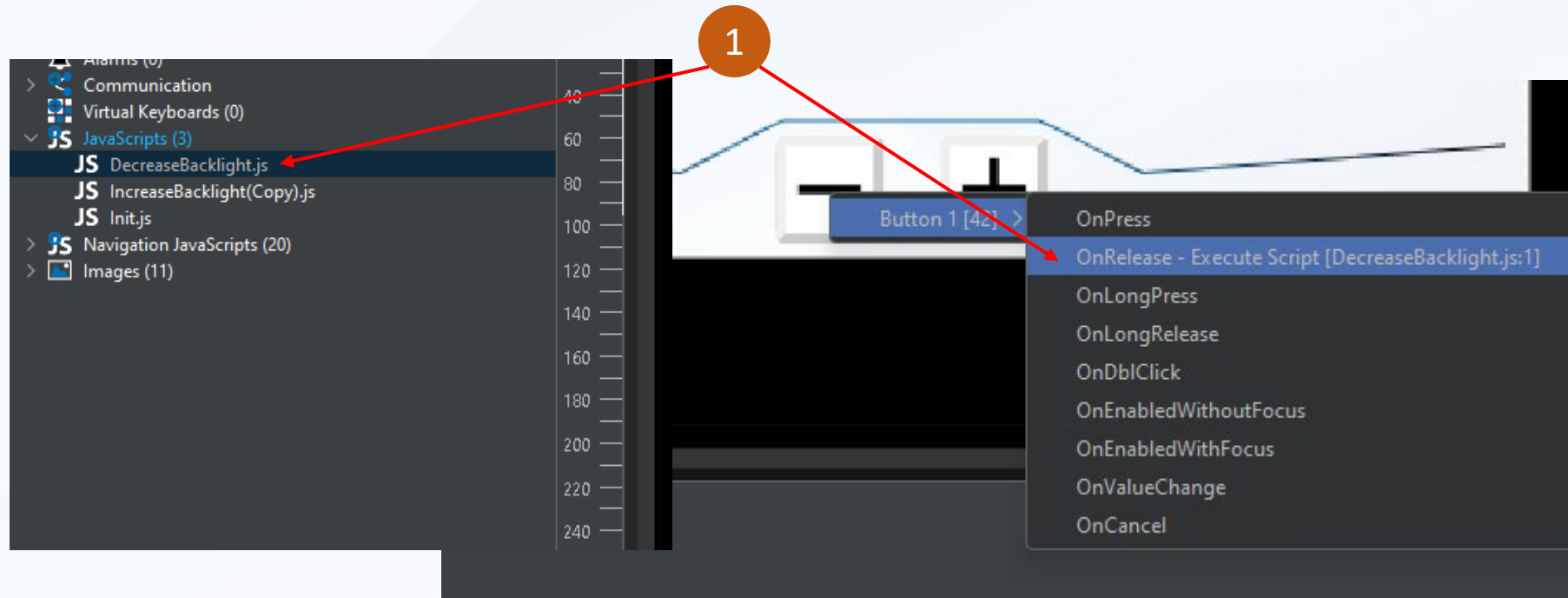
The variables can be created and searched for via the Variable View window in the lower area. The backlight “DispBacklightIntensity” is created as a default and can also be added to the JavaScript here via drag & drop.



PROJEKTOR TOOL TRAINING

Objects & Buttons

Once the JavaScript has been completed, it can also be dragged and dropped onto the button. A button has different signal types. Here you can select when the script should be executed. Carry out this step for the other button (+) and the other script (IncreaseBacklight) as well.

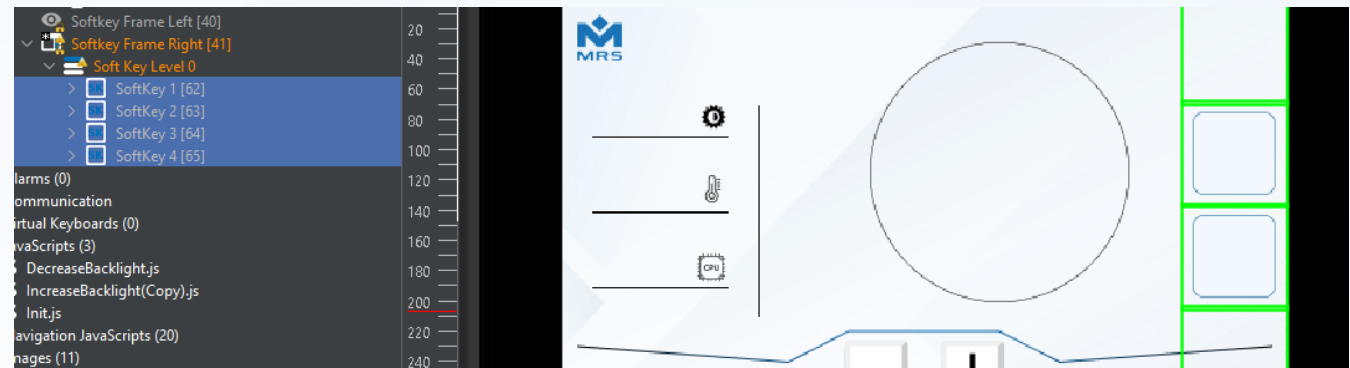
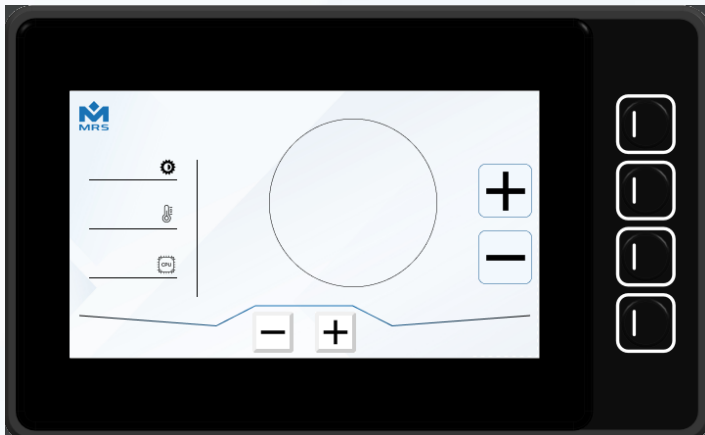
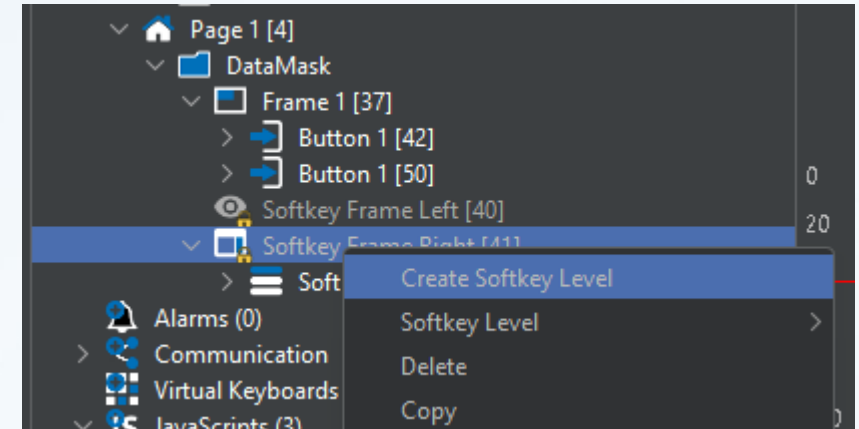


PROJEKTOR TOOL TRAINING

Using the side buttons

The side buttons function almost identical to the normal touch buttons.

1. To use the side keys, a Frame Right soft key must be created.
2. Right-click on the Frame Right softkey in the structure tree and create a new soft key level.
3. Four new fields appear on the right-hand side of the UI. These fields have identical functions to a button and are automatically linked to the 4 haptic keys.
4. Repeat the points from the “Objects & Buttons” topic: Add symbol and Add JavaScript.
5. The normal buttons and the haptic buttons should now have the same function.



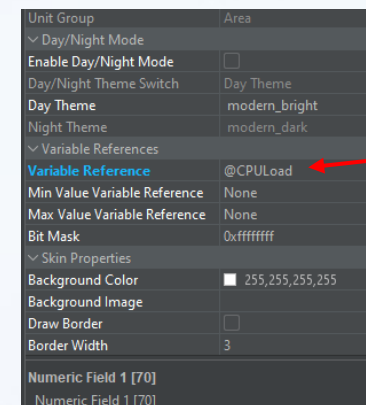
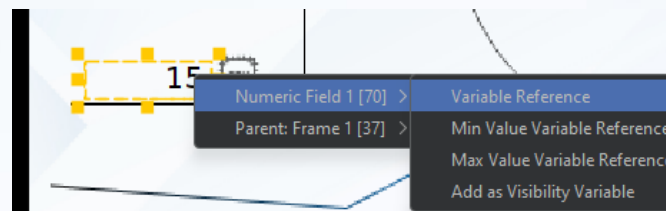
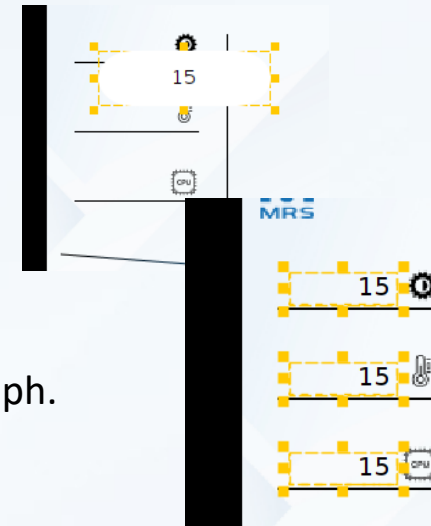
PROJEKTOR TOOL TRAINING

Using variables

To display operating parameters, the following parameters are now added to the left-hand side of the UI:

- Backlight
- temperature
- CPU

1. As the parameters are all numbers, a numeric field must now be added to the positions. This is also done via drag & drop.
2. The numeric field can now be changed using the properties.
3. Set the field to transparent and adjust it to the size. Also, right-align the paragraph.
4. Add the variable from the Variable View to the Numeric Field via Drag & Drop.
 1. @DispBacklighIntensity
 2. @SensorTerminalTempC
 3. @CPULoad



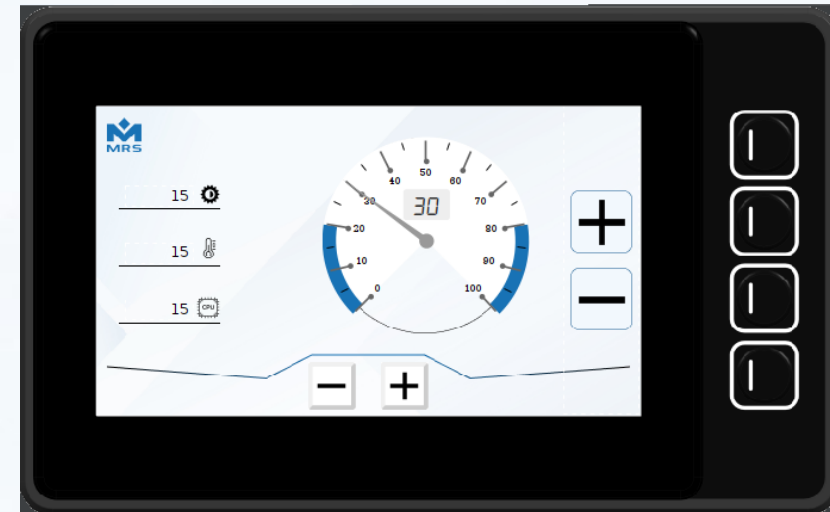
The variable is then stored in the properties of the numeric field

PROJEKTOR TOOL TRAINING

Adding pointer diagram

To visualize the display brightness, we now add a pointer diagram

1. Add the object Gauge 270 via Drag & Drop and adjust it to the position accordingly
2. Add the variable @DispBacklighIntensity as a Reference variable
3. Theoretically, the pointer diagram now works. However, as the pointer diagram is very useful and offers many setting options, a few more settings can be made here in the properties:
 - Draw Boarder -> deactivate
 - Max Value = 80
 - Min Value = 20
 - Absolute Max Value = 100
 - Absolute Min Value = 0
 - Color Above Minimum = 24,114,181,255
 - Color Above Maximum = 24,114,181,255
 - Color between = 255, 255, 255, 255, 255
 - Number of Ticks = 11
 - Number of Minor Ticks = 1

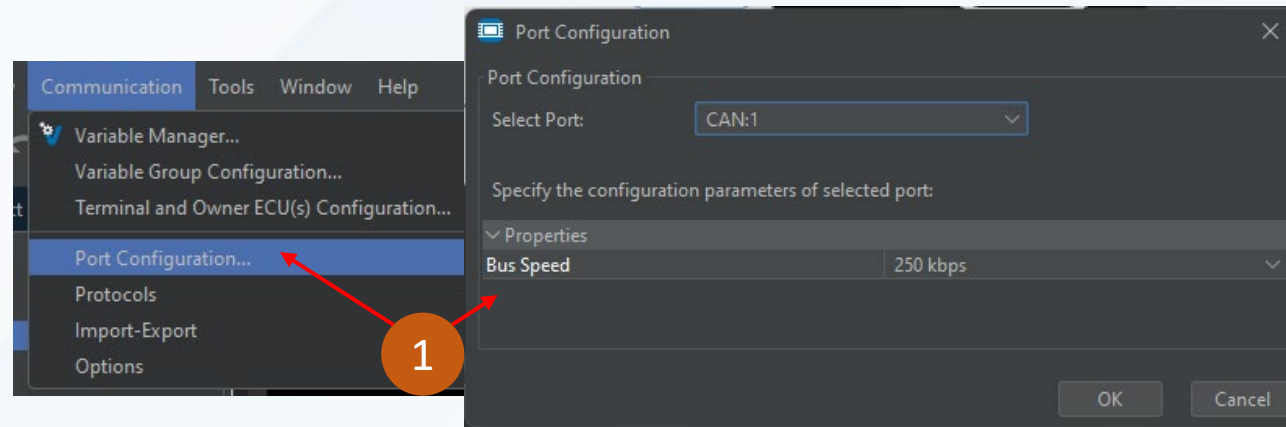


PROJEKTOR TOOL TRAINING

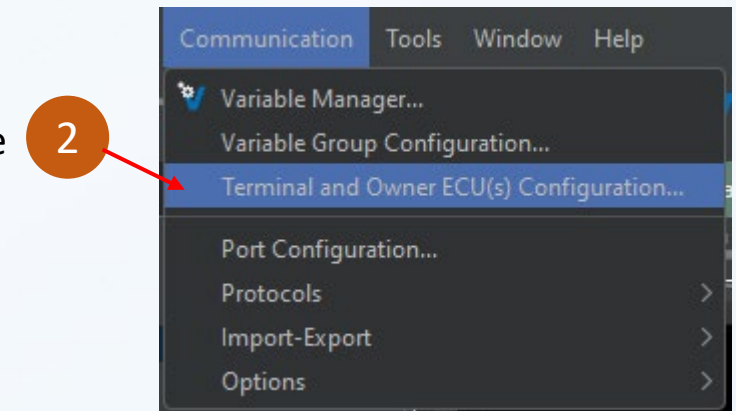
CAN-bus communication

The target is now to send the backlight variable via CAN bus. To do this, the communication must first be set.

1. The basic settings of the CAN bus can be set via the port configuration. The bus speed of port CAN1 and port CAN2 can be set here.



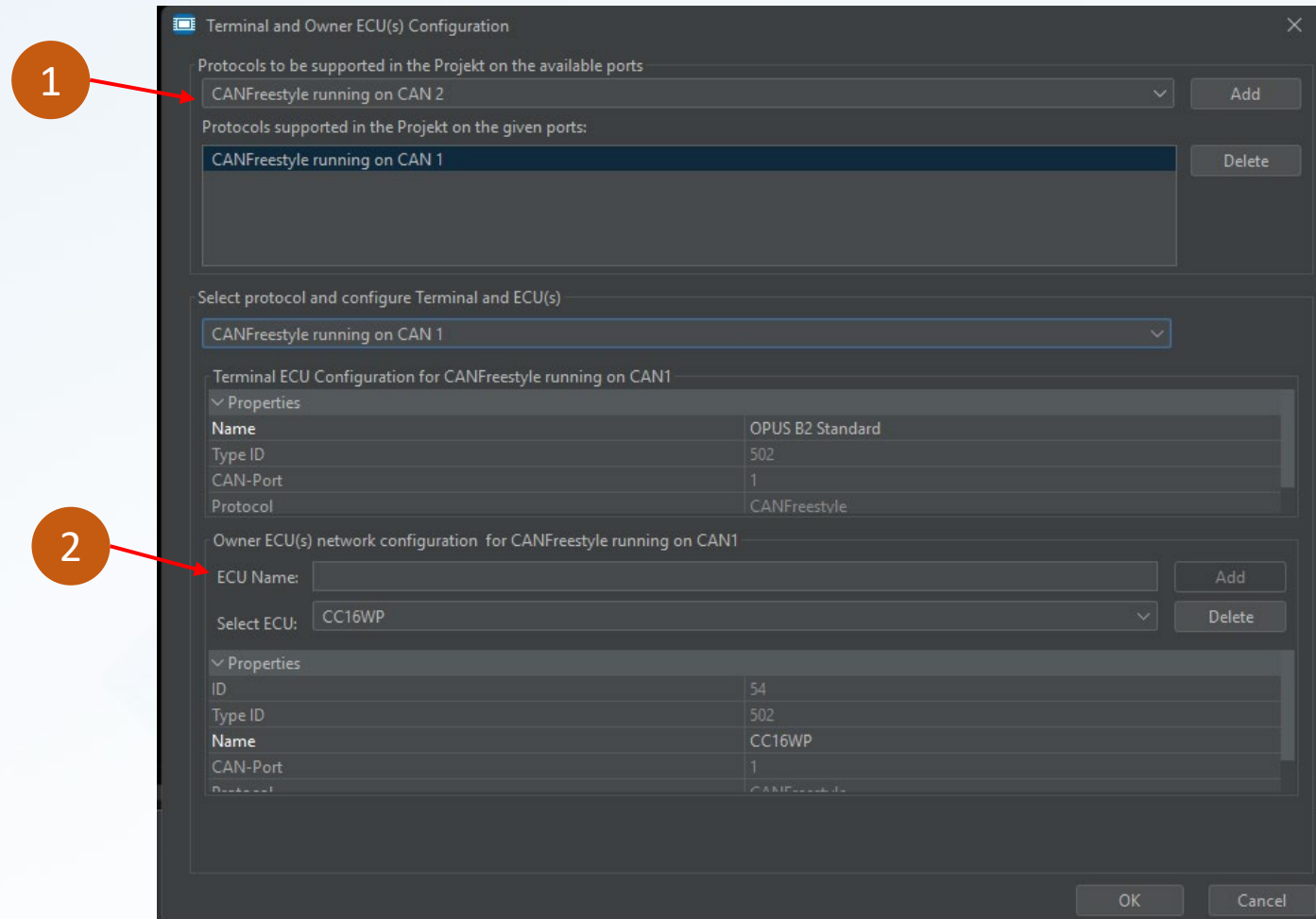
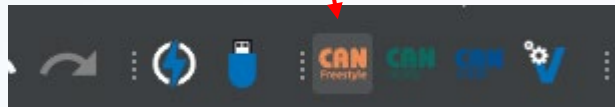
2. A protocol must then be set.
We create a CANFreestyle protocol, which means that we have complete freedom in the freedom in the design of the CAN communication and are not bound by communication protocols such as CANopen etc.



PROJEKTOR TOOL TRAINING

CAN-bus communication

1. Select CANFreestyle running on CAN 1 and click on “Add”.
2. Choose a name for your ECU and also click on “Add”.
3. Close the window and then click on the icon in the top bar to create the first CAN IDs

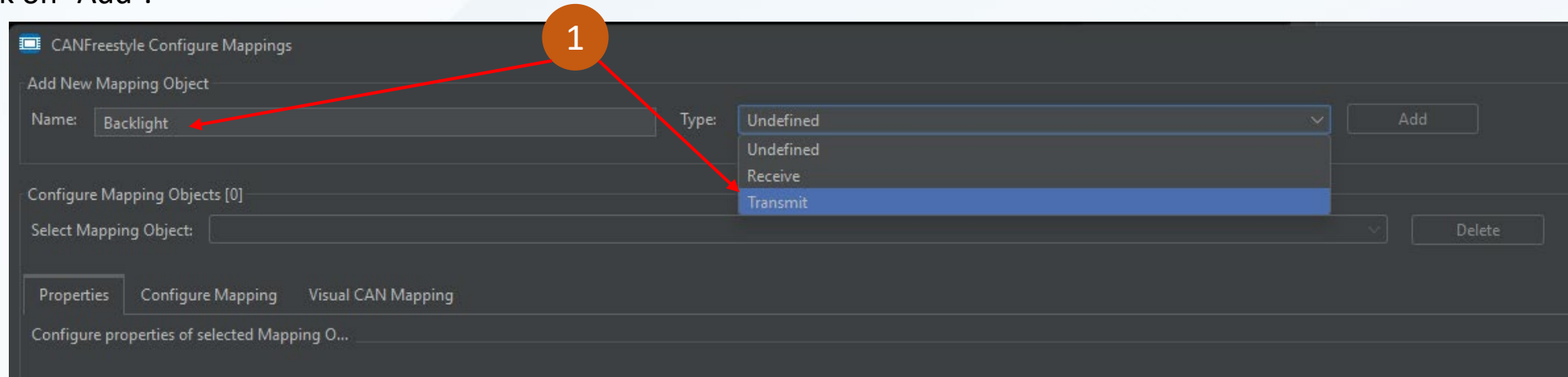


PROJEKTOR TOOL TRAINING

CAN-bus mapping

There are basically two different ways to create a CAN mapping. Either we generate a new mapping with all CAN IDs etc. or we load a corresponding .dbc file that exists from other projects. We will create a new mapping.

1. Select a name for the CAN ID and specify the type. Whether the ID is intended for sending or receiving. Then click on “Add”.

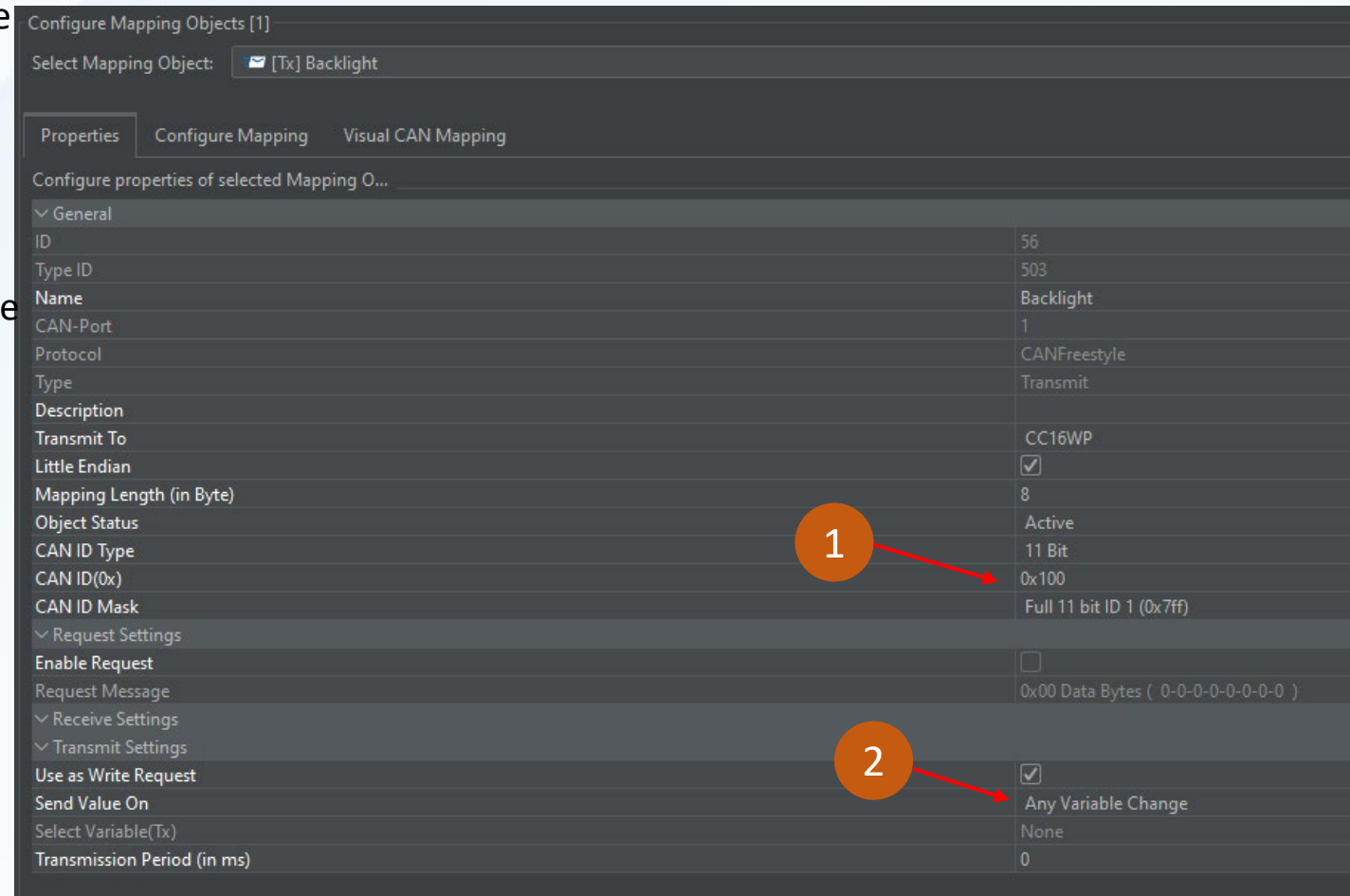


The CAN ID settings can then be made in the properties. The CAN ID can be set under the setting item “CAN ID(0x)”. Example 0x100

PROJEKTOR TOOL TRAINING

CAN-bus mapping

1. Various settings for the CAN ID can then be made in the properties. The CAN ID can be set under the setting item “CAN ID(0x)”. Example: 0x100
2. In the “Transmit Settings” tab, settings can be made that affect the sending of messages. The time at which the message is sent must be set here under “Send Value On”.



Configure Mapping Objects [1]

Select Mapping Object: [Tx] Backlight

Properties | Configure Mapping | Visual CAN Mapping

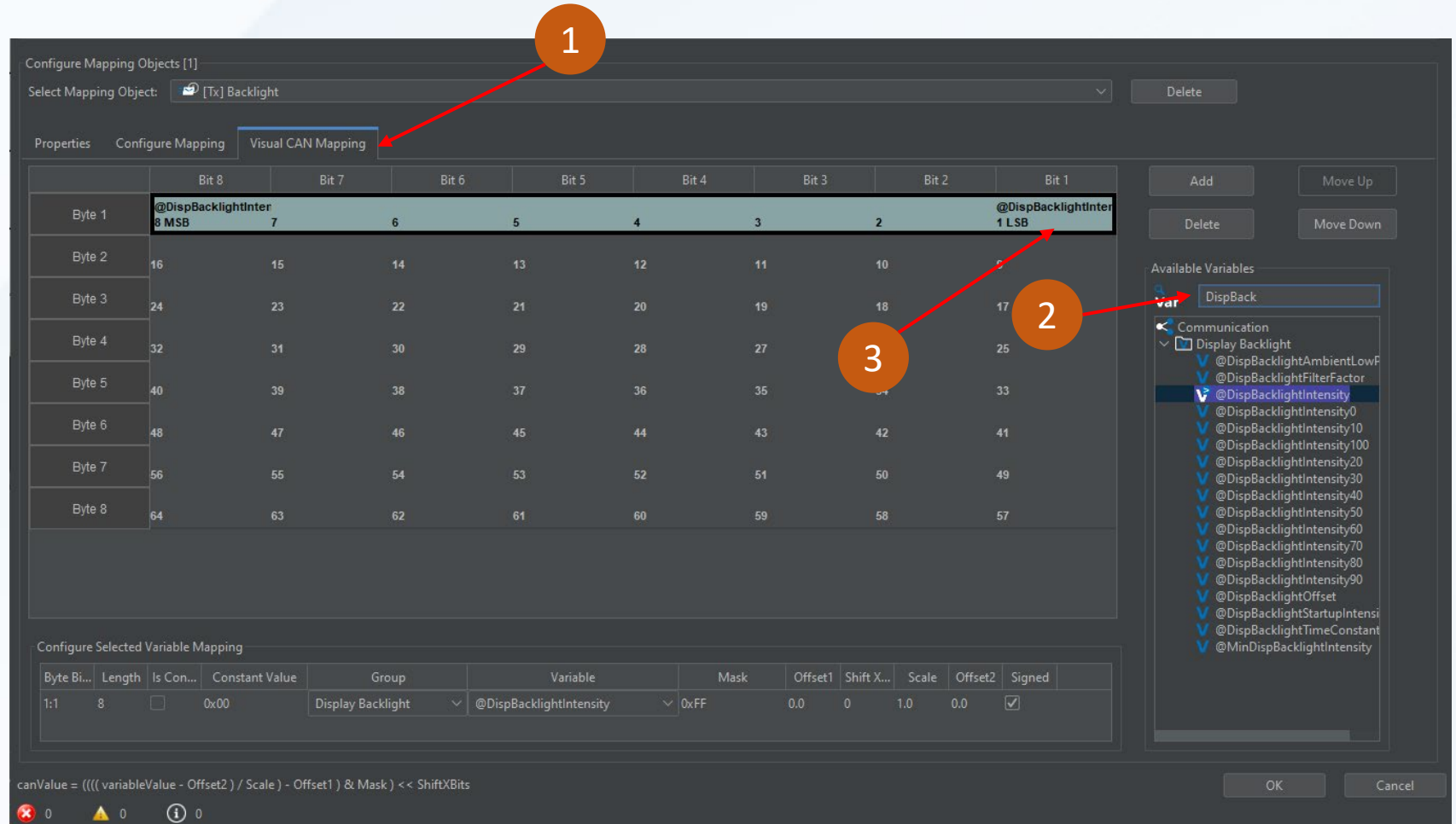
Configure properties of selected Mapping O...

General	
ID	56
Type ID	503
Name	Backlight
CAN-Port	1
Protocol	CANFreestyle
Type	Transmit
Description	
Transmit To	CC16WP
Little Endian	<input checked="" type="checkbox"/>
Mapping Length (in Byte)	8
Object Status	Active
CAN ID Type	11 Bit
CAN ID(0x)	0x100
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

PROJEKTOR TOOL TRAINING

CAN-bus mapping

1. The actual mapping of the CAN message is carried out in the “Visual CAN Mapping” tab.
2. Search for the backlight variable
3. Drag and drop the variable into the window to the Bit 1 position.



Configure Mapping Objects [1]

Select Mapping Object: [Tx] Backlight

Properties Configure Mapping **Visual CAN Mapping**

	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
Byte 1	@DispBacklightInter 8 MSB	7	6	5	4	3	2	@DispBacklightInter 1 LSB
Byte 2	16	15	14	13	12	11	10	9
Byte 3	24	23	22	21	20	19	18	17
Byte 4	32	31	30	29	28	27	26	25
Byte 5	40	39	38	37	36	35	34	33
Byte 6	48	47	46	45	44	43	42	41
Byte 7	56	55	54	53	52	51	50	49
Byte 8	64	63	62	61	60	59	58	57

Available Variables

- var DispBack
- Communication
 - Display Backlight
 - @DispBacklightAmbientLowF
 - @DispBacklightFilterFactor
 - @DispBacklightIntensity**
 - @DispBacklightIntensity0
 - @DispBacklightIntensity10
 - @DispBacklightIntensity100
 - @DispBacklightIntensity20
 - @DispBacklightIntensity30
 - @DispBacklightIntensity40
 - @DispBacklightIntensity50
 - @DispBacklightIntensity60
 - @DispBacklightIntensity70
 - @DispBacklightIntensity80
 - @DispBacklightIntensity90
 - @DispBacklightOffset
 - @DispBacklightStartupIntensi
 - @DispBacklightTimeConstant
 - @MinDispBacklightIntensity

Configure Selected Variable Mapping

Byte Bi...	Length	Is Con...	Constant Value	Group	Variable	Mask	Offset1	Shift X...	Scale	Offset2	Signed
1:1	8	<input type="checkbox"/>	0x00	Display Backlight	@DispBacklightIntensity	0xFF	0,0	0	1,0	0,0	<input checked="" type="checkbox"/>

canValue = (((variableValue - Offset2) / Scale) - Offset1) & Mask) << ShiftXBits

OK Cancel

PROJEKTOR TOOL TRAINING

Installing the application on the TConn

There are different ways to install the Application on the TConn.

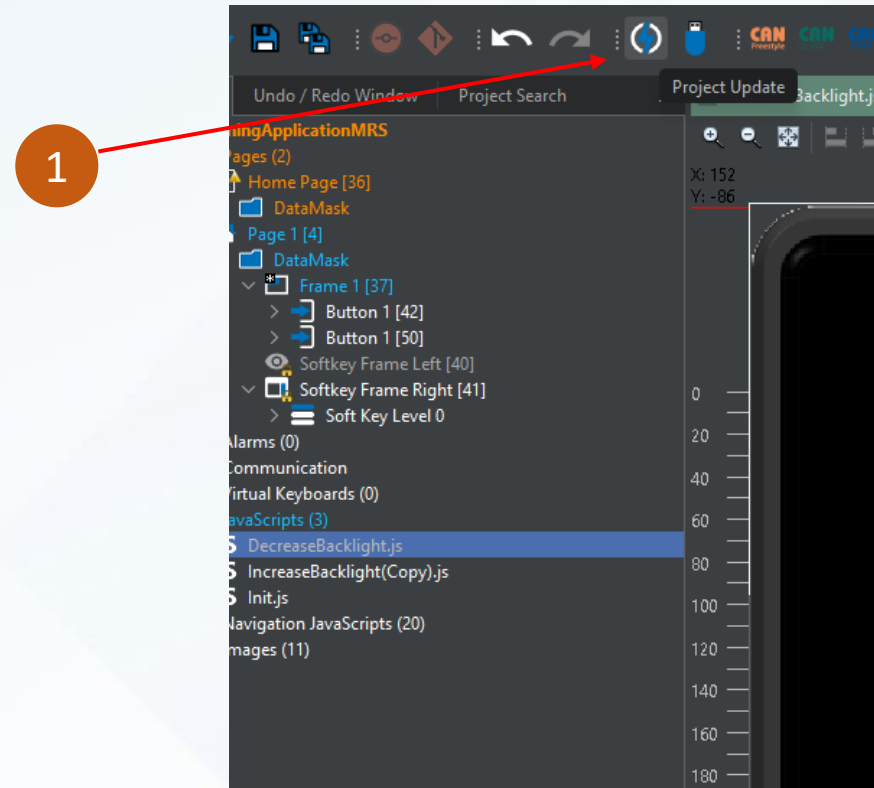
1: via CAN bus to CAN 1 or CAN 2

2: via USB-C with a USB stick

!! For larger projects, transfer via USB stick is recommended, as the files may become larger depending on the size of the project!

1. Step 1: Click on Project Update in the upper area

The update wizard opens

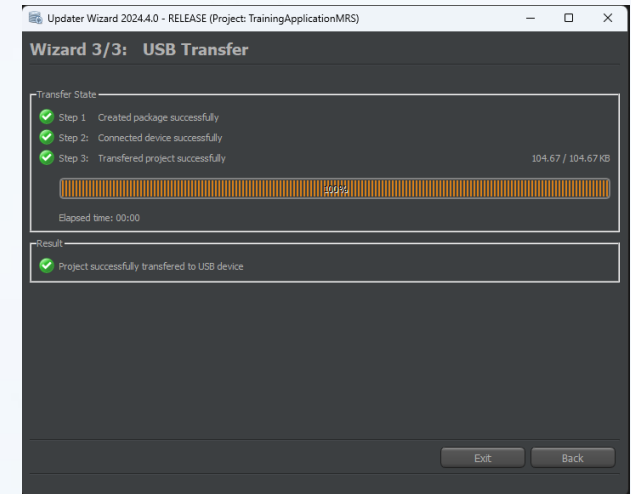
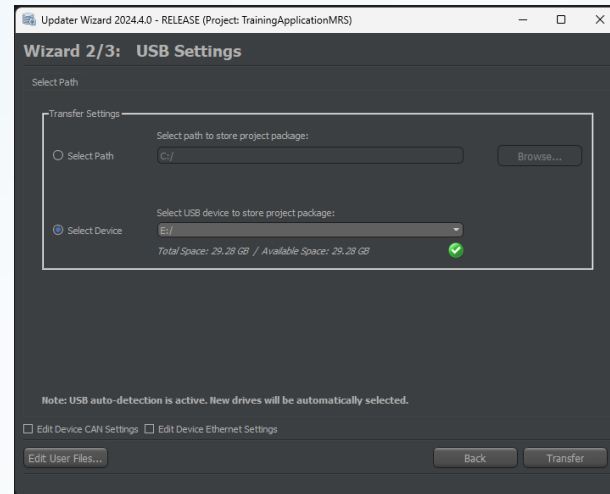
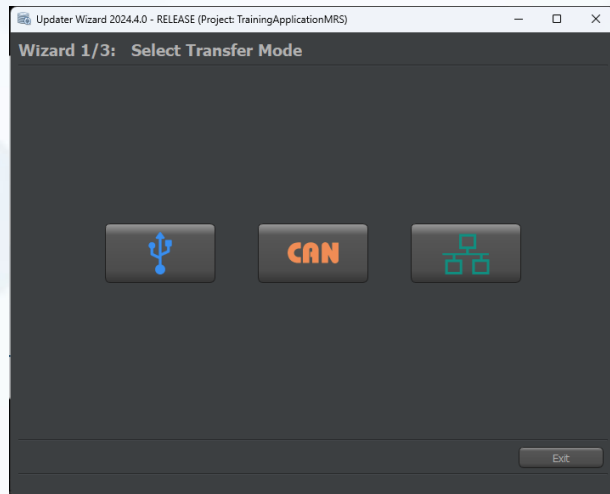


PROJEKTOR TOOL TRAINING

Installing the application on the TConn

Step 2: Follow the instructions of the Updater

1. Select USB in the first window.
2. Connect the USB stick to the PC and make sure that it has a FAT32 file system.
3. Select the correct path in the wizard and click on Transfer
4. The transfer starts automatically. As soon as the transfer is finished, the USB stick can be plugged into the back of the TConn.
5. The update runs automatically. The application then starts automatically.



You have successfully completed the training!

If you have any further questions, please do not hesitate to contact our support team at support@mrs-electronic.com