

# TWN4

## AppBlaster User Guide

DocRev14, October 28, 2025



ELATEC GmbH

# Contents

1	Overview . . . . .	4
2	System Requirements . . . . .	5
3	Installation . . . . .	6
4	Starting AppBlaster . . . . .	7
5	Programming Firmware Image . . . . .	8
6	Read Version of Connected USB Devices . . . . .	9
7	Programming Firmware of BLE Module . . . . .	10
8	Configurable Project . . . . .	12
8.1	Step 1: Select an Application Template . . . . .	13
8.2	Step 2: Add Transponder Types . . . . .	15
8.2.1	Remove Transponder Types . . . . .	16
8.3	Step 2a: Specify Data Source . . . . .	17
8.4	Step 2b: Specify Bit Manipulation . . . . .	18
8.5	Step 2c: Specify Output Format . . . . .	19
8.6	Step 3: Specify Prefix, Delimiter, Suffix (Optional) . . . . .	20
8.7	Step 4: Signalling/Behaviour (Optional) . . . . .	22
8.8	Step 5: Specify Options (Optional) . . . . .	23
8.9	Step 6: Specify Security Information . . . . .	24
8.10	Step 7: Specify Version Information (Recommended) . . . . .	25
8.11	Step 8: Create Image . . . . .	26
8.12	Step 9: Program Image . . . . .	27
8.13	Step 10: Save Project (Recommended) . . . . .	28
9	Configurable Project (PAC, OSDP, Wiegand) . . . . .	29
9.1	PAC Settings - General . . . . .	30
9.1.1	General - Mode of Operation . . . . .	30
9.1.2	General - Properties . . . . .	30
9.1.3	General - LED Mapping . . . . .	31
9.2	PAC Settings - OSDP . . . . .	32
9.2.1	OSDP - Properties . . . . .	32
9.2.2	OSDP - Security . . . . .	32
9.2.3	OSDP - Local Event Handling . . . . .	33
9.3	PAC Settings - Wiegand . . . . .	34
9.3.1	Wiegand - Local Event Handling . . . . .	34
9.3.2	Wiegand - Interface . . . . .	34
9.4	Touchpad & Backlight . . . . .	35
9.4.1	Settings . . . . .	35
9.4.2	Output Format (Wiegand) . . . . .	35
10	Source Code Project . . . . .	36
10.1	Step 1: Specify Type of USB . . . . .	37
10.2	Step 2: Specify Source Code Module . . . . .	37
10.3	Step 3: Specify Options (Optional) . . . . .	38
10.4	Step 4: Specify Security Information . . . . .	39
10.5	Step 5: Specify Version Information (Recommended) . . . . .	40
10.6	Step 6: Create Image . . . . .	41

10.7 Step 7: Program Image . . . . .	41
10.8 Step 8: Save Project (Recommended) . . . . .	41
11 Creation of Apps with make . . . . .	42
12 Setting Up AppBlaster . . . . .	43
13 Disclaimer . . . . .	44

# 1 Overview

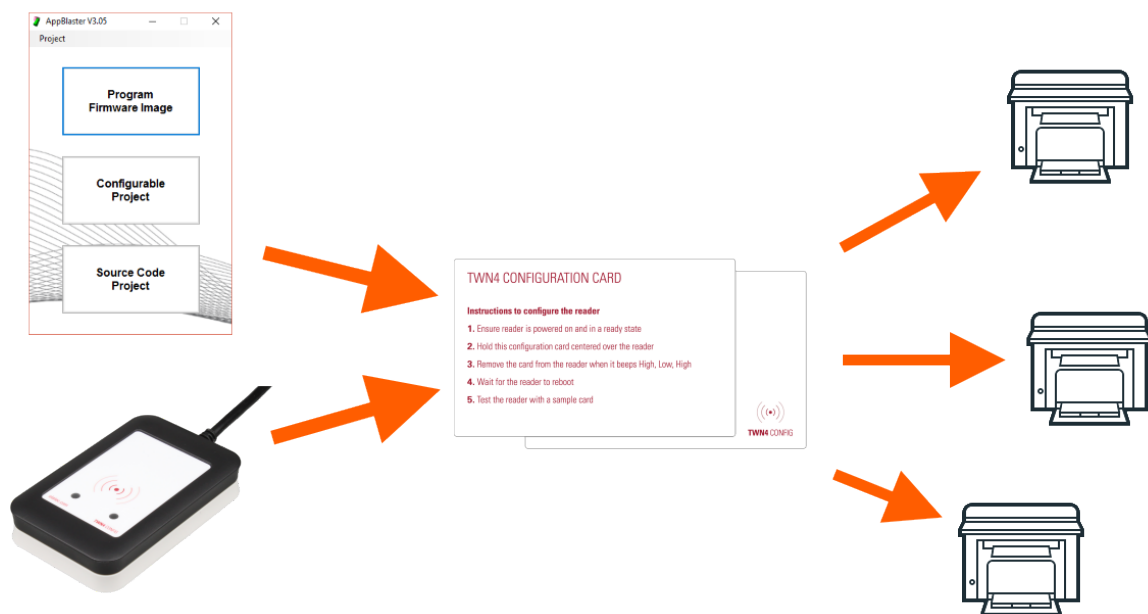
You received the AppBlaster as part of the developer pack for TWN4, which is distributed as a zip file.

AppBlaster is a program, which allows to configure and program TWN4 for a specific application.

There are three ways to prepare TWN4 for operation:

1. Directly program an appropriate firmware image into TWN4
2. Interactive configuration of TWN4 with AppBlaster
3. Write an App for TWN4 in programming language C

Moreover, TWN4 now can be configured with configuration cards via RF interface:



For usage of configuration cards, please see separate document "TWN4 Configuration Cards".

## 2 System Requirements

There are the minimum system requirements for a serious use of the TWN4 developer pack :

- Operating system: Microsoft Windows 7 or later, 32 or 64 bit
- Microsoft .NET Framework 4.7.2
- Processor (CPU): 2 GHz
- Hard Disk: 100 MB
- RAM: 4 GB

## 3 Installation

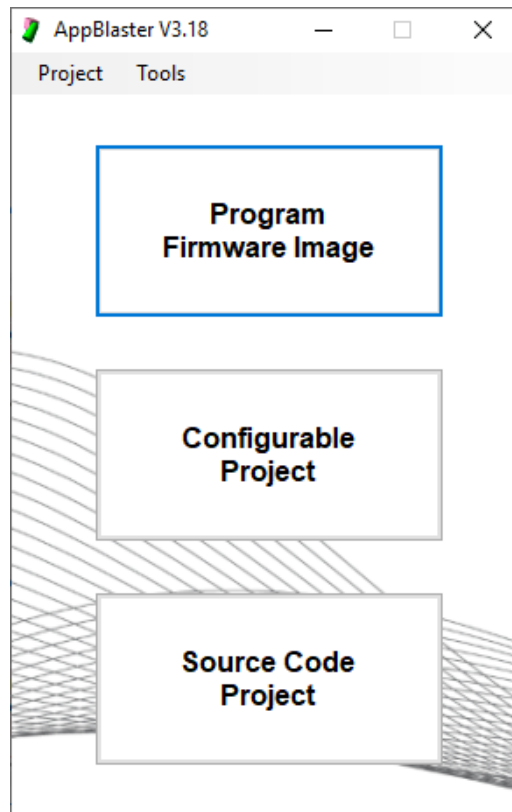
You received AppBlaster as part of the developer pack for TWN4, which is distributed as a zip file.

In order to install the package, please follow these steps:

- Create an empty directory on your hard disk
- Unzip the entire content of the zip file into this empty directory
- The program AppBlaster can be found in the top directory of the TWN4 developer pack.

## 4 Starting AppBlaster

In order to start AppBlaster, move to base directory of the installed TWN4 developer pack and execute program AppBlaster.exe. The start dialog appears:



The start dialog offers three choices.

Depending on the task being achieved, click appropriate button:

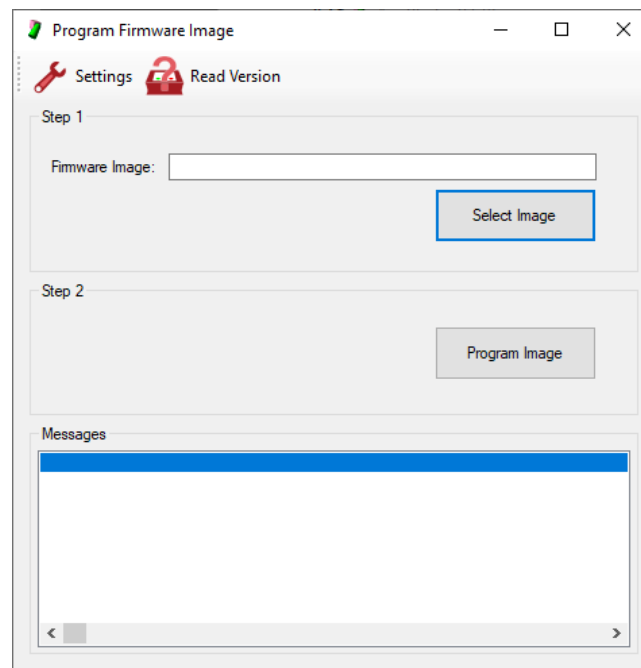
- Button "Program Firmware Image": Program TWN4 with a firmware image
- Button "Configurable Project": Start a new project for TWN4, where setup is done via an interactive configuration.
- Button "Source Code Project": Write your App for TWN4 in C programming language.

Additionally, an existing project can be loaded from disk via selection in the menu.

- Menu "Project - Load": Load a previously saved project. This might be either a configurable project or a source code project.

## 5 Programming Firmware Image

If you got a compiled firmware image, there is the possibility to program such files from within AppBlaster into a TWN4 device. In order to do this, click button "Program Firmware Image" in the start dialog. Another dialog opens:



Follow these steps:

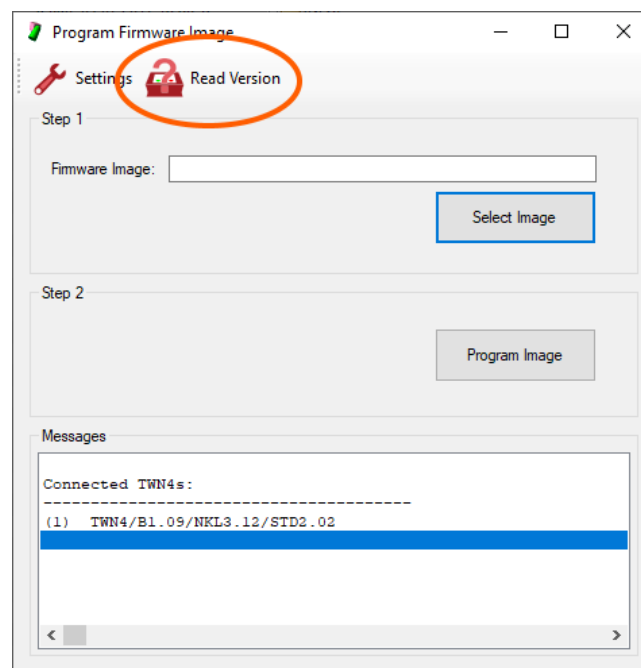
1. In "Step 1" choose the file of interest by clicking button "Select Image". Another way is to enter the known path and file name directly into the text box. The file extension for appropriate firmware images is ".bix".
2. In "Step 2" click button "Program Image". This will immediately start the programming sequence. After a few seconds, the step will be completed.



## 6 Read Version of Connected USB Devices

Following steps:

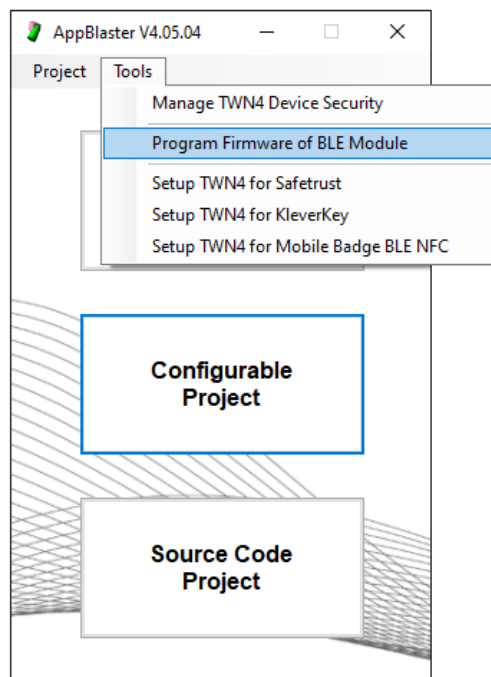
1. In the start dialog click button "Program Firmware Image". The dialog "Program Firmware Image" appears.
2. Click "Read Version" from the tool bar. AppBlaster lists all TWN4, which can be found on the USB bus.



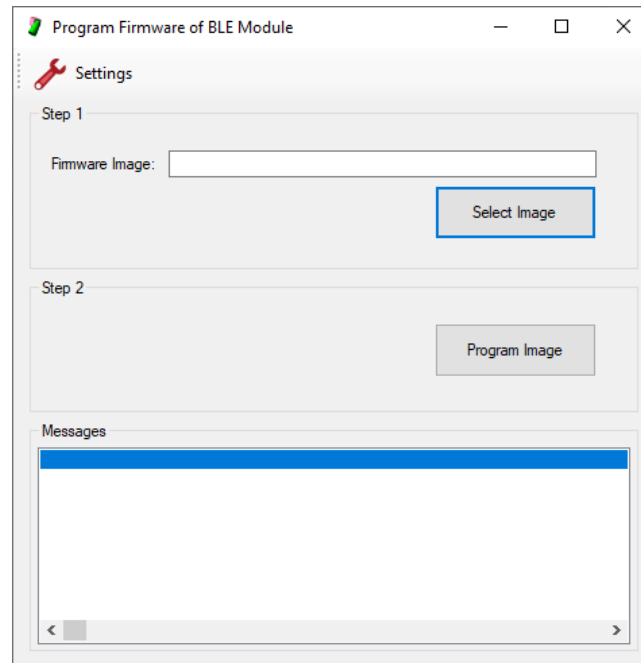
## 7 Programming Firmware of BLE Module

Following steps:

- In the start dialog select Tools - Program BLE Module. A dialog opens, which guides through the next steps.



- In "Step 1" choose the file of interest by clicking button "Select Image". Another way is to enter the known path and file name directly into the text box. The file extension for appropriate BLE firmware images is ".gbl".
- In "Step 2" click button "Program Image". This will immediately start the programming sequence. After a few seconds, the step will be completed.

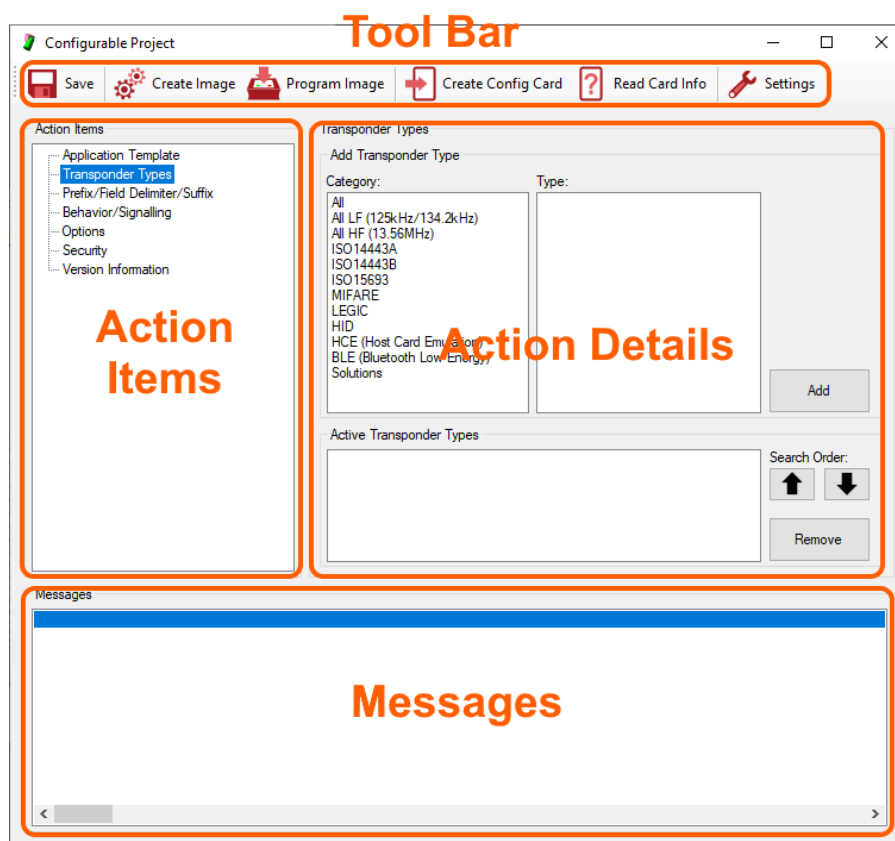


## 8 Configurable Project

In order to create a configurable project, click button "Configurable Project" in the start dialog. Another dialog appears, the project dialog.

This project dialog is separated into several sections, which allow user input and do output from the program:

- Tool Bar: Buttons in the tool bar let you start specific actions at any time
- Action Items: There are several action items, which can or must be done to complete a project
- Action Details: For every action item, there are details which can or must be entered
- Messages: AppBlaster is giving feedback on actions, which are started from the tool bar. Furthermore, error messages are displayed in this area.



Once a configured project has been started, there are several steps, which must or can be followed.

## 8.1 Step 1: Select an Application Template

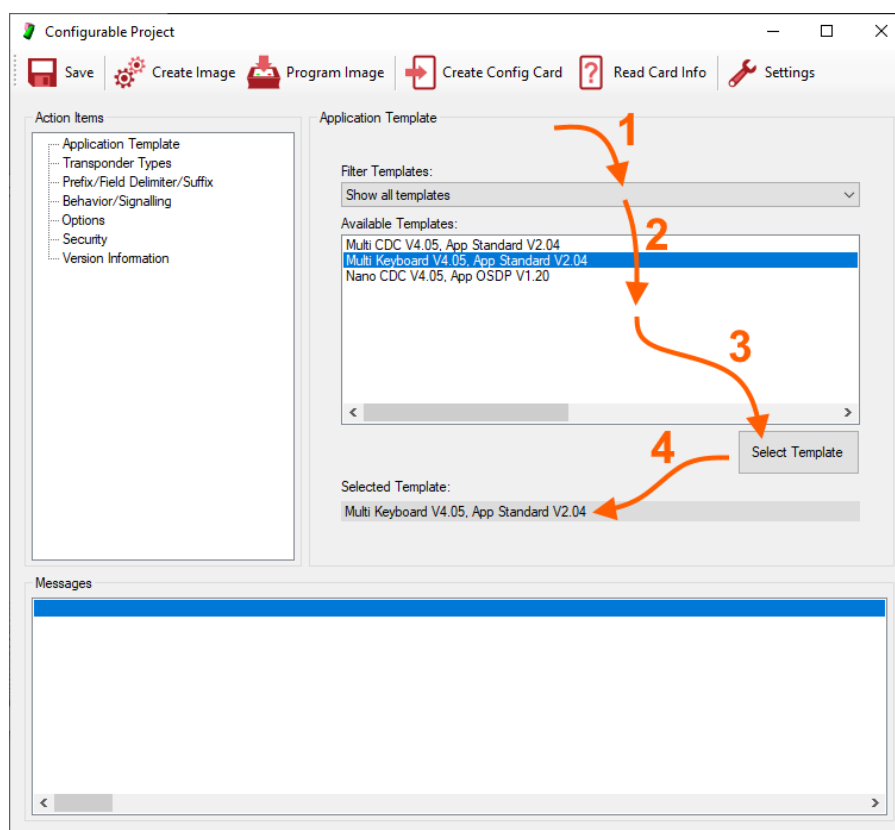
AppBlaster is delivered with several Application Templates ("template"). A template defines the type of application to be created. The selection of the template has an impact on the supported hardware and functionality. Even though, the selection must be done carefully, AppBlaster makes the choice of appropriate template easier, because the templates supplied with the latest developer pack do not distinguish between the supported type of hardware.

A template, which supports several types of hardware is called MultiBIX-template. It can be recognized by the word "Multi" in the file name instead of "Core", "Mini" or "Nano". If a template can only support a single type of hardware, it is still delivered as a normal template, e.g. "Nano" for the OSDP template, because the entire TWN Palon series is based on the Nano-type of hardware.

Notes:

- If a MultiBIX-template is selected, the resulting firmware image is also a MultiBIX firmware image and therefore can be installed on multiple types of TWN4 hardware.
- Please contact ELATEC GmbH, if non-MultiBIX-templates are required for special purposes, e.g. programming TWN4 with older tools, which cannot be replaced easily.

The list of available templates can be filtered by the support of a specific type of hardware. This is accomplished by the selection in the combo box "Filter Templates".



1. Select appropriate template in the list box "Available Templates". Currently, there are three templates available:

- USB CDC (virtual COM port) with configurable standard app

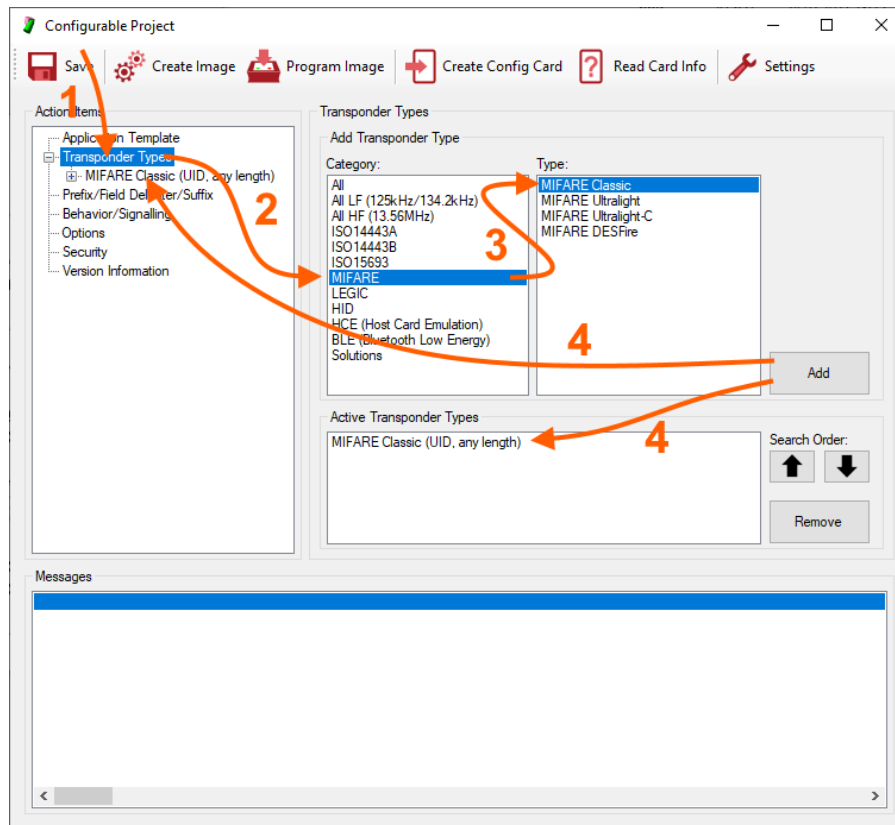
- USB Keyboard with configurable standard app
  - USB CDC with configurable OSDP stack (typically programmed via USB and normal operation with built-in RS-485 interface).
2. Double click template or click button "Select Template"
  3. The selected template is displayed in a text box.

Note:

- If you would like to operate TWN4 via RS-232, selecting CDC or keyboard makes no difference. RS-232 is supported by both types of templates identically.

## 8.2 Step 2: Add Transponder Types

In your specific application, one, two or even more types of transponders might be involved. Here are the typical steps through the dialog in order to add one type of transponder:



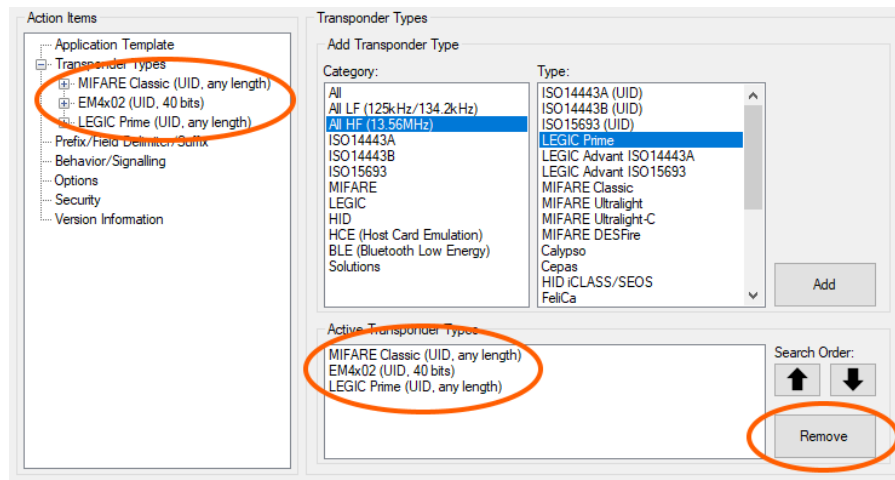
Following (sub-)steps:

1. Select "Transponder Types" from "Action Items".
2. The list box "Category" divides all technologies ("types") supported by TWN4 into several criteria. A supported technology can be listed under several positions depending on their properties. E.g. "MIFARE Classic" can be found under "All", "All HF (13.56MHz)", "ISO14443A" and "MIFARE".
3. Selecting a specific category lists all available technologies belonging to that category in the list box "Type".
4. Once the wanted type is selected, the button "Add" can be clicked. As a result, the selected type is added at two places in the dialog:
  - a) In the list of "Active Transponder Types"
  - b) In the list of "Transponder Types in "Action Items"

Continue with item 2, if the application requires more types of transponder to deal with.

### 8.2.1 Remove Transponder Types

If a transponder type was added unintentionally, it can be removed.



1. Select "Transponder Types" from "Action Items".
2. In the list box "Active Transponder Types" select transponder type to be removed.
3. Finally, click button "Remove" to remove the transponder type in question.

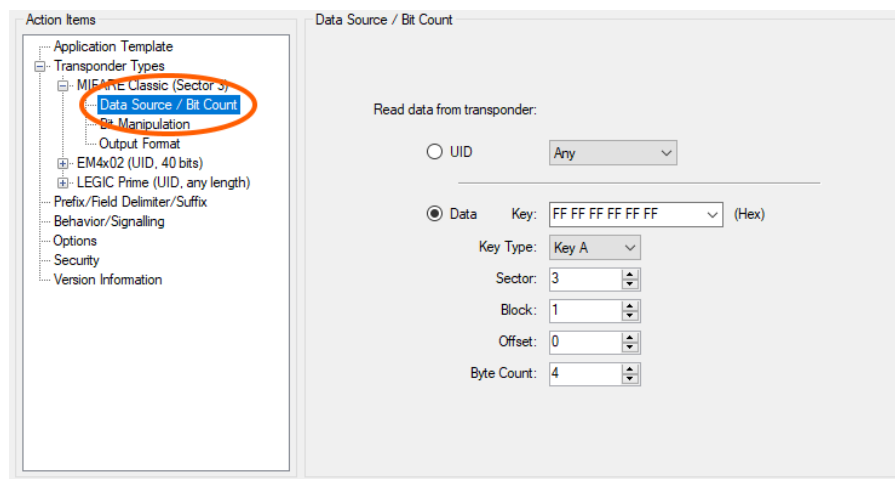


### 8.3 Step 2a: Specify Data Source

The data source specifies the place on a transponder, where data in question is stored. Depending on the type of transponder such storage places are called UID, sector, block, segment, page, file or something similar. In many cases, simply the UID should be read ("data source") from the transponder. It is the default setup, once a transponder type is added.

The specification of the data source depends on the selected type of transponder. In many cases and by default, the UID is treated as data source.

Here is an example, how a block within a sector of MIFARE Classic can be used as data source:

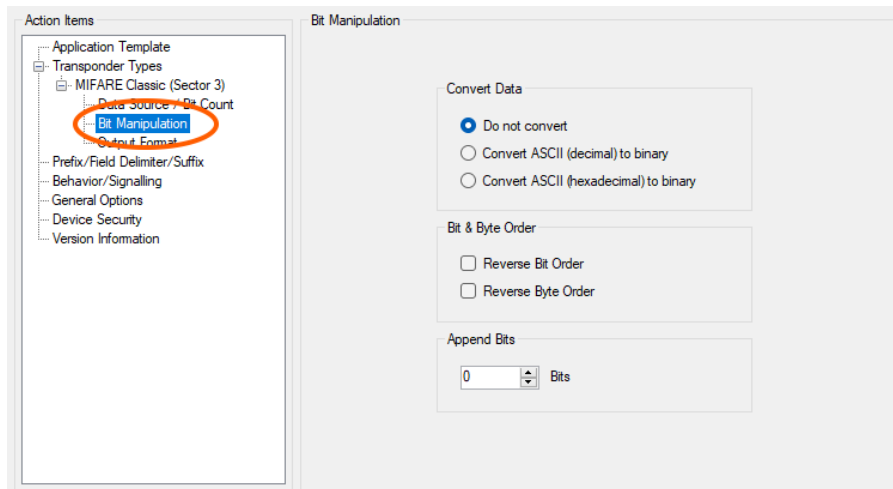


- Instead of reading UID choose "Data", which means read data from a specific block in a sector of MIFARE Classic.
- Specify the key, which gains access to the sector in question. There are some well-known default keys, which can be selected directly from the combo box.
- Select the sector number (0 to 15) in question.
- Select the block number (0 to 3) within the sector.
- Select offset within the block, where the data is stored.
- Finally, select the number of bytes, which data in question occupies. Depending on the specified offset, this can be up to 16 bytes (the size of a MIFARE block).

**Notes:**

- For other types of transponders, the specification of the data source can be different but works in similar manner.
- Many transponders do allow specification of UID as the only source of data.

## 8.4 Step 2b: Specify Bit Manipulation



### Convert data:

Do not convert: Data remains unchanged

Convert ASCII (decimal) to binary: Data (the bytes read from the transponder) is interpreted as a decimal number in ASCII format:

Example: "1234" (0x31 0x32 0x33 0x34) => 0x00 0x00 0x04 0xD2

Convert ASCII (hexadecimal) to binary: Data (the bytes read from the transponder) is interpreted as a hexadecimal number in ASCII format:

Example: "12AB" (0x31 0x32 0x41 0x42) => 0x00 0x00 0x12 0xAB

Note: Data must be a multiple of 8 bits, in other words a sequence of bytes.

### Reverse Bit Order:

Example: 0x12 0x34 => 0x2C 0x48

### Reverse Byte Order:

Example: 0x12 0x34 => 0x34 0x12

### Append Bits:

Example, append 16 bits: 0x12 0x34 (16 bits) => 0x12 0x34 0x00 0x00 (32 bits)

## 8.5 Step 2c: Specify Output Format

The screenshot displays the 'Output Format' configuration interface. On the left, the 'Action Items' tree shows a hierarchy starting with 'Application Template' and 'Transponder Types'. Under 'MIFARE Classic (Sector 3)', the 'Output Format' item is highlighted with a red circle. The main panel on the right is titled 'Output Format' and contains settings for 'Field 1'. The 'Output Bits' section has two radio buttons: 'All Bits' (selected) and 'Some Bits'. Below these are input fields for 'First Bit' (set to 0) and 'Number of Bits' (set to 32). The 'Output Format' section has four radio buttons: 'Binary', 'Octal', 'Hexadecimal' (selected), and 'ASCII'. A 'Prefix' section contains an empty text box. The 'Length of Output' section has three radio buttons: 'Automatic' (selected), 'Minimum', and 'Exact', followed by a 'Digits' input field set to 0. On the right side of the panel, there are 'Move Field' buttons (left and right arrows), an 'Add Field' button, and a 'Remove Field' button.

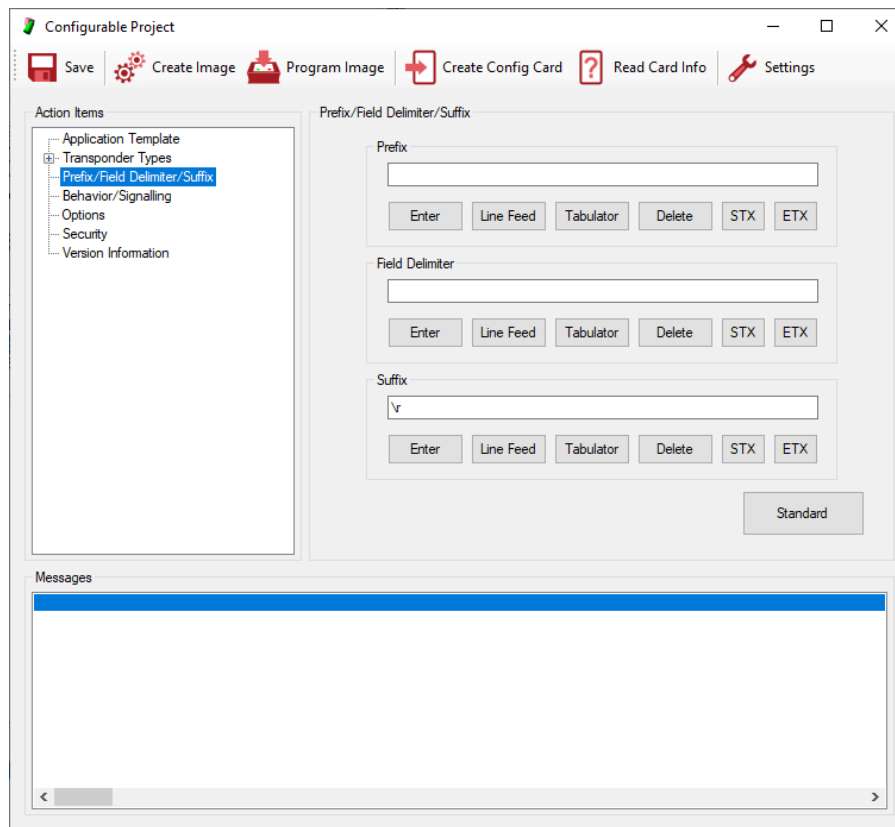
## 8.6 Step 3: Specify Prefix, Delimiter, Suffix (Optional)

AppBlaster allows to specify prefix, delimiters and suffix for the output data. There are several special characters available, which might be helpful for the specific project. By default, there is no prefix sent, no delimiters and "Enter" is sent as the suffix. Delimiters separate fields specified in the output format. Two or more fields are needed for a delimiter being sent.

Note: Due to the nature of AppBlaster, which passes the data entered in this dialog to a C compiler, all non-printable characters must be entered with a backslash as preceding character. Following escape sequences are supported:

Escape Sequence	Hex Value	Character Represented
\a	0x07	Alert (Beep, Bell)
\b	0x08	Backspace
\e	0x1B	Escape character
\f	0x0C	Formfeed Page Break
\n	0x0A	Newline (Line Feed)
\r	0x0D	Carriage Return
\t	0x09	Horizontal Tab
\v	0x0B	Vertical Tab
\\	0x5C	Backslash
\'	0x27	Apostrophe or single quotation mark
\"	0x22	Double quotation mark
\nnn	any	The byte whose numerical value is given by nnn interpreted as an octal number
\xhh...	any	The byte whose numerical value is given by hh... interpreted as a hexadecimal number. The sequence of hexadecimal numbers is terminated by a non-hexadecimal character or end of the string

Source: [Wikipedia, Escape sequences in C](#)



## 8.7 Step 4: Signalling/Behaviour (Optional)

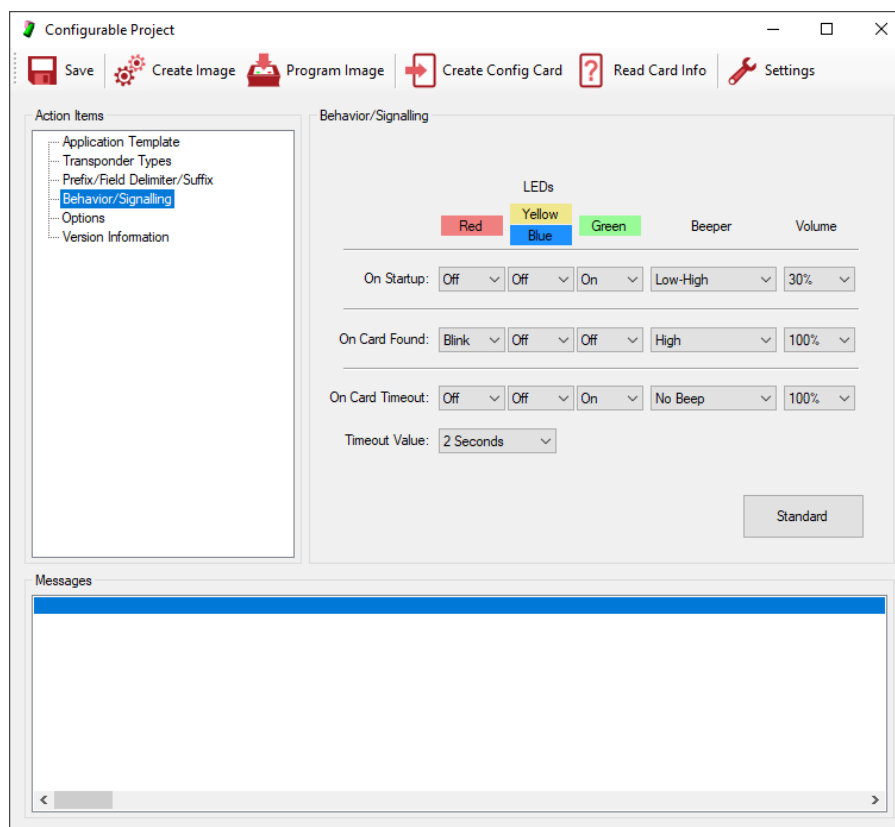
During operation of TWN4, there are several types of events, where action can take place. These events are

- **Startup**  
When TWN4 is turned on or performs a reset, the event occurs.
- **Found Card/Transponder**  
When a transponder enters the RF field of TWN4, the event occurs.
- **Card/Transponder Timeout**  
When a transponder leaves the RF field for a specific time, the event occurs. The timeout value can be adjusted.

By default, following actions take place:

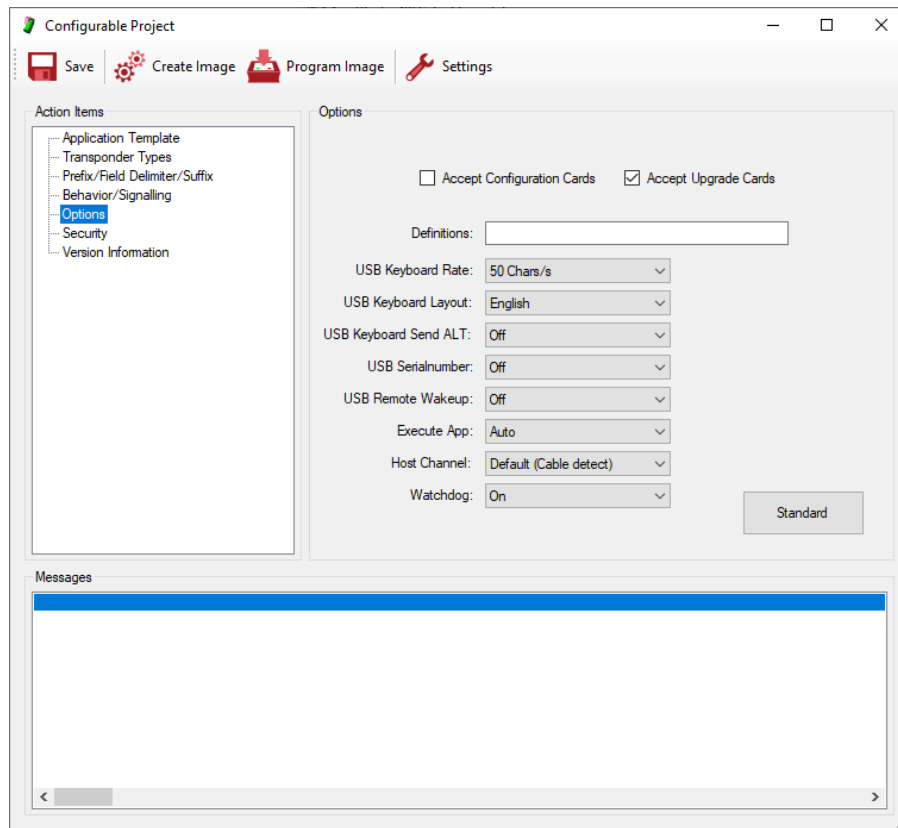
- **Startup**  
Turn off red LED, turn on green LED, sound a low/high beep at 30% volume
- **Found Card/Transponder**  
Let red LED blink, turn off green LED, sound a high beep at 100% volume
- **Card/Transponder Timeout**  
Turn off red LED, turn on green LED. The timeout value is set to 2 seconds.

The behaviour can be adjusted to specific requirements coming from the application. A simple modification could be to suppress any output from the beeper. In this case, select "No Beep" for all three types of events.



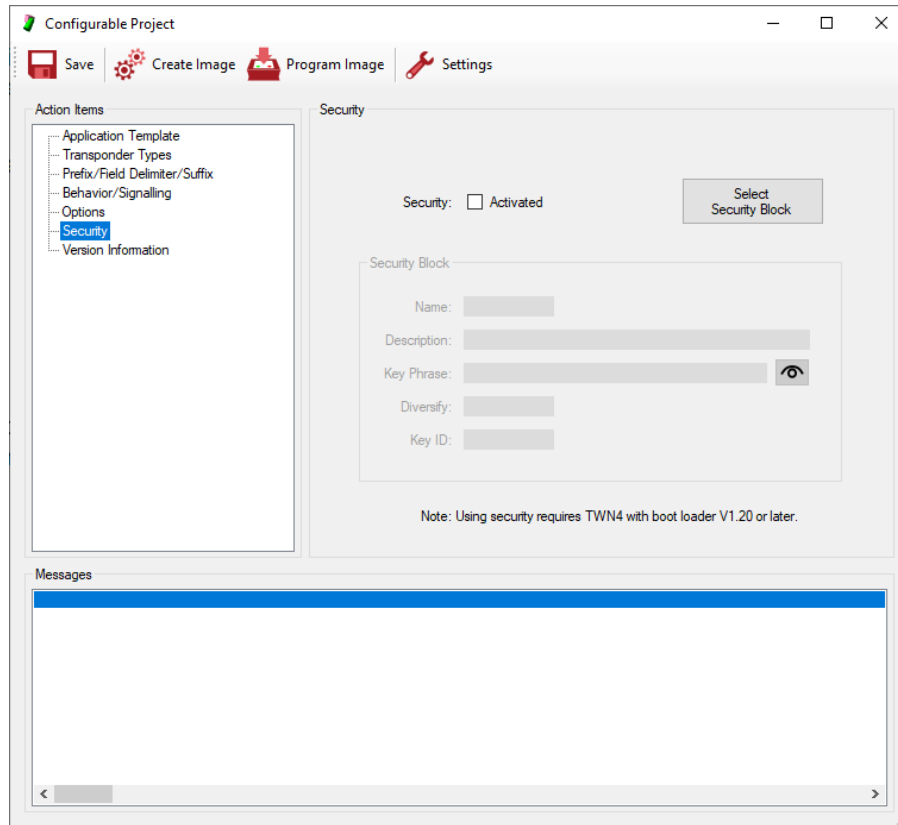
## 8.8 Step 5: Specify Options (Optional)

In this section some special requirements for the application can be set up.



## 8.9 Step 6: Specify Security Information

With AppBlaster V4, the concept of TWN4 Device Security is introduced. There is a separate document "TWN4 Device Security", which explains the subject in detail.





## 8.10 Step 7: Specify Version Information (Recommended)

It is recommended to use a standard naming scheme for the firmware image. The information contained in the file name of the firmware image will be used at several places and allows recognition of correct version of a firmware image or TWN4.

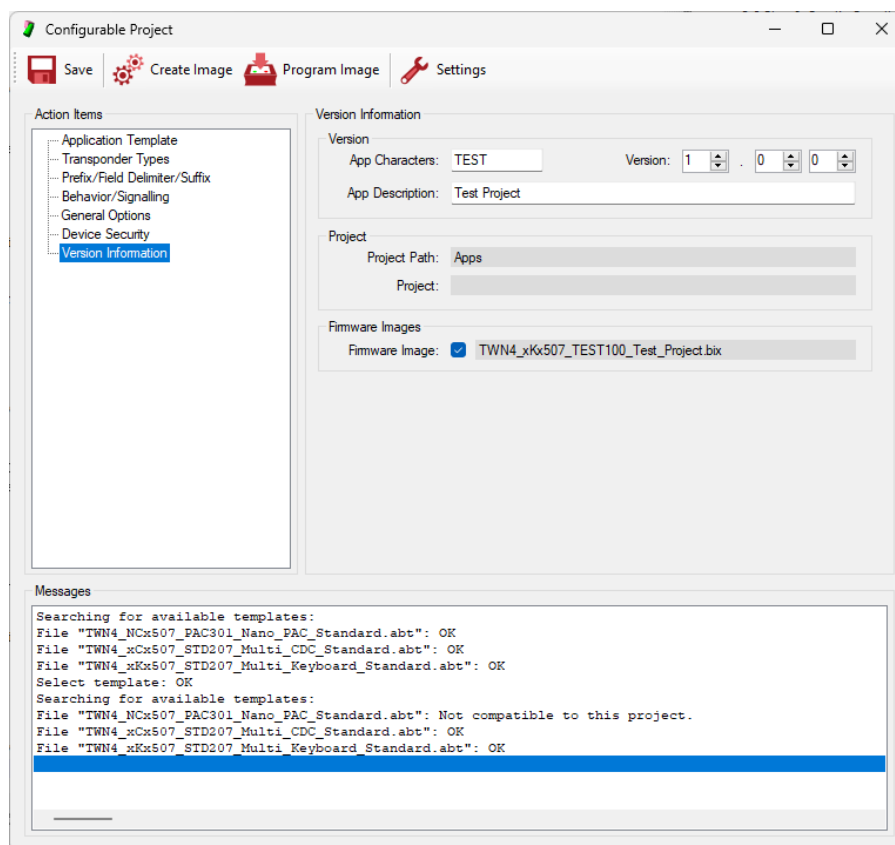
Version information of TWN4 is therefore identical under following conditions:

- The name of the firmware image file itself.
- Information contained on the label of TWN4.
- Version information returned by the application running on TWN4. This is applicable for firmware, which allows request for the firmware version and is achieved via the system function `GetString`.
- Version information returned by USB as product string

AppBlaster supports the naming scheme by offering a dialog, where appropriate information can be entered:

- App Characters: These characters are 1 to 4 letters ('A'-'Z') or digits ('0'-'9'). Please use at least 2 characters which identify the project.
- Version: The version always consists out of 3 digits, which can be set up here.
- App Description: You may do a short description of the project here.

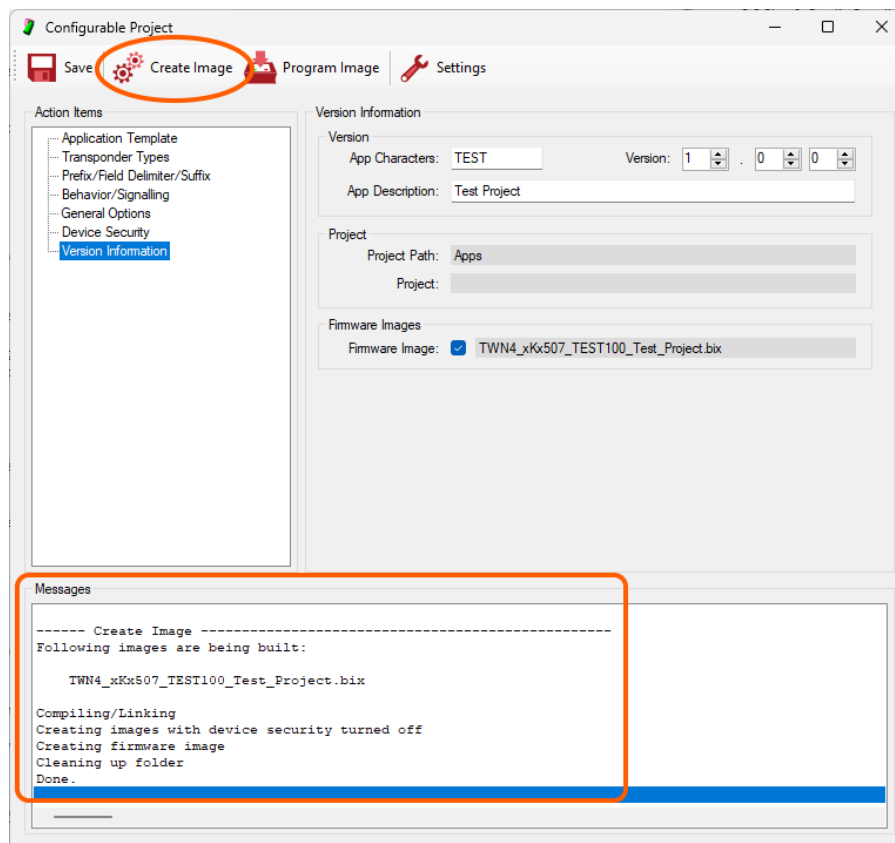
Note: In front of every image which can be created there is a check box, which allows you to turn generation of that image on and off. For non-PAC images there is only the option to select a single image. For PAC images, there are choices to create images for several purposes.



## 8.11 Step 8: Create Image

Before a configuration can be tested, the firmware image must be created.

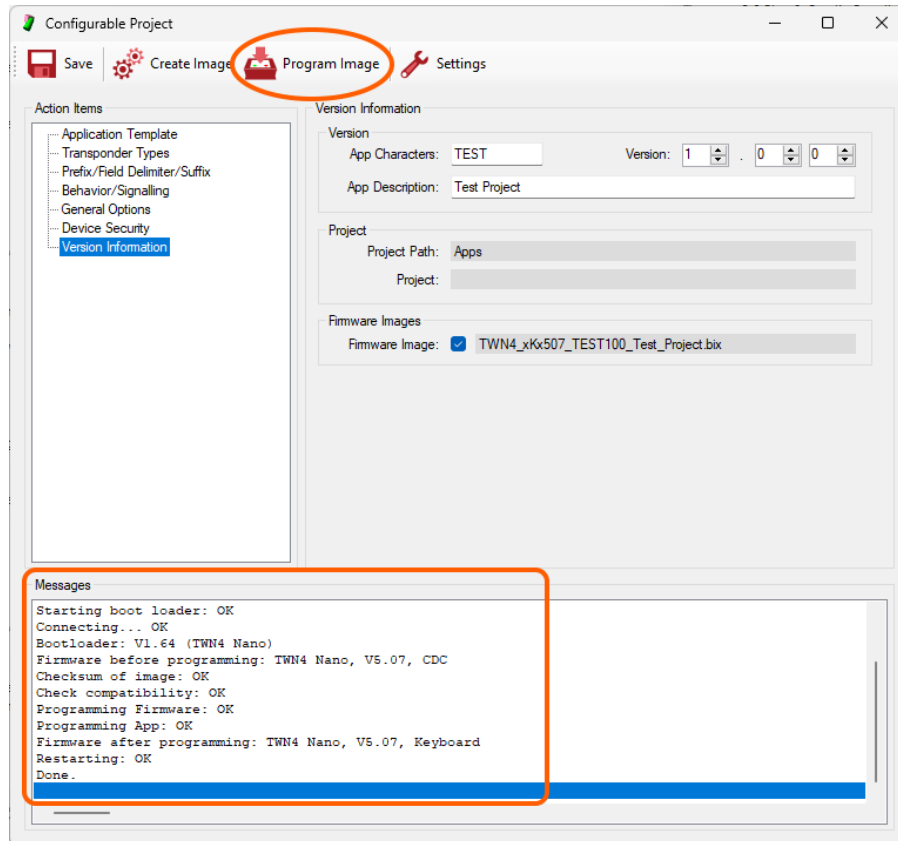
In order to create the firmware image, click button "Create Image". If the project never was saved before, the project will be created in the sub directory "Apps". Otherwise, it will be created, where the project file resides.



## 8.12 Step 9: Program Image

If the firmware image was created successfully it can be programmed into TWN4.

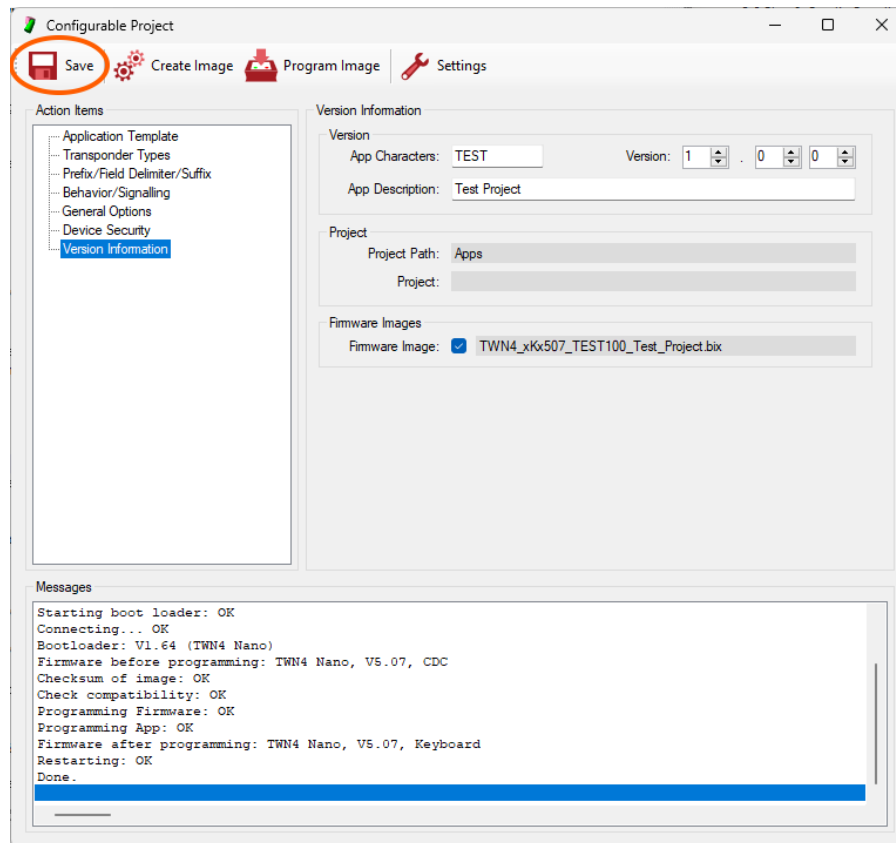
In order to program firmware image, click button "Program Image".



### 8.13 Step 10: Save Project (Recommended)

After finishing work, you may save entire information in a project file. Compared to the production image, the project file allows to modify settings of the configuration later, the production image does not.

As a suggestion, AppBlaster will choose the name for the project file identical to the name of the production image.



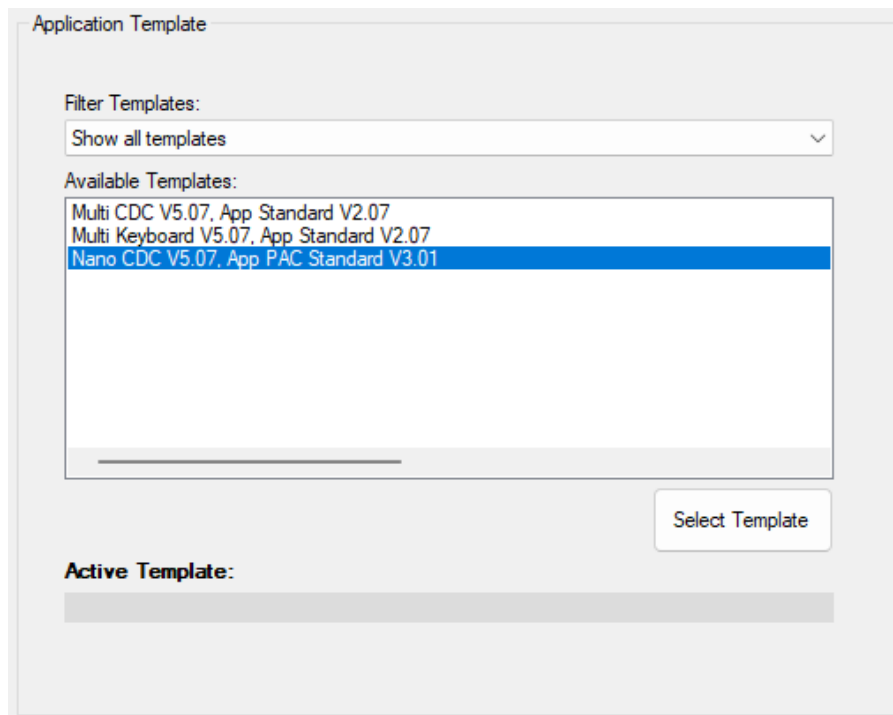
## 9 Configurable Project (PAC, OSDP, Wiegand)

The chapter above explained, how to configure a standard App. Additionally, there is the possibility to create an App for physical access (PAC). This means the reader is working in either one of the two modes OSDP or Wiegand.

Please note that the signalling is configured in a different way, because it can also be under the control of the host (controller).

Even though, it is possible to install a PAC application on other reader models, it is only supported by devices, which are part of the TWN4 Palon and Secustos family.

To create an PAC App please select a template:

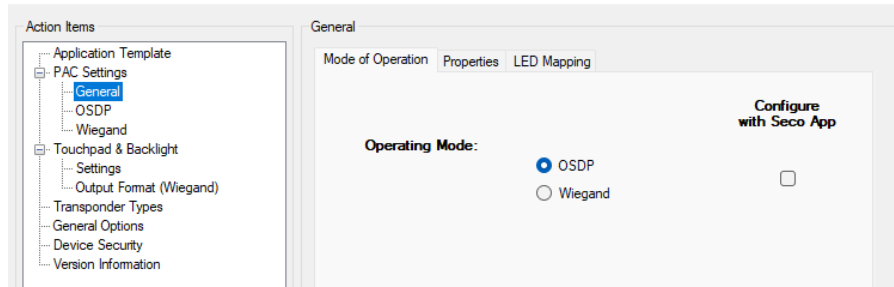


The screenshot shows a software window titled "Application Template". Inside, there is a "Filter Templates:" section with a dropdown menu currently showing "Show all templates". Below this is an "Available Templates:" list box containing three entries: "Multi CDC V5.07, App Standard V2.07", "Multi Keyboard V5.07, App Standard V2.07", and "Nano CDC V5.07, App PAC Standard V3.01". The third entry is highlighted with a blue background. To the right of the list box is a "Select Template" button. At the bottom of the window, there is an "Active Template:" label followed by an empty text field.

## 9.1 PAC Settings - General

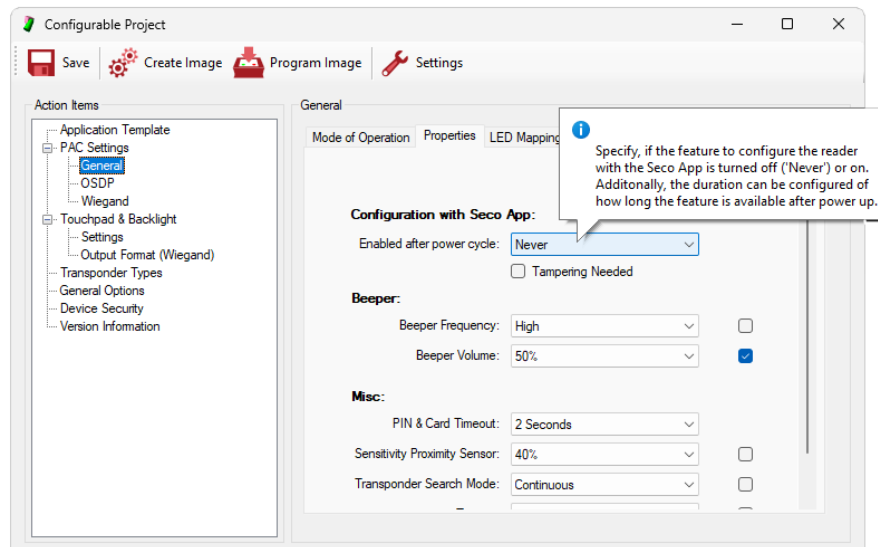
### 9.1.1 General - Mode of Operation

Depending on the kind of connection between controller and the device, the very first decision is the mode of operation, which is either OSDP or Wiegand.



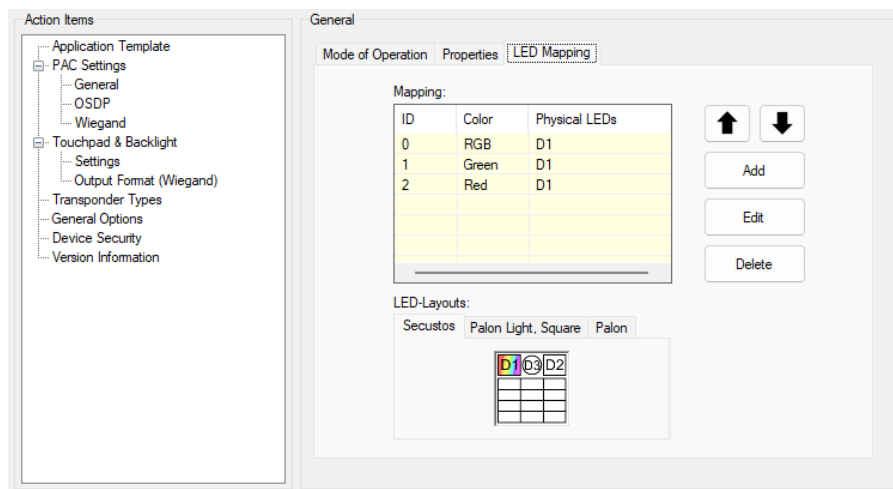
### 9.1.2 General - Properties

Set up further properties of TWN4 according to your requirements. A hovering tool tip is showing details about the specific option.



### 9.1.3 General - LED Mapping

Set up LED mapping of TWN4 according to your requirements:



## 9.2 PAC Settings - OSDP

### 9.2.1 OSDP - Properties

Set up properties of TWN4 according to your requirements:

The screenshot shows the 'OSDP' configuration window with the 'Properties' tab selected. The left sidebar shows a tree view with 'OSDP' highlighted under 'PAC Settings'. The main area contains the following settings:

- RS-485 Interface:**
  - Address: 0 (dropdown) ☒
  - Baud Rate: 9600 (dropdown) ☐
  - RS-485 Termination: On (dropdown) ☒
  - RS-485 Bias: Off (dropdown) ☐
- Misc:**
  - Host Interface: COM1 (dropdown)
- OSDP Protocol Options:**
  - Report Card Data: RAW message (dropdown)
  - Report Keypad Data: KEYPAD message (dropdown)

A 'Configure with Seco App' button is located on the right side of the window.

### 9.2.2 OSDP - Security

Set up security of TWN4 according to your requirements:

The screenshot shows the 'OSDP' configuration window with the 'Security' tab selected. The left sidebar shows a tree view with 'OSDP' highlighted under 'PAC Settings'. The main area contains the following security settings:

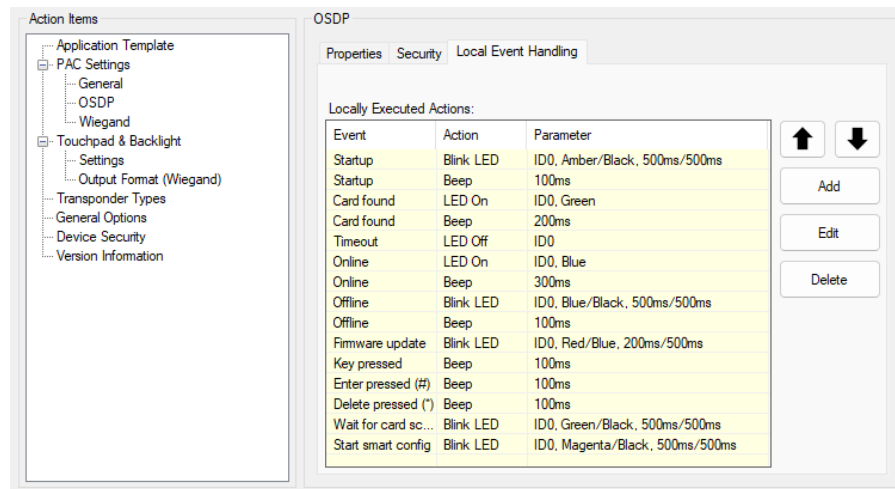
- Security: Security Off (dropdown)
- SCBK\_D: (dropdown) (Hex)
- SCBK: (dropdown) (Hex)
- Masterkey: (dropdown) (Hex)



### 9.2.3 OSDP - Local Event Handling

Set up local event handling of TWN4 according to your requirements:

Note: Events for OSDP are set up independently from the events for Wiegand.

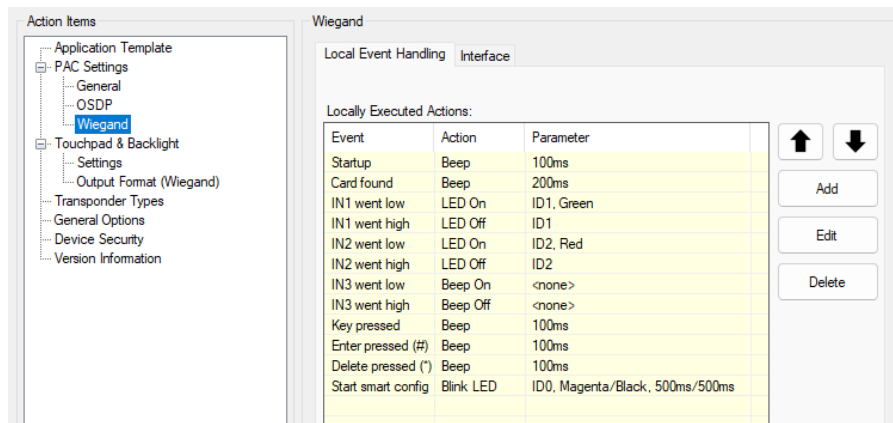


## 9.3 PAC Settings - Wiegand

### 9.3.1 Wiegand - Local Event Handling

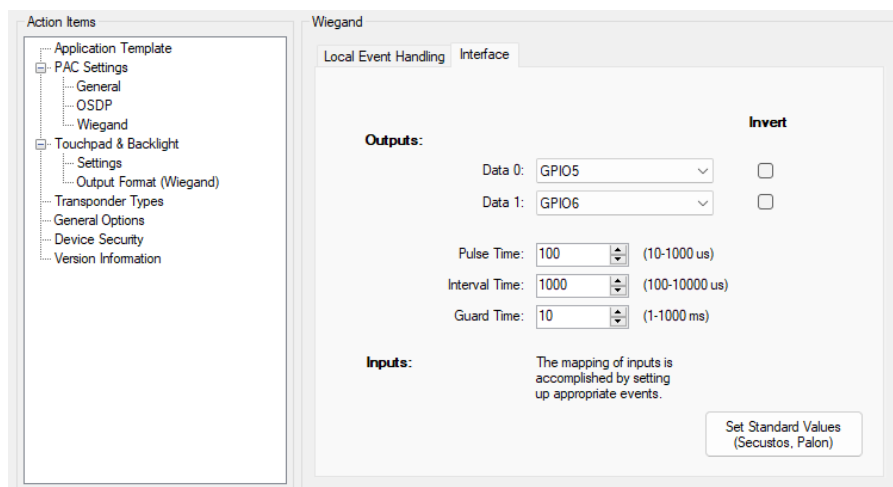
Set up local event handling of TWN4 according to your requirements.

Note: Events for Wiegand are set up independently from the events for OSDP.



### 9.3.2 Wiegand - Interface

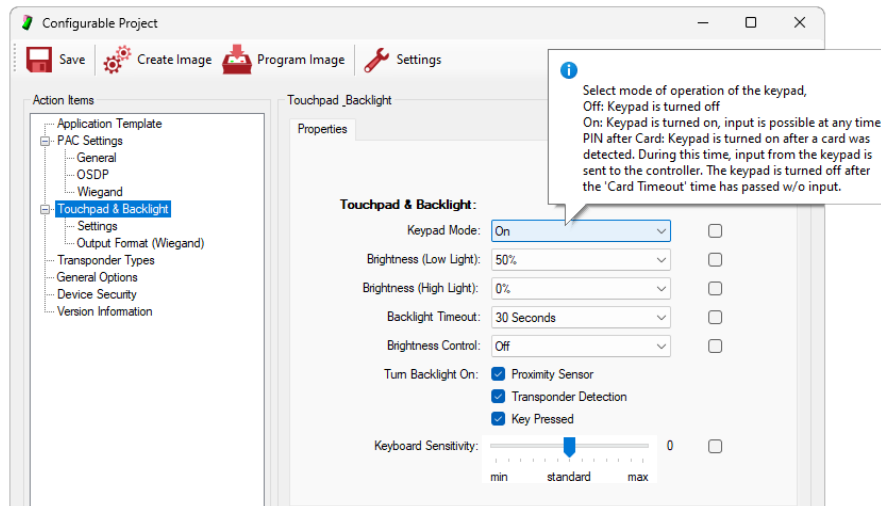
Change these settings only for very special requirements. You can restore standard values for Secustos and Palon at any time by clicking the according button on the lower right of the dialog.



## 9.4 Touchpad & Backlight

### 9.4.1 Settings

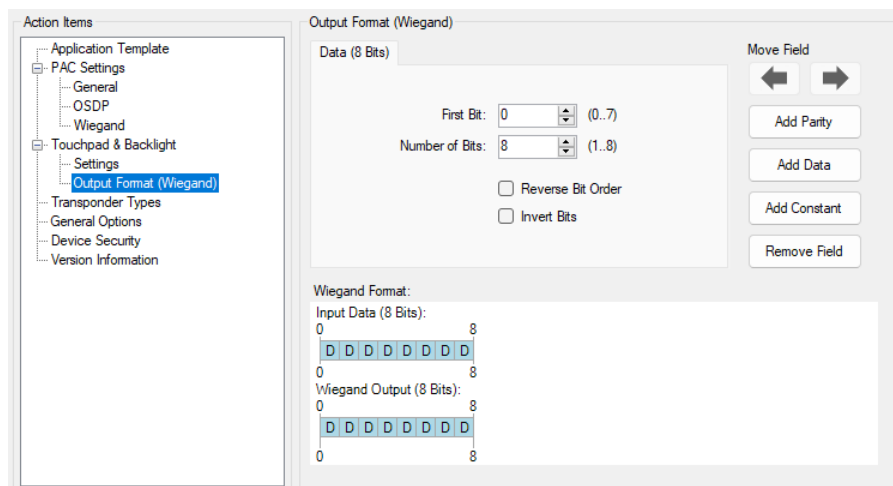
Set up properties of touch and backlight according to your requirements. A hovering tool tip is showing details about the specific option.



### 9.4.2 Output Format (Wiegand)

Set up properties of touch and backlight according to your requirements. A hovering tool tip is showing details about the specific option.

Note: While the output format of OSDP is standardized, the output for Wiegand may vary depending on the type and model of the controller. Please contact the provider of the controller for format related information.



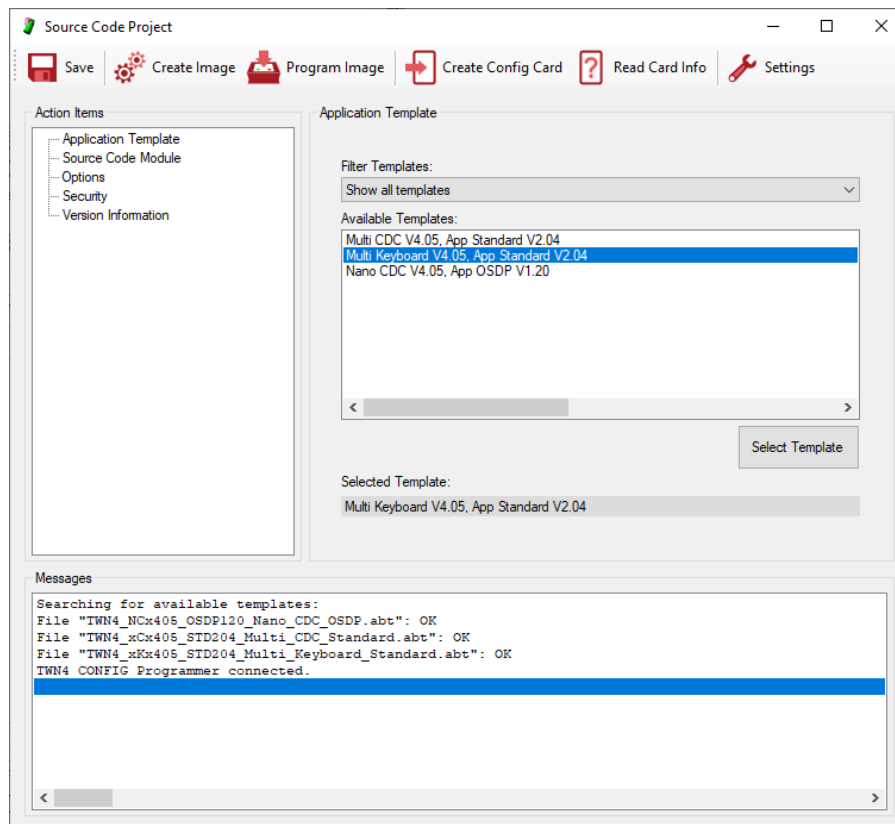
## 10 Source Code Project

For several reasons, there might be a need for creating a configuration, which is not covered by a configurable project:

- There is some source code, you got from 3rd party.
- You have some specific requirements for reading data from a transponder, where data is in some tricky data section of the transponder or some kind of special authentication is required to do reading of data.
- There is a requirement to change entire behaviour of TWN4. This could be e.g. implementing some kind of host based communication, modifying behaviour of beeper, LEDs or doing input/output via other interfaces of TWN4.

The way out of this situation, which is offered by AppBlaster, is to create a project based on custom source code. This is called source code project.

In order to create a source code project, click button "Source Code Project" in the start dialog. The project dialog appears in a somewhat reduced version because - compared to a configurable project - most of setup is not needed here.

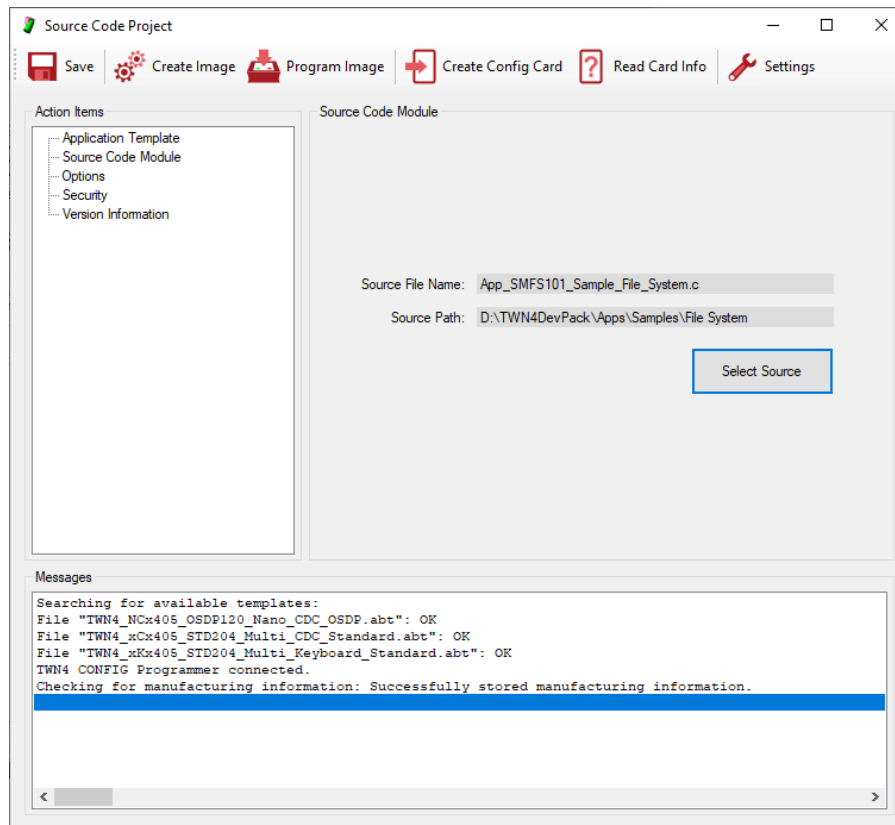


## 10.1 Step 1: Specify Type of USB

TWN4 can be programmed to act as several different type of USB. Select action item "Type of USB (Template)" in order to select appropriate operating mode.

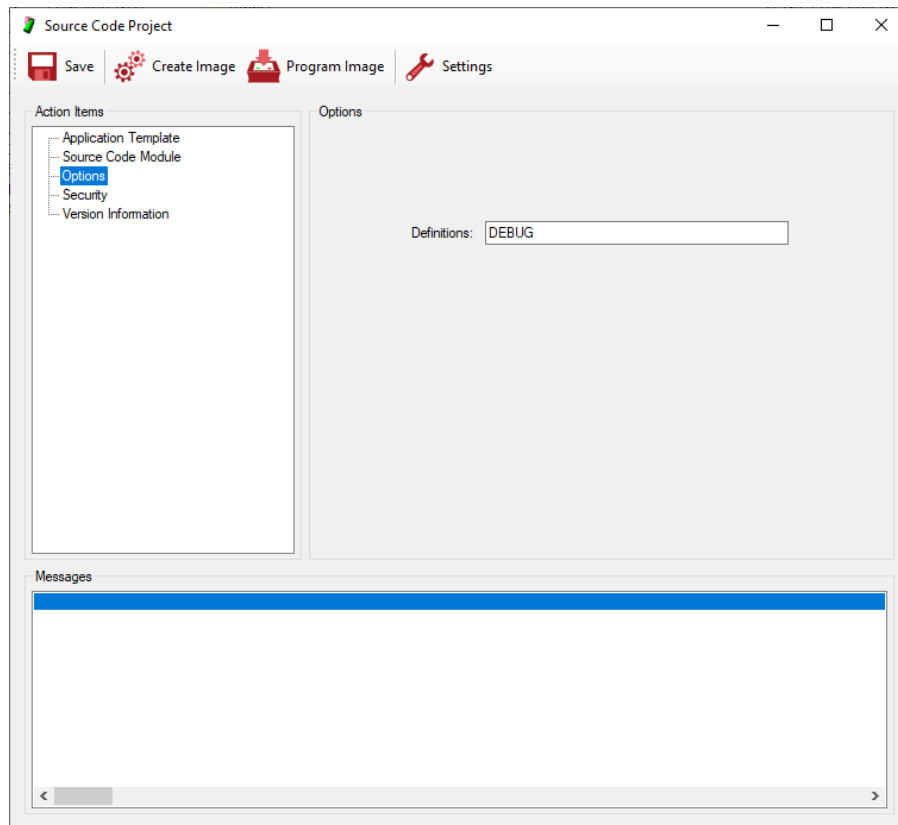
## 10.2 Step 2: Specify Source Code Module

Click button "Select Source" to select the file, which contains the source code in question.



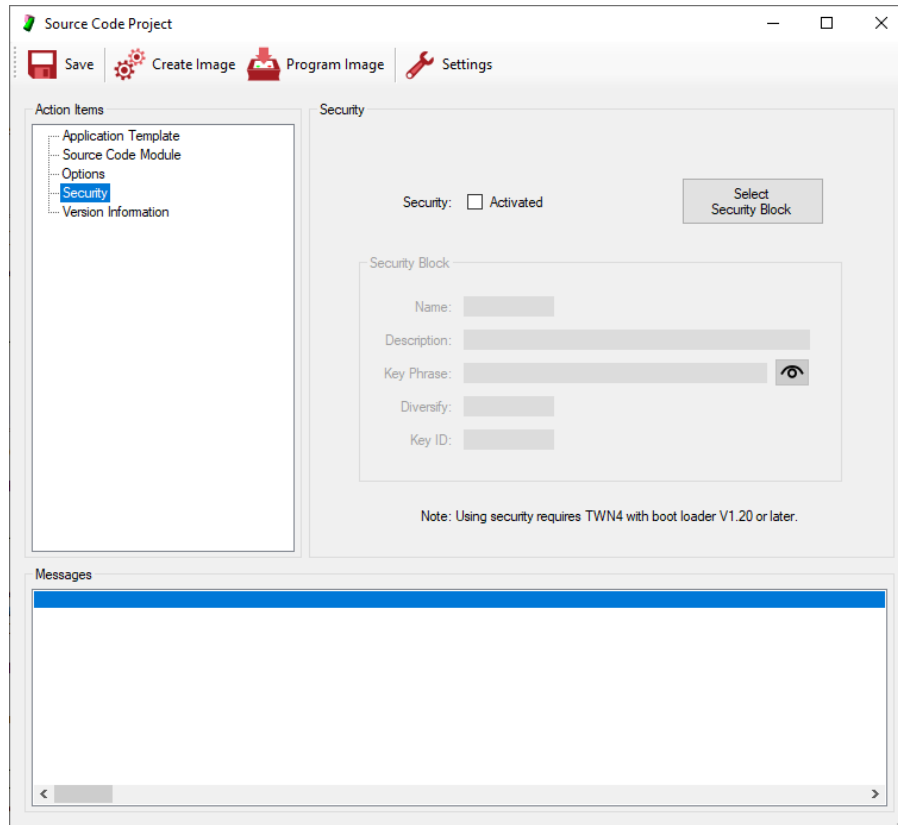
### 10.3 Step 3: Specify Options (Optional)

In this section, you may specify definitions, which are needed to compile the application for a specific purpose. A typical use-case is e.g. the definition of DEBUG, which activates additional diagnostic output by the App. The specific behavior itself must be implemented by the programmer.



## 10.4 Step 4: Specify Security Information

With AppBlaster V4, the concept of TWN4 Device Security is introduced. There is a separate document "TWN4 Device Security", which explains the subject in detail.



## 10.5 Step 5: Specify Version Information (Recommended)

It is recommended to use a standard naming scheme for the firmware image. The information contained in the file name of the firmware image will be used at several places and allows recognition of correct version of a firmware image or TWN4.

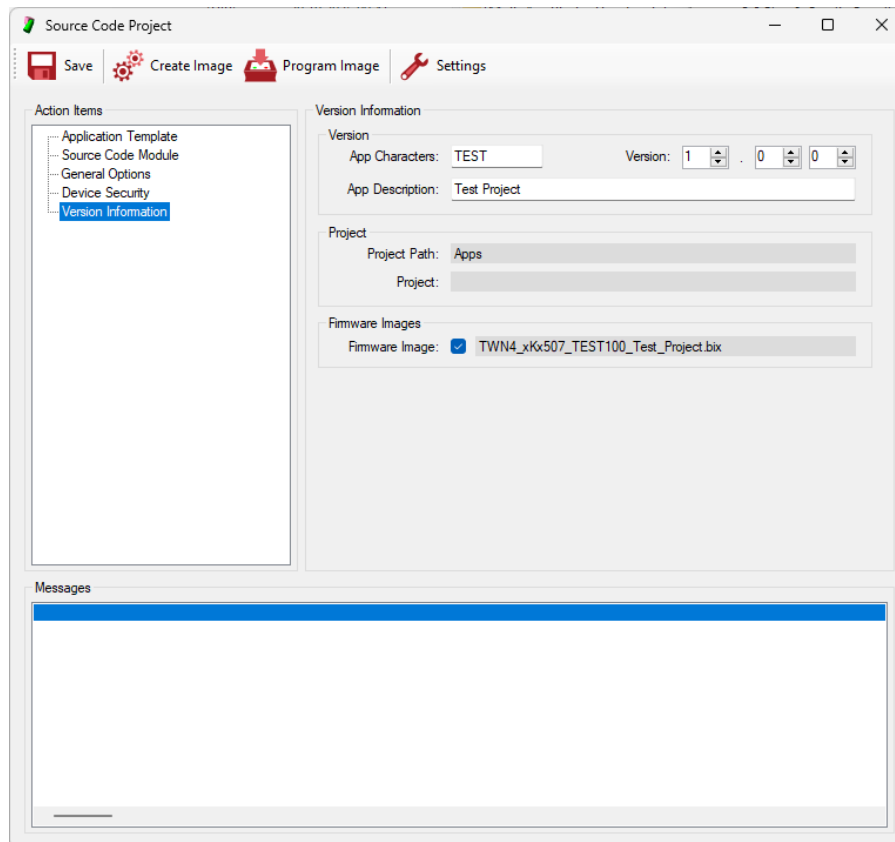
Version information of TWN4 is therefore identical under following conditions:

- The name of the firmware image file itself.
- Information contained on the label of TWN4.
- Version information returned by the application running on TWN4. This is applicable for firmware, which allows request for the firmware version and is achieved via the system function `GetString`.
- Version information returned by USB as product string

AppBlaster supports the naming scheme by offering a dialog, where appropriate information can be entered:

- App Characters: These characters are 1 to 4 letters ('A'-'Z') or digits ('0'-'9'). Please use at least 2 characters which identify the project.
- Version: The version always consists out of 3 digits, which can be set up here.
- App Description: You may do a short description of the project here.

Note: In front of every image which can be created there is a check box, which allows you to turn generation of that image on and off. For source code projects there is only the option to select a single image.





## 10.6 Step 6: Create Image

Before a configuration can be tested, the firmware image must be created.

In order to create the firmware image, click button "Create Image". If the project never was saved before, the project will be created in the sub directory "Apps". Otherwise, it will be created, where the project file resides.

## 10.7 Step 7: Program Image

If the firmware image was created successfully it can be programmed into TWN4.

In order to program firmware image, click button "Program Image".

## 10.8 Step 8: Save Project (Recommended)

After finishing work, you may save entire information in a project file. Compared to the production image, the project file allows to modify settings of the configuration later, the production image does not.

As a suggestion, AppBlaster will choose the name for the project file identical to the name of the production image.

## 11 Creation of Apps with *make*

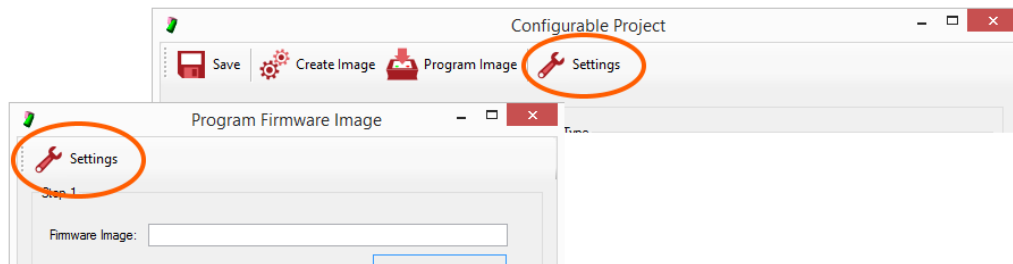
Within AppBlaster there are limitations, which might not be acceptable by advanced users, such as:

- Only one file which contains the source code for the entire App. Especially bigger projects do have a demand for splitting source code into several source and header files.
- Requirement to compile entire source code even in case of small changes.
- AppBlaster is a solution, which is away from well-known programming environments such as Eclipse.

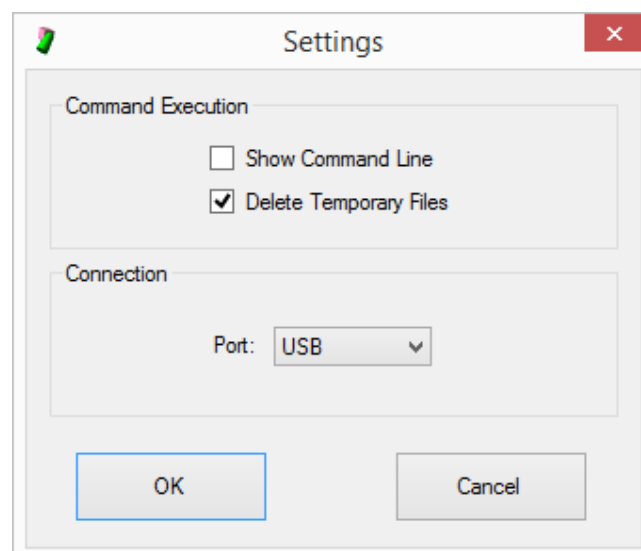
The tools for creation of an App from a command line are part of the TWN4 developer pack. AppBlaster is a graphical user interface (GUI) for the tool chain which resides in the subdirectory `Tools`. In order to become more convenient with how an App is built, there is an option to display the command line parameters during creation of an App. This option can be found in "Settings" of AppBlaster.

## 12 Setting Up AppBlaster

There are a few settings, which can be made for AppBlaster. The settings can be accessed from both a project dialog or dialog for programming firmware:



- "Command Execution - Show Command Line": By activating this option, the commands, which are used for creation of images are displayed in the protocol section. This is helpful, if you plan to set up a project using make.
- "Command Execution - Delete Temporary Files": Turn this option off, if you are interested in temporary files, which are part of the compilation process.
- "Connection - Port": Use USB as long as TWN4 is connected via USB to the host (even virtual COM port). If you are using TWN4 RS232, please select appropriate COM port here.



## 13 Disclaimer

ELATEC GmbH reserves the right to change any information or data in this document without prior notice. The distribution and the update of this document is not controlled. ELATEC GmbH declines all responsibility for the use of product with any other specifications but the ones mentioned above. Any additional requirement for a specific custom application has to be validated by the customer himself at his own responsibility. Where application information is given, it is only advisory and does not form part of the specification.

All referenced brands, product names, service names and trademarks mentioned in this document are the property of their respective owners.