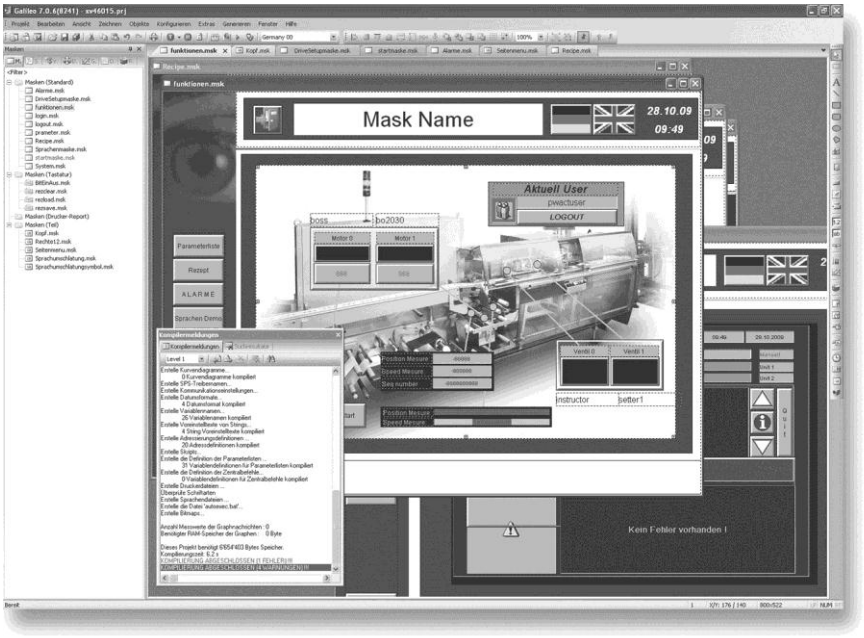
01/2013 MN04802108Z-EN
M004348-01

GALILEO User Help



Powering Business Worldwide

Imprint

Manufacturer
Eaton Automation AG
Spinnereistrasse 8-14
CH-9008 St. Gallen
Switzerland
www.eaton-automation.com
www.eaton.com

Support
Region North America
Eaton Corporation
Electrical Sector
1111 Superior Ave.
Cleveland, OH 44114
United States
877-ETN-CARE (877-386-2273)
www.eaton.com

Other Regions
Please contact your supplier or send an E-Mail to:
automation@eaton.com

Original instructions
The German version of this document is the original instructions.

Editor
Jürgen Wagner

Brand and product names
All brand and product names are trademarks or registered trademarks of the owner concerned.

Imprint

1 General

This document is designed as a user help for creating projects with the GALILEO visualization software.

The «How to do» of special objects like «Help Informations», «Enhanced Submasks with User defined Data Types», «User logger», «Graph», «Parameter lists» and more, is not described in the GALILEO Online-Help.

This document is continuously updated and does not refer to a specific GALILEO version. Examples, screenshots and the functions described may therefore differ from those of the GALILEO version you are using.

This document should help you to generate special objects in GALILEO, with less time expense, based on examples, so that they can be used immediately. There will not be a complete description of the functions, but a useful procedure to generate the objects described.

This document is not necessarily complete.

2 Mask design

2.1 Introduction

This chapter serves as a guide for the design of project masks. Problems with regard to the generation of masks and the updating of data in the runtime program can be avoided by observing a few simple rules when creating masks during the project design. This particularly applies to graphically demanding projects.

2.2 Basic losses in performance

The following factors in particular may considerably reduce the performance of the runtime program:

- Objects in which «Transparent» is activated.
- Several overlapping dynamic and transparent objects.
- Badly designed order of objects.
- Overlapping operable objects that are visible simultaneously.

2.3 Purpose

This document is designed to enable the user of GALILEO to do the following:

- ensure the optimum design of masks already during planning and project creation,
- detect and eliminate the problems arising from poorly created project masks that require a long time to be generated or updated.

2.4 Definitions

2.4.1 Objects:

There are basically two types of objects that have an effect on performance: dynamic and non-dynamic objects. An object is dynamic if one or several of the following conditions are fulfilled:

- the size can be dynamically adjusted by means of tags,
- the position can be dynamically adjusted by means of tags,
- accessibility or visibility can be defined,
- the content of the object (e.g. value entry / display, selector switch, status display with several display options for texts and bitmaps, …) can be changed

All other objects are classed as non-dynamic.

2.4.2 Object order:

When objects are created, these are numbered in ascending order. This order can be viewed in the menu «View» – «Tooltip» – «Object Order» or via the «Project Tools» toolbar.

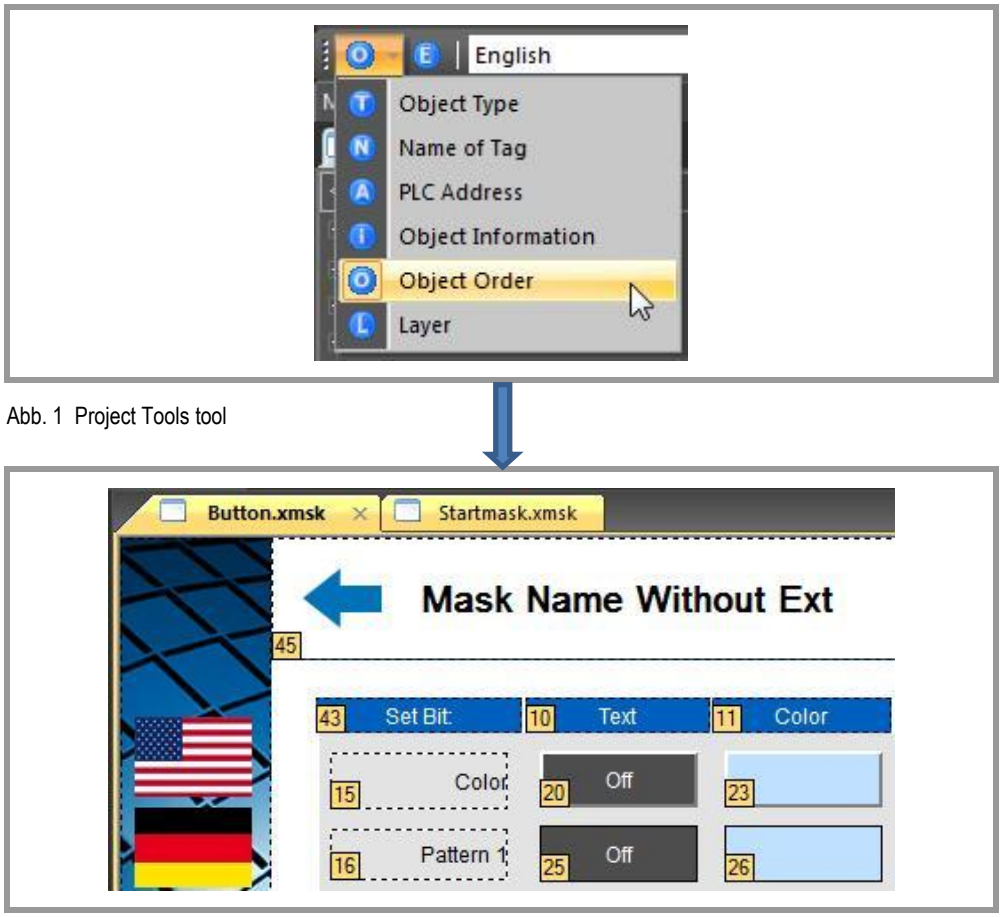


Abb. 1 Project Tools tool

Abb. 2 View object order

Proceed as follows in order to change the order of an object:

Select the object to be moved. Press the + (Plus) or – (Minus) keys. With every key activation the selected object will be moved forward or backward (also possible via the context menu – «Move Forward» or «Move Backward»).

2.4.3 Overlapping:

Overlapping objects are objects that lie on top of each other in part or in full. If a dynamic object (i.e. the status, the tag value, position, …) changes, all objects that have a higher object order and overlap the dynamic object are updated.
If Transparent is also activated in this dynamic object, the objects beneath it, i.e. the objects lower down in the object order, are updated if the dynamic object overlaps them.

☞ **Diagonal lines are considered as transparent objects, i.e. a line inside a transparent rectangle around the end points of the line.**

2.4.4 Background image:

The GALILEO runtime scans through the objects on a mask in ascending order (from 0 upwards) and notes the first dynamic object. All non-dynamic objects up to this dynamic object are shown as a background image. The objects after the first dynamic object are then refreshed as described under «Overlapping».
The special feature of the background image is the fact that it is not updated.

2.5 Resources for optimized mask generation

The following basic rules for optimum mask generation can be derived from the definitions described above:

- Ensure the correct order of objects. Drawn objects like rectangles, lines, polygons etc. but also images that are used as static background images, must have the lowest position in the object order (Object order, see chapter 2.4.2).
- Overlap transparent, dynamic objects as little as possible with other objects, since all objects above or below them are updated if the object is changed.
- Particularly ensure that transparent objects do not overlap large objects on which many other transparent objects are placed. These are all updated if one object is updated. This can cause a chain reaction which considerably reduces performance (see chapter 2.6 «Example»).
- The accessibility of operable objects that overlap each other must be configured so that only one object is visible.

2.6 Example

A project with 5 production lines.

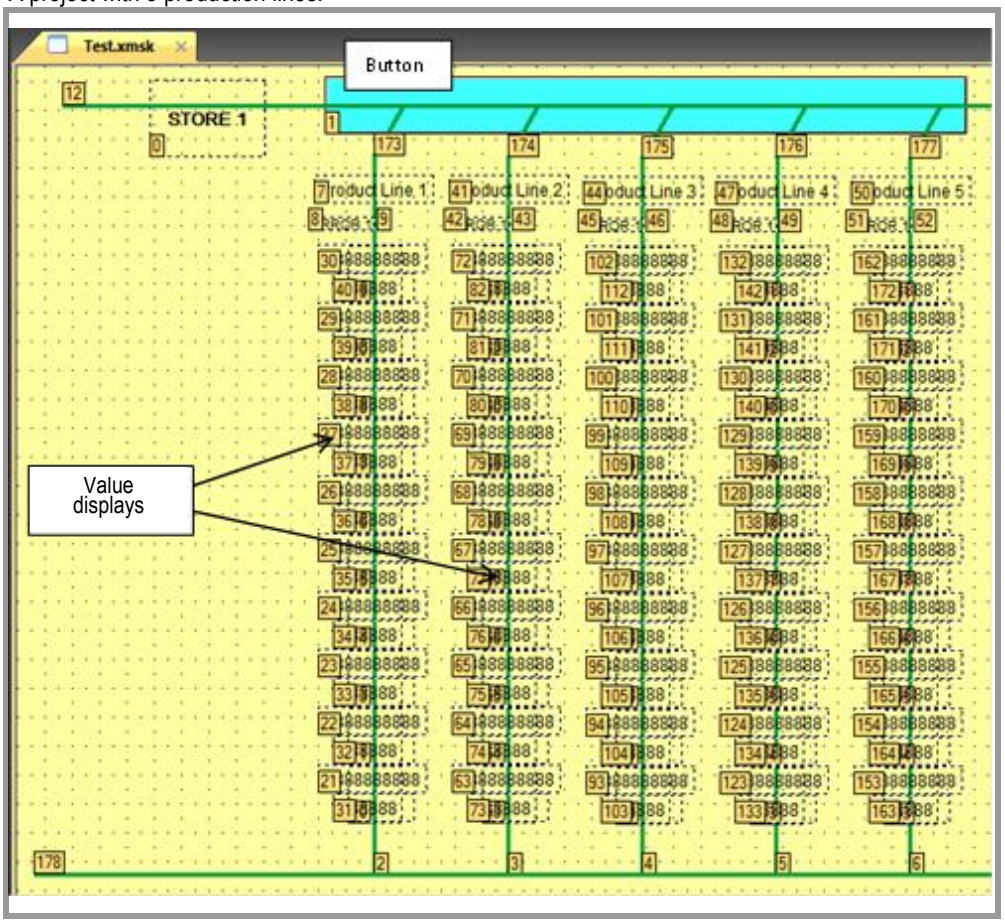


Abb. 3 Example with considerable performance losses

Over 20 value displays are defined in each production line. All value displays are transparent so that the vertical green lines underneath them are visible. The value displays touch each other and do not overlap.
All vertical green lines lead to the blue button (this is a dynamic object) at the top of the image. As this blue button is positioned lower in the object order than the vertical green lines, these are not regarded as a background image although they are meant to be. As soon as one of the 100 values changes, the green line underneath it will also be updated because the value is transparent. The green line in turn overlaps the diagonal green line and is regarded as transparent (see chapter 2.4.3 - Overlapping *Special note*). The diagonal green line overlaps the horizontal green line. All other production lines are likewise interconnected via the horizontal line (also overlapping). If a value display therefore changes, all 100 display values are updated. This example therefore requires long update times.

To increase the performance, it is only necessary to set the order number of the blue button as high as possible so that all green lines are positioned beneath it → In this case to order number 6. The objects 0 ... 5 (all vertical lines underneath it) are therefore declared as background images that are no longer updated so that the updating in a production line no longer takes place. If a value changes, only the changed object is updated.

Blue button on object order position 6

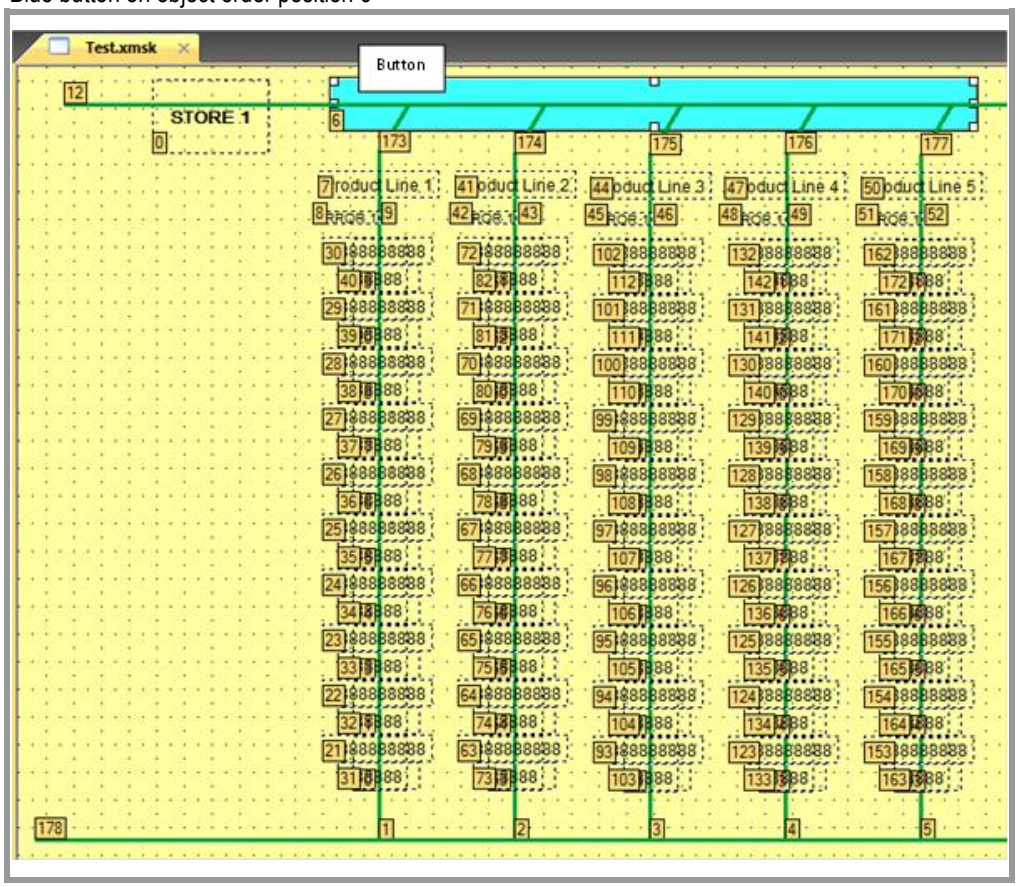


Abb. 4 Example with performance losses rectified

3 Copying masks and their content between projects

3.1 Introduction

This chapter describes the best way to copy masks, objects and their parameters from project A to project B.

3.2 Basic requirement

- From GALILEO version 6.01.
- It is only possible to copy between projects created in the same GALILEO version.

3.3 What can be copied

When copying objects on masks, the bitmap files, texts and tag assignments are copied as well.

Bitmaps

If the target file already exists and the source and target file are different, the user can create a new file name. Also when importing bitmaps, the user is also only required to enter a new file name if the target file already exists and the source and target file are different.

Texts

The language name serves as a reference for the language.

- All matching languages are determined.
- A search for texts of the first matching language is carried out.
- If the text is found, the corresponding text ID is transferred.
- If the text is not found, the text is added in all matching languages. If possible, the same text ID is used as in the source project. Otherwise the text is added at the back of the list.
- No texts are added if no matching languages are found.

Mask change

When mask change and submask objects are added, the corresponding mask name is searched for in the target project and assigned. If the mask name does not exist, the mask assignment is deleted.

Recipes

The tag assignments are retained if the tags already exist in the target project. This applies to all tags inside the recipe including the write protection settings. The security request masks are also transferred if the mask files already exist.

Tags

When copying tags, all parameters are copied as well. The dynamic limit value tags are also transferred correctly if any are defined. However, for this the limit value tags must be copied first of all. The communication parameters must also be set so that the addressing is copied as well. For this the correct communication number and order in the communication table must be ensured (not the communication type).

Parameter list entries

The parameter list entries (in the parameter list manager) such as group, accessibility (see also the note on «Tags»), number etc. are correctly transferred. If required the group is created anew.
The group name is used as a reference.

Graphs

The tag assignments are retained if the tags already exist in the target project. This applies to all tags inside the graph including the trigger variables. The security request masks are also transferred if the mask files already exist.

Printer forms

The tag assignments are retained if the tags already exist in the target project. The texts are copied as well.

3.4 Preparation and basic settings

In GALILEO, various preparations and settings are necessary which have to be made before copying in order not to lose different settings. Follow these step-by-step instructions.

3.4.1 Recommendation

Save the target project (under another name or as a backup) in order to access the original version in case the result of copying the project is unsatisfactory.

3.4.2 Preparation

The communication parameters of the source and the target project must be identical. Otherwise the addresses may not be transferred correctly (with several communication channels the communication types must correspond to the order in the communication table).
Check the general project settings for the source project.
Menu Config → Settings. Transfer this to the target project.
Example: in the «Others…» tab.

Note: If the panels used have different sizes, objects may be positioned outside of the mask when they are added (e.g. if copied from a larger panel to a smaller one). In this case the objects must be positioned correctly before copying.

3.4.3 Languages

Languages must be created with the same name. The copied texts are only transferred to the languages that also have the same name. Our examples are based on a target project which does not yet contain any texts. A new, empty project has therefore been created.

Different options are therefore possible:

- Creating languages in GALILEO
- Importing
- Creating empty XML language files
- Copying

Creating languages in GALILEO
Select «Config» → «Language». A language 00 is automatically created when a new project is opened for the first time. This language name must correspond to the language name in the source project. For example, let us call it 'Germany'. In order to create a new language, click the «right mouse button» → «Add language» or «function key F6», enter a name and confirm with the Enter key.

Importing
This method makes sense if most or all texts are to be copied from the source to the target project.
Menu «Config» →«Export and import texts».
This enables all texts required to be exported via an Excel spreadsheet and imported into the target project.

Creating empty XML language files
Creating an empty XML file which carries the name of the language we wish to add later. As already mentioned, a «Language00» is created automatically. The target project folder contains a folder «lang» for this purpose. This is used for storing all the languages of a project in XML format. Example: Language00_00.xml.
This file can then be copied as often as required and added to the «lang» folder. Rename the file after copying and enter the names of the languages present in the source project.

Copying XML
Example: The «lang» source project folder contains all the languages of the project, such as Germany_00.xml.
This file can be copied directly to the «lang» folder of the target project. In this way, all texts are transferred, although this is not always desired.

3.5 Copying

Observe the following order when copying:

1. User-defined data types

The user-defined data types are automatically generated if the associated instantiated tags are copied.

2. Tags

Copy all required tags from the source project to the target project.
Only one data type can be copied. Only structures can be copied simultaneously.
With tags that are interdependent (e.g. with dynamic limit values), the tags of the limit values must be copied first.
If the structure is an instance of a user-defined data type, this is created automatically.

3. Data-dependent objects

There are many objects that have a database, such as recipes, parameter lists, graphs, scripts.
This database must be copied before the objects! Recipes and graph data blocks can be copied directly. This is also possible with scripts, although with many scripts exporting and importing is faster.
In order to copy parameter lists, all entries that are in the parameter list manager must be copied first.

4. Creating masks

Copying one or a few masks:
Create the submasks and keyboard masks first of all, followed by the standard masks.
Ensure here that all attributes are the same as in the source project.

Copying many masks:
With this procedure, all mask attributes (size, colors etc.) and names can be transferred!
Close the source and target project. In Windows Explorer copy all *.msk or *.xmsk files (from version 8.0.0 masks are in *.xmsk format) that are required from the source project to the target project.

Note: All assignments of objects are lost.

First transfer all submasks and keyboard masks. Open the target project in GALILEO and move to the overview folder Masks. Move to the masks (Standard) with the right mouse button in the «Project masks» menu. Transfer the masks that you need to the right window (project masks). Now save the target project and open all submasks and keyboard masks in the project. Use Ctrl+A to select all objects on the mask and then delete all objects.
Status → All masks with names are available but are empty.

5. Copying objects in masks

Now copy the objects on the submasks and keyboard masks from the source project to the target project. Always keep the same masks open. The copying takes from mask to mask. Save the target project. Close the target project and open it again. Repeat these steps also with the standard project masks. When objects are copied, images, texts, variable assignments etc. are copied as well if they are present in the project.

4 Colors in GALILEO

4.1 Introduction

In GALILEO it is possible to select the color depth for every panel type (normally 256 or 65536). Bitmaps (images) and colors in objects with color selection are always saved with a maximum color depth of 256 colors. When displaying bitmaps and objects with colors, the question arises which colors will be shown with these images on the panels.

4.2 Purpose

This chapter explains how the GALILEO software treats objects and bitmaps with regard to the colors.

4.3 Colors in objects and bitmaps

4.3.1 Color properties of objects with color selection

With drawn objects and for color selection in objects, it is possible to select the color in objects from the color palette defined in «Config» - «Color palette». This color palette can be changed individually by replacing unused colors with new ones. Alternatively, a completely new color palette can be created. This may be useful if colors are needed that are not available in the existing color palette. However, only one color palette is permitted per project. This means there is a maximum limit of 256 colors.
The objects are shown on the panel in exactly the same way as defined in GALILEO as the color palette is transferred to the panel.

4.3.2 Color properties of the bitmaps

Bitmaps are converted to PCX format during compilation. A color palette with a maximum of 256 colors is created for every individual bitmap. If the color depth of the bitmap is greater than 256 colors, a color palette with 256 colors that is optimized to the bitmap is created during the compilation to PCX format using the color space of the bitmap. The creation of an individual color palette for each bitmap ensures that the color characteristics of the image are retained.
For further information on the PCX format see: http://de.wikipedia.org/wiki/Picture_exchange

4.3.3 Example:

Image properties Original image with a color depth of 60641 colors

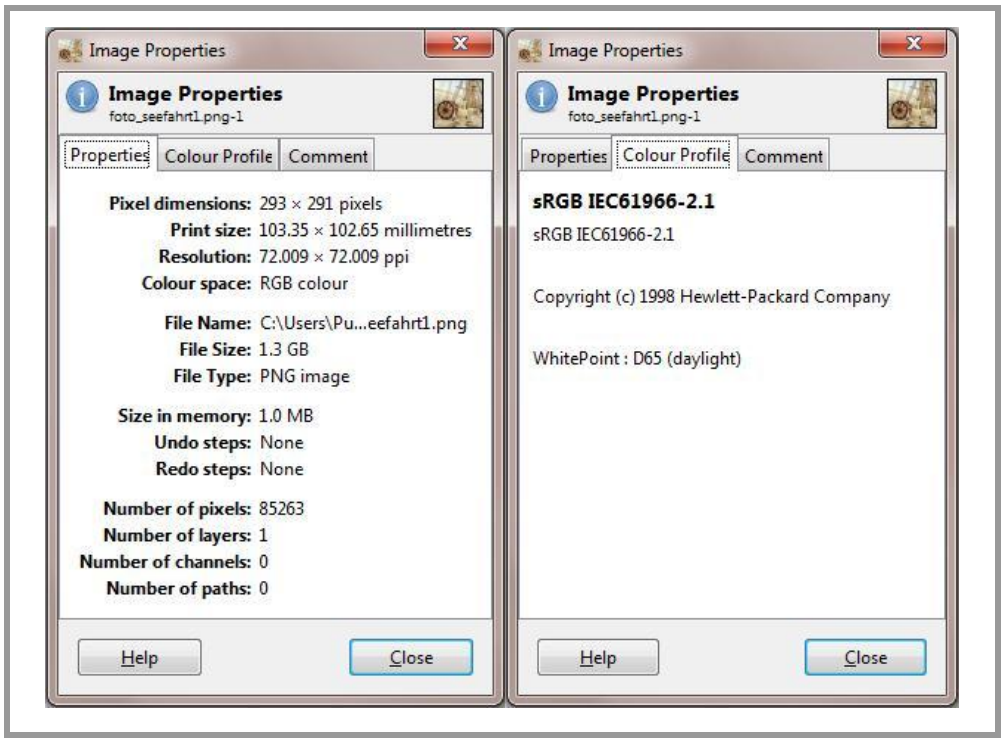


Abb. 5 Image properties, original image

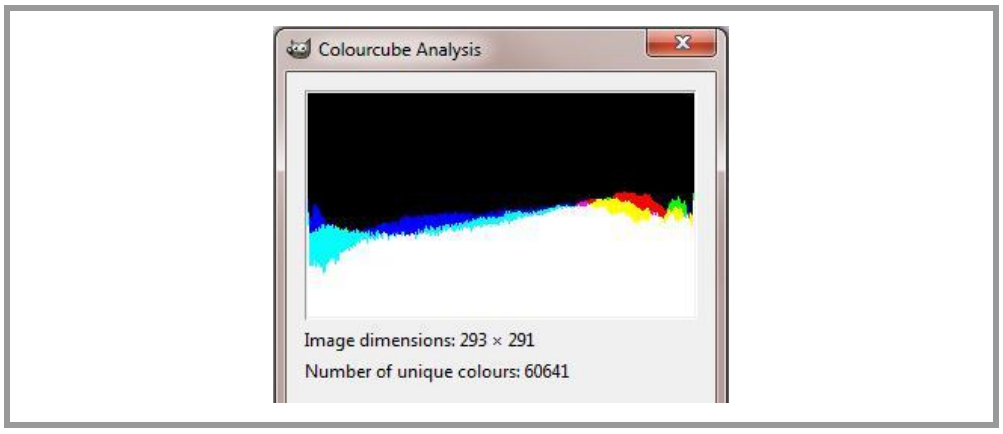


Abb. 6 Colorcube analysis, original image

Image properties after compiling – optimized color palette with 256 colors in PCX format

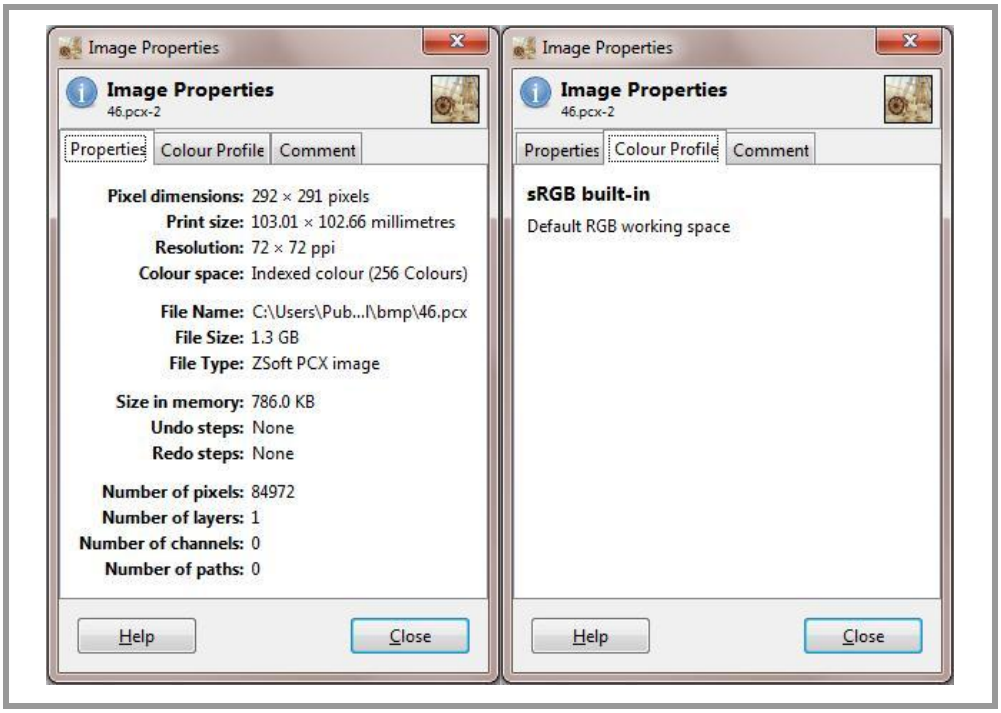


Abb. 7 Image properties, compiled image

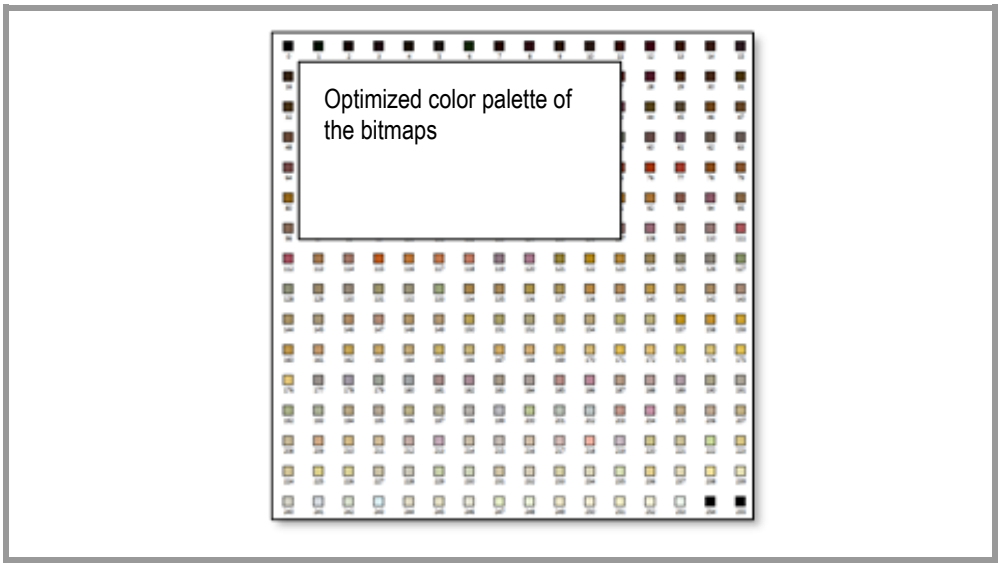


Abb. 8 Color space analysis, original image

The created color palette is then loaded onto the panel with the bitmap.

4.4 Optimized bitmap color palette

The following 2 images and their color palettes show that each color palette is individually created with the colors characteristics for the color space of the image.

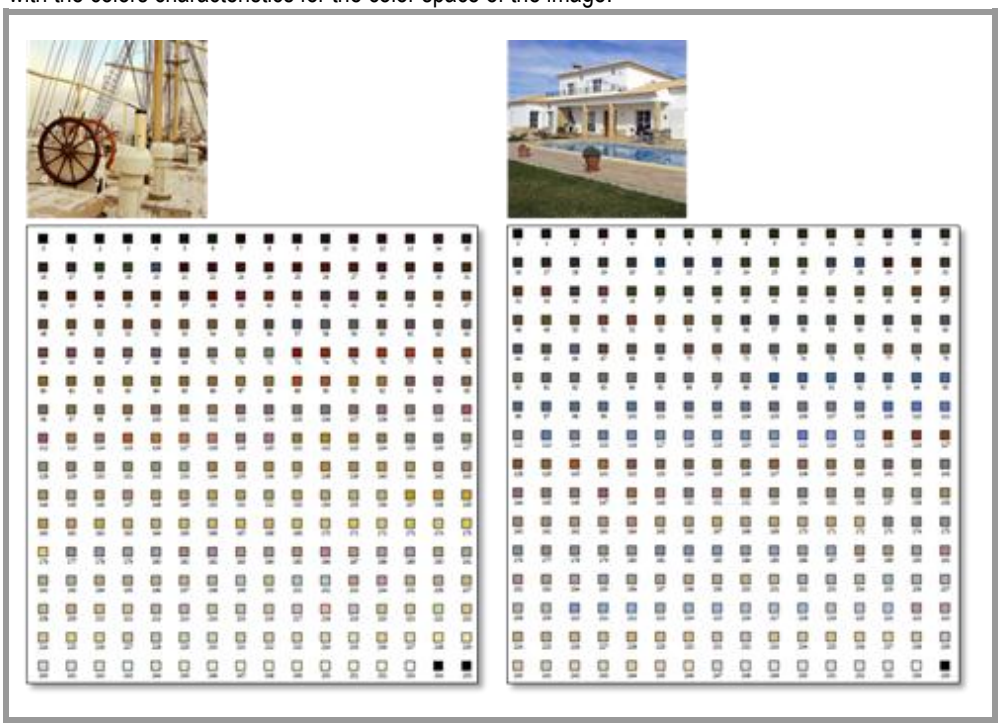


Abb. 9 Optimized color palettes

4.5 Limits

The panel can display a maximum of 65536 colors. This means that the color palette cannot have in all more than 65536 different colors for the objects with color selection and all optimized bitmap palettes. If all images and bitmaps each had 256 different colors in their color palette, a maximum of 255 images in their optimized colors could be shown (65536 / 256 – 1 × color palette for objects with color selection).

4.6 Summary

- 256 colors are available for drawing objects and for the color selection of objects. This color palette can be adapted by the user as required or extended with new colors.
- For each bitmap a separate color palette with 256 colors optimized for the image is created and transferred with the bitmap to the panel. The color characteristics are thus retained for each bitmap (contained in the header of the *.pcx format).
- Also with several images on the same mask, each image has its own optimized color palette.

5 User logger

5.1 Introduction

All user entries, such as data changes or function calls are automatically logged by the User logger when activated. If individual values and functions are not to be logged, lists can be created of tags and functions that are ignored for logging.

Note:
Values that are changed by the controller are not logged.

5.2 Field of application

The user logger can be used for the following:

- Quality assurance
- Statistical evaluation of data
- Troubleshooting problems
- Production logging

5.3 Operation of the user logger

All user entries are logged in a data buffer. The log data is written from the data buffer to a log file as soon as it contains 50 entries or after max. 1 minute.

If the panel is disconnected from the supply voltage when the User logger is activated, all log data in the data buffer is lost. The logger is automatically started on restart.

6 Basic procedure - Logger

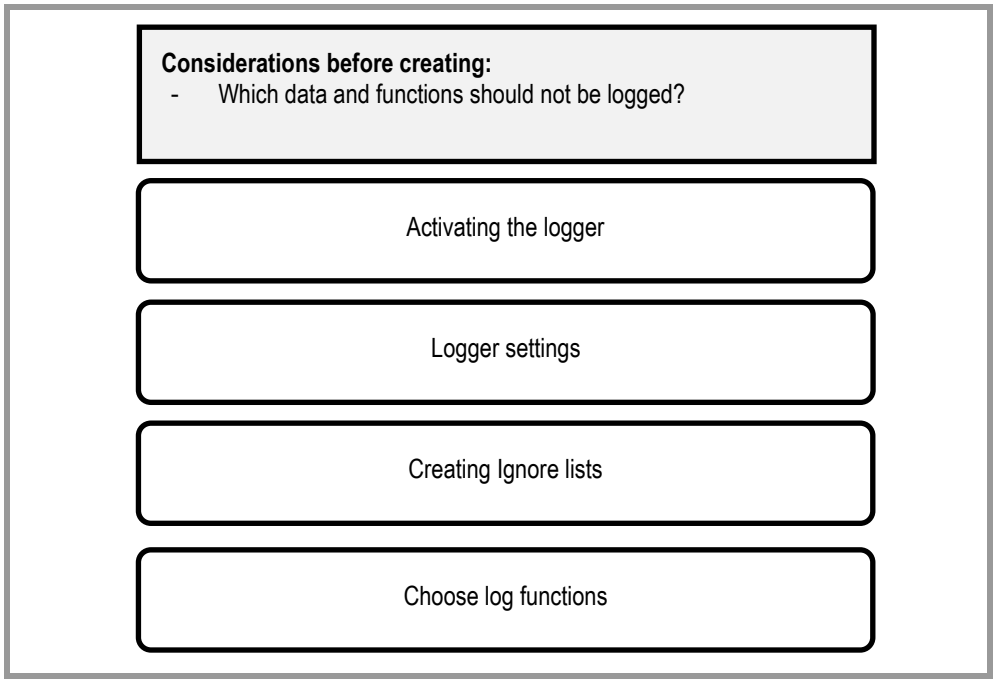


Abb. 10 Basic procedure

6.1 User logger project settings

In the menu «Config» - «Settings…» the window «Project Settings» will be opened.

Within «User Logger» the following settings can be made:

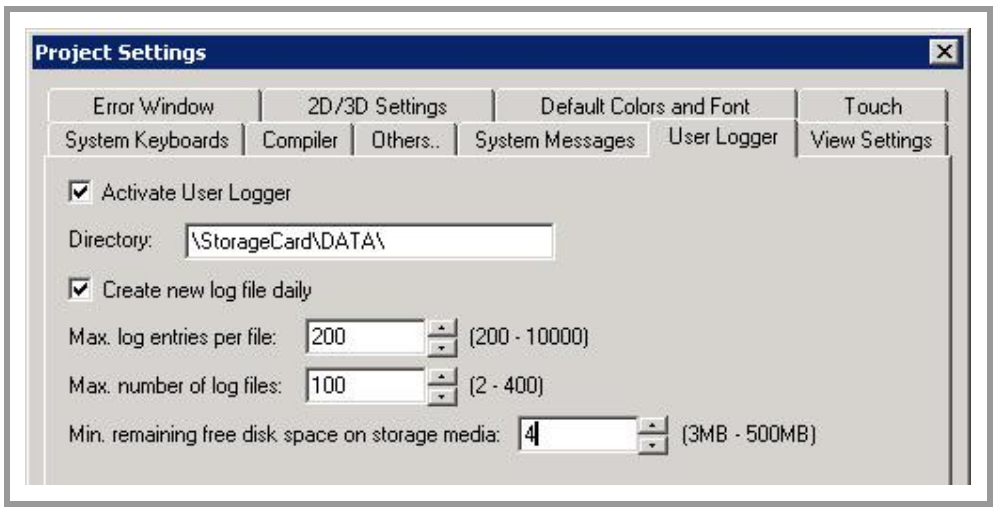


Abb. 11 Project settings

Setting	Description
Activate User Logger	Activates the Logger
Directory	Path where the log files are to be stored. The path must not be the same medium as that on which GRSW is run (operational safety). Otherwise this will be indicated with a system message and the Logger cannot be used.
Create new log file daily	This option ensures that entries of different days are not contained in the same log file. In other words, when the date changes and a new log entry occurs, a new log file is created automatically even if the maximum number of entries has not been reached.
Max. log entries per file	Maximum number of entries per log file. If the maximum number of log entries is reached, a new log file is automatically created and the index is incremented by 1. If a date change has taken place since the last time a log file was created, the index restarts at 0.
Max. number of log files	Maximum number of log files that can be created. If the number is reached, the oldest log file is deleted before a new one is created.

Setting	Description
Min. remaining free disk space on storage media	Minimum free space on storage medium. Must be set for a log file to be created successfully. If the free space is less than required, a system error message will be displayed with every attempt to write a log file. To stop the error message from appearing, sufficient free memory must be provided or the logger must be deactivated.

Tab. 1 Project settings

6.2 Ignore List

The Ignore list is used to stop tags and functions from being logged.

Two types of Ignore lists are possible:

- Tags that are not to be logged

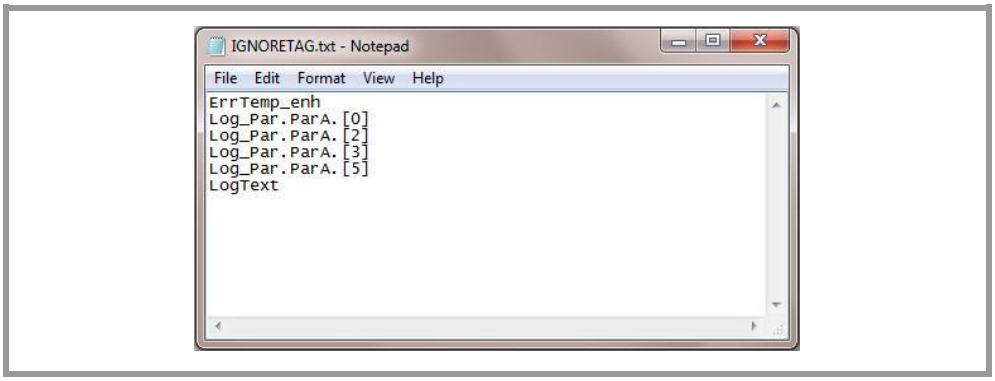


Abb. 12 Ignore List: IGNORETAG.txt

- Functions that are not to be logged

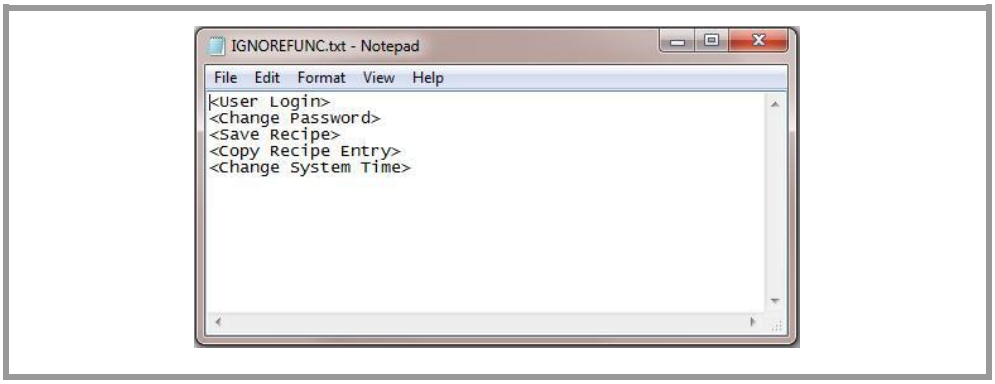


Abb. 13 Ignore List: IGNOREFUNC.txt

The files have fixed names and are called «IGNORETAG.TXT» and „«IGNOREFUNC.TXT», and must be stored in ASCII format. These files must be stored in the folder «Data».

Example:

\StorageCard\DATA

When GRSW is started up, these two files are read (if present) and interpreted.

The tags and functions listed are set to an internal Ignore list so that they are then no longer listed in the log files.

The names must have the same syntax as in the log file, although they are not case sensitive.

Each line must contain 1 entry. Each line must be separated by CR/LF.

6.3 Log file entries

Each log file starts with a header consisting of the following entries:

Log File:	C:\Log\GalileoLog_CONTROLSTATUSDEMO_2012-06-12_014.CSV
Project:	\StorageCard\appl\CONTROLSTATUSDEMO
Program Version:	8.1.0 (12345)

Tab. 2 Log-File Header

This is then followed by the actual log entries.

Timestamp	User Unit	Event	Tag/Function Name	Old Value	New Value
2012-05-23 08:00:30		<Info>	<Startlog>		
2012-05-23 08:05:15		<Function>	<Userlogin>		Müller
2012-05-23 14:20:12	Müller m/min	<Tag Change>	<MachineSpeed>	100	120
2012-05-23 14:20:15	Müller	<Function>	<Userlogin>		Meier
2012-05-23 14:50:23	Meier	<Tag Change>	<ValveV1>	0	1
.....					
.....					

Tab. 3 Log entries

6.4 Special functions

Function button with parameters.

Group:

- User Logger

Function:

- Add String Tag Log
- Add Text Log
- Add Value Tag Log
- Flush Logger
- Logger On/Off
- View Current Log File

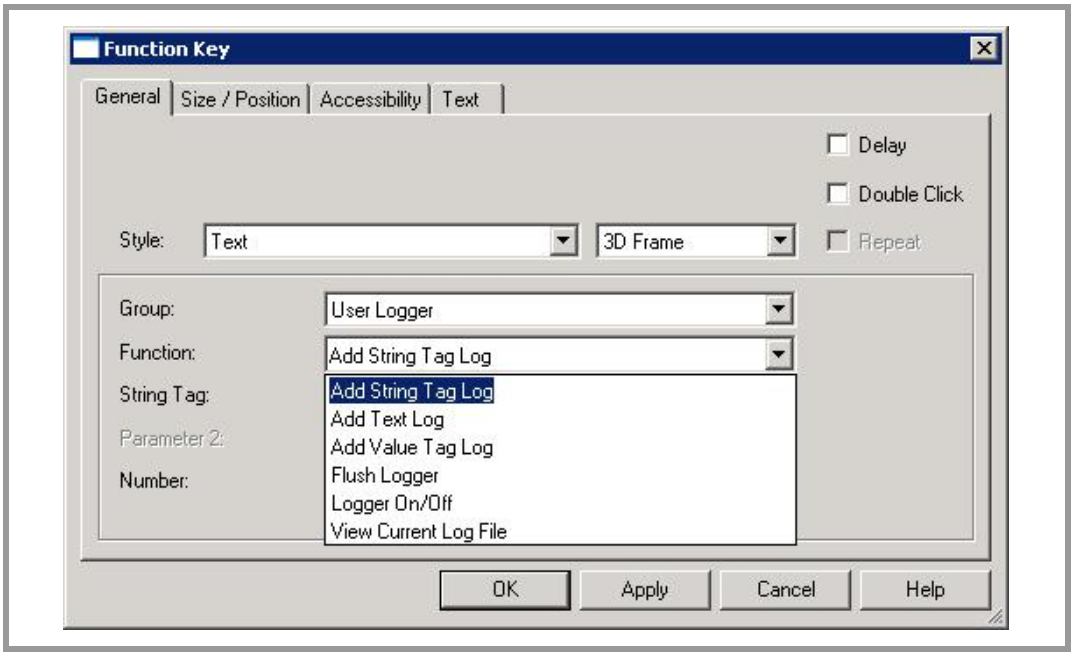


Abb. 14 Special functions

6.4.1 Add String Tag Log (writing String Tags)

The value of a string tag can be written into the protocol file as a log entry.

6.4.2 Add Text Log (writing a predefined character sequence)

A predefined character sequence can be written into the protocol file as a log entry.

6.4.3 Add Value Tag Log (writing Value Tags)

The value of a value tag can be written into the protocol file as a log entry.

6.4.4 Flush Logger (saving of log entries)

The saving of log entries in the data buffer to the log file can be triggered manually

with these functions (e.g. before disconnecting the panel power).

6.4.5 Logger ON-OFF

If logging is started, this is recorded with the entry «Start Logging» in the data buffer.

If logging is stopped, this is recorded with «Stop Logging» in the data buffer and all existing log entries in the data buffer are stored in the log file.

6.4.6 View Current Log File

The current protocol file can be displayed using an external program (e.g. NotepadCE).

In order to use the integrated text editor in Windows CE 5.0 (from version 2.26.1), the following parameter can be indicated:

- NotepadCE.exe