ToolBoxV2 – User Manual

Introduction

ToolBoxV2 is an engineering utility designed to simplify and accelerate routine development tasks.
The application provides a set of integrated tools that help you:

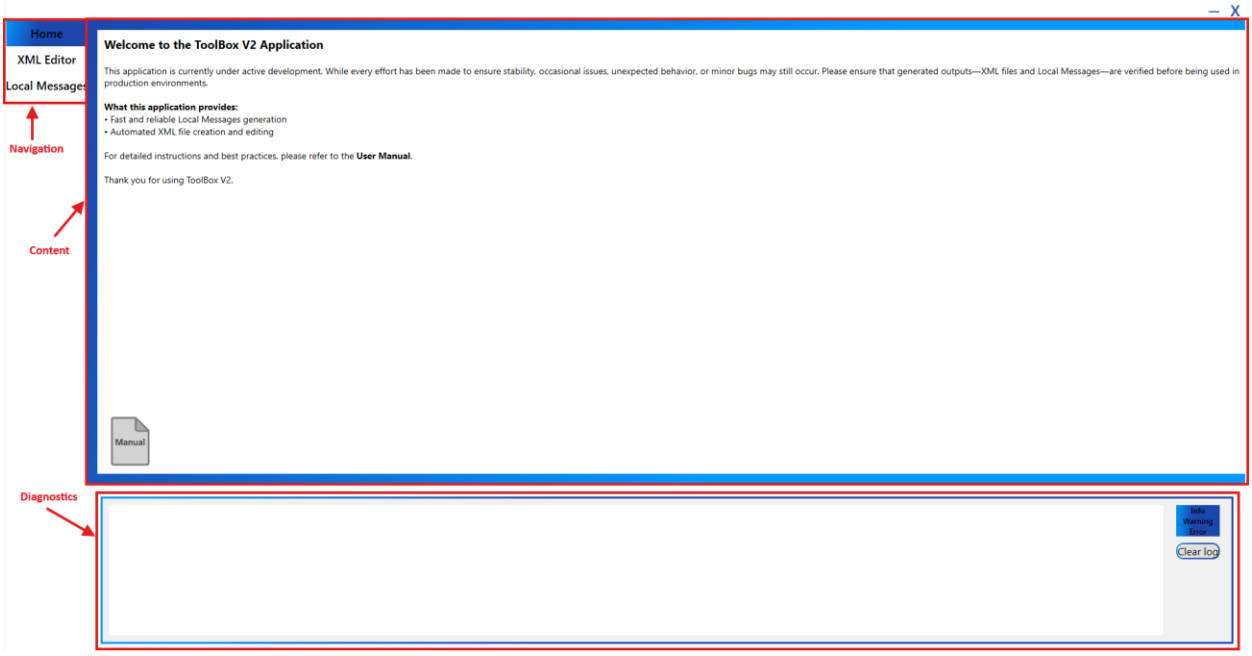
- Generate **Local Messages** from Excel files
- Create, edit, and update **XML files** using configurable templates
- Streamline repetitive engineering steps that are prone to manual errors

The application helps maintain consistency across projects by providing structured tools that automate common steps. Each feature is designed to guide the user through the process, offering clear diagnostic messages and feedback when something needs attention.

ToolBoxV2 is still under development, and some features may evolve over time. Occasional issues or unexpected behaviors may occur, especially when handling complex files or custom templates. It is recommended to always verify generated outputs—particularly XML files and Local Messages—before using them in any final or production environment.

Thank you for choosing ToolBoxV2 to enhance your engineering workflow.

Overview



ToolBoxV2 is organized into three main areas that define the user experience and workflow:

Left Navigation Panel

The left-side panel provides quick access to all available tools within the application. Each button or menu item represents a specific module—such as Local Messages or XML Editor.

This panel remains visible at all times, allowing the user to switch between features without losing context.

Center Content Area

The central area displays the currently selected tool or screen.
This is where the main work takes place, including:

- Loading and editing data
- Viewing previews
- Configuring settings
- Running generation or update operations

Each tool is designed with its own interface but follows a consistent layout to keep navigation intuitive.

Diagnostics Panel (Bottom Section)

The lower section of the application displays diagnostic messages.
This panel shows:

- Informational messages
- Warnings
- Errors
- Status updates from running operations

The diagnostics panel helps track what the application is doing internally and provides clear feedback when something requires user attention. It also supports filtering and clearing messages for better readability.

XML Editor

Introduction

The XML Editor in ToolBoxV2 is designed to simplify work with structured XML files by providing two complementary modes of operation: **Generate** and **Update**.
This tool helps automate repetitive XML creation tasks and ensures consistency when modifying existing XML structures.

Generate Mode

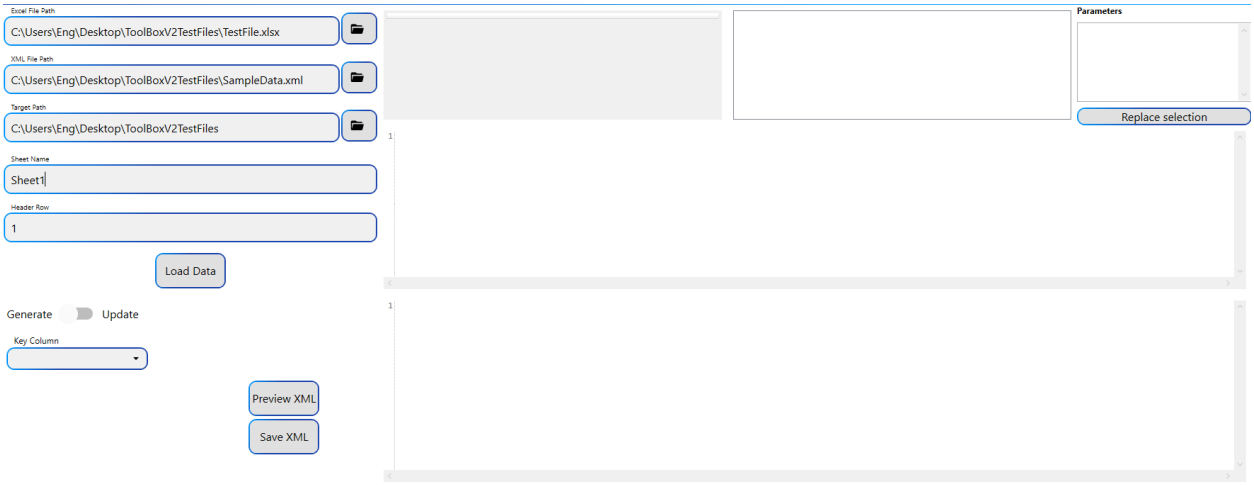
Generate Mode allows the user to create new XML content from a predefined template. The template can contain placeholders or parameters that are dynamically replaced using values from an Excel file.
Once configured, the tool produces fully formatted XML output based on the template and selected data, greatly reducing manual typing and eliminating common mistakes.

Update Mode

Update Mode is intended for modifying existing XML structures.
The user defines a *key attribute*—such as name, id, or any unique identifier.
The editor then locates elements with matching keys and updates their values using data from the Excel file.
This is especially useful when maintaining large XML configurations where only certain items should be modified while preserving the existing structure.

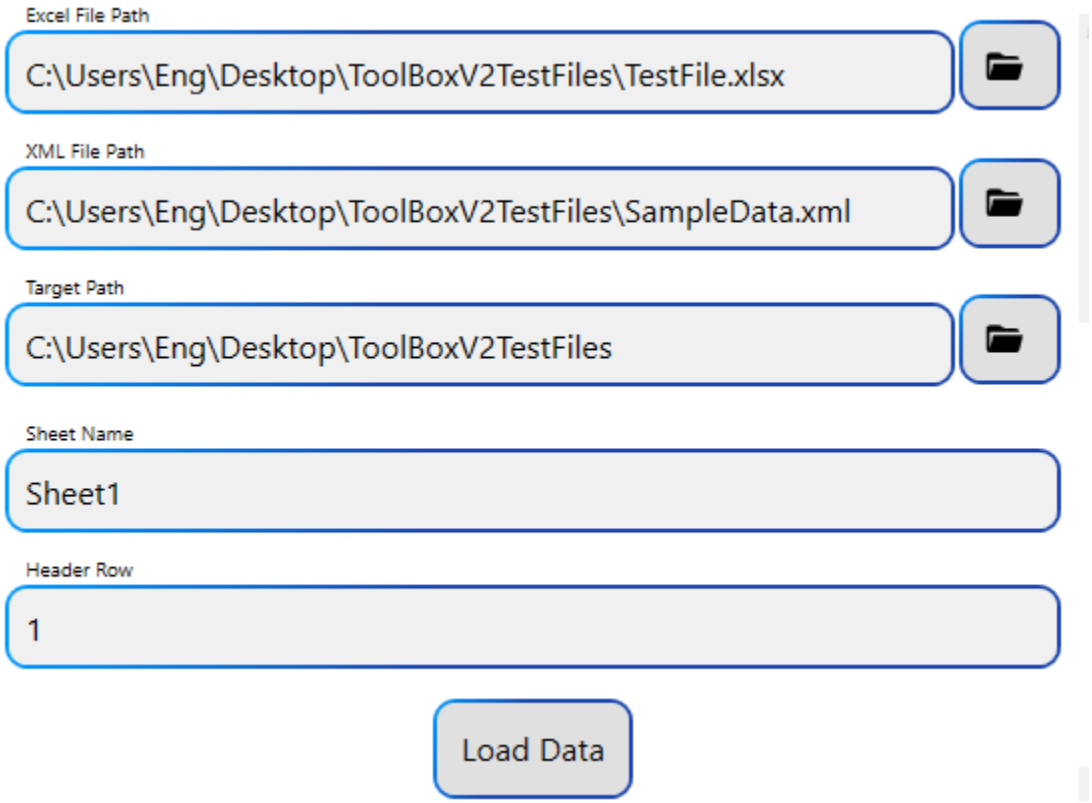
The XML Editor provides a preview of the generated or updated content, allows template adjustments, and offers clear diagnostics to ensure that each operation is completed successfully.
Whether creating new XML files or updating existing ones, this tool is designed to make the process faster, more reliable, and more consistent across projects.

Overview



The XML Editor screen is divided into several functional areas, each designed to support a specific part of the generation or update workflow. Understanding these sections helps navigate the tool efficiently and ensures a smooth editing process.

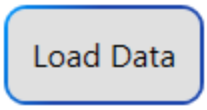
Source



In the upper-left section of the screen, you will find several text boxes, each labeled with its purpose: **Excel File Path**, **XML File Path**, **Target Path**, **Sheet Name**, and **Header Row**. The **Sheet Name** and **Header Row** fields determine which Excel sheet should be loaded and where the column headers are located. A detailed explanation of the required Excel structure can be found later in this manual.

To select files or choose a target folder, use the buttons with the folder icon next to each path field.

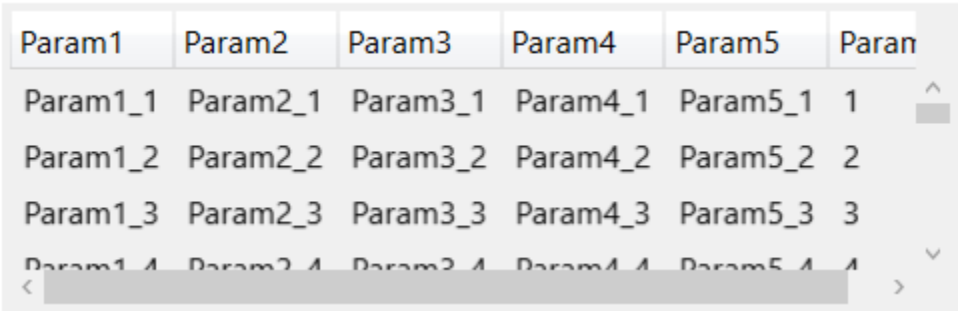
Loading the data is done by pressing the **Load Data** button. This action reads the selected Excel file, loads the XML file, and prepares all necessary information for further processing in the editor.



If you want to generate new XML instances from the template, leave the switch set to **Generate**.
If you need to modify an existing XML file based on a key attribute, switch to **Update**.

Data

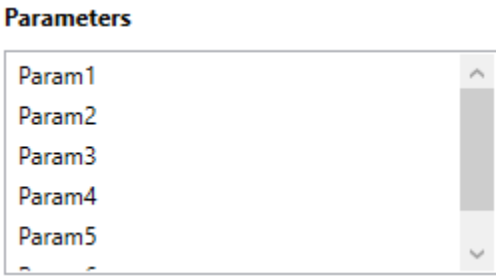
The data loaded from the Excel file is displayed in the Data Grid. Each column represents a header from the specified header row, and each row represents a single instance that can be used for generation or updating.



Param1	Param2	Param3	Param4	Param5	Paran
Param1_1	Param2_1	Param3_1	Param4_1	Param5_1	1
Param1_2	Param2_2	Param3_2	Param4_2	Param5_2	2
Param1_3	Param2_3	Param3_3	Param4_3	Param5_3	3

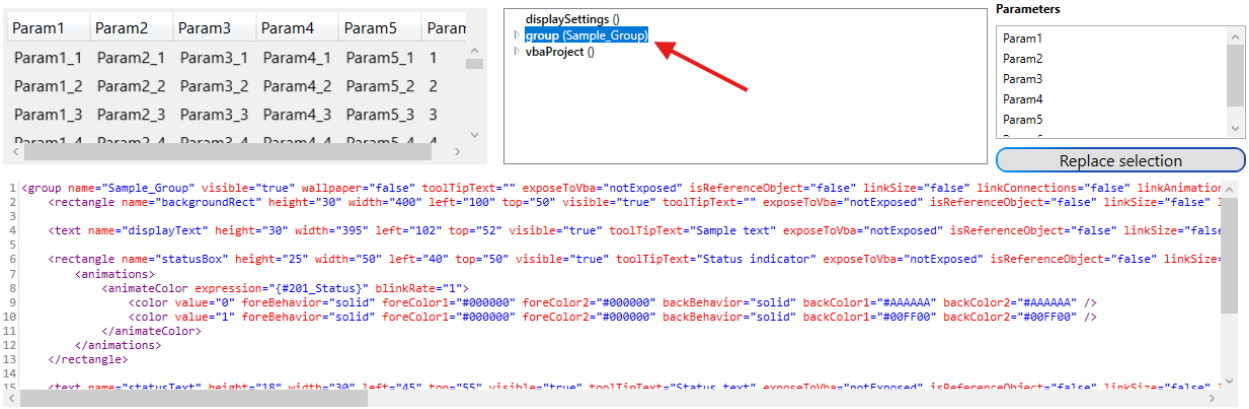
The XML structure is displayed in the Tree View, where you can navigate through individual nodes. This view allows you to select the node that will be used as the template for generation, or choose the node to which the key attribute should be assigned when working in Update Mode.

All parameters—corresponding to the columns from the Excel file—are listed in the ListBox. These parameters can be inserted into the XML template wherever dynamic values are needed.

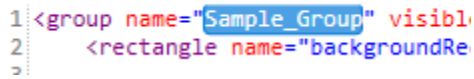


Generating workflow

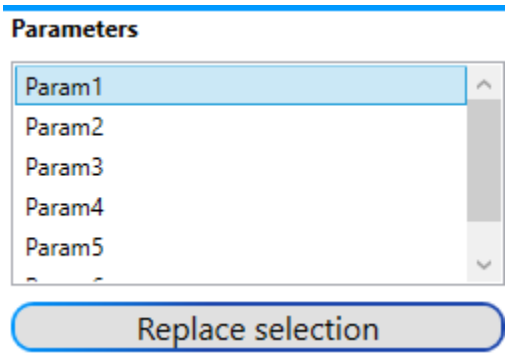
When you select a node in the Tree View, its corresponding XML content is displayed in the XML editor. This allows you to review or modify the node structure before using it as a template or applying updates.



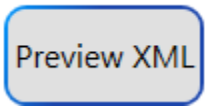
To insert a parameter into the XML template, first select the exact position in the XML editor where the parameter should be placed.



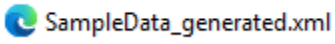
To insert a parameter, select it from the parameter list and then press the **Replace Selection** button. The chosen parameter will be inserted into the XML template at the current cursor position or replace the selected text.



Once all desired parameters have been placed in the template, press **Preview XML**. The application will generate a preview based on the current template and Excel data. The resulting XML output will appear in the preview panel below the editor.

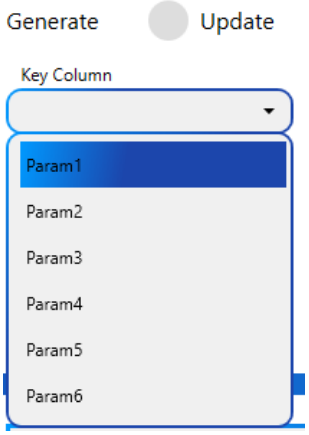


After reviewing the preview and confirming that everything looks correct, press the **Save XML** button. The application will generate the final XML file and place it in the selected target folder, using a filename that includes the suffix **_generated**.

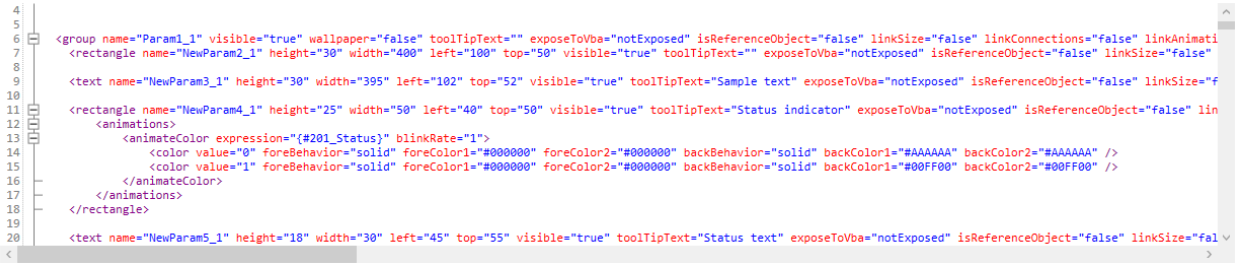


Editing workflow

If you want to edit an existing XML file, switch the mode to **Update** and specify which parameter should be used as the key. The key determines how rows from the Excel file are matched to the corresponding XML nodes during the update process.



Other than selecting a key, the workflow in Update Mode remains the same: assign the key parameter, insert any additional parameters as needed, generate a preview, and finally save the updated XML file.



Local Messages

Introduction

Local Messages are a feature used in FactoryTalk View applications to store operator-facing text messages. These messages are typically displayed on HMI screens to provide status information, alarms, instructions, or diagnostic feedback. Each Local Message contains multiple message items, and each item corresponds to a specific index that triggers the message at runtime.

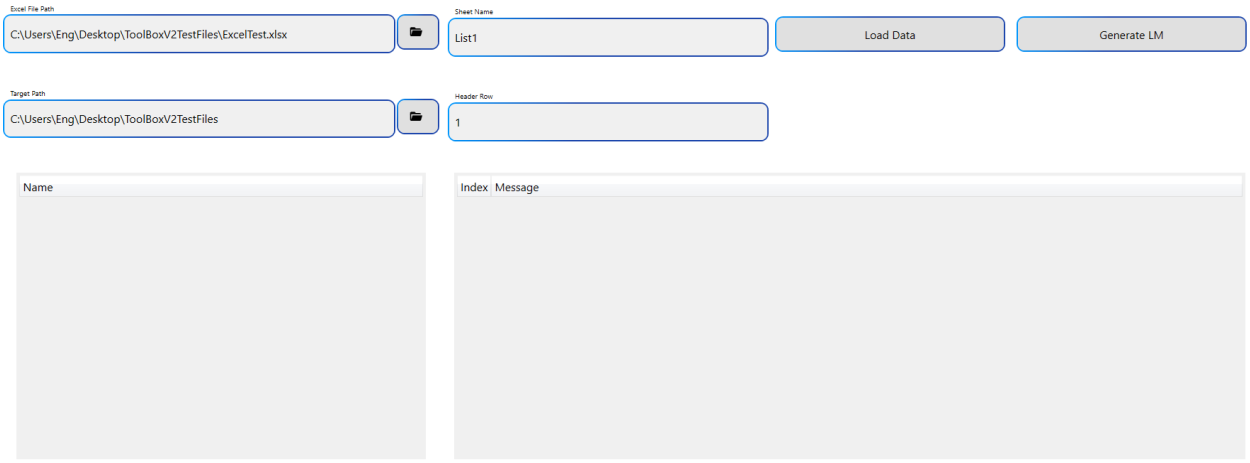
In FactoryTalk View, Local Messages are stored inside a **Compound File**—a structured binary file format similar to a miniature filesystem. Because of this format, Local Messages cannot be edited or created simply by modifying plain text files. They must be handled using the correct internal structure and stream definitions.

ToolBoxV2 simplifies this process by allowing the user to:

- Load an Excel file containing message definitions
- Convert the content into properly formatted Local Message structures
- Generate .loc compound files with the correct streams
- Export the results directly to the selected output path

This eliminates the need to manually manipulate the compound file, reduces the risk of mistakes, and speeds up the workflow significantly—especially when working with large sets of messages or recurring message structures across multiple screens or projects.

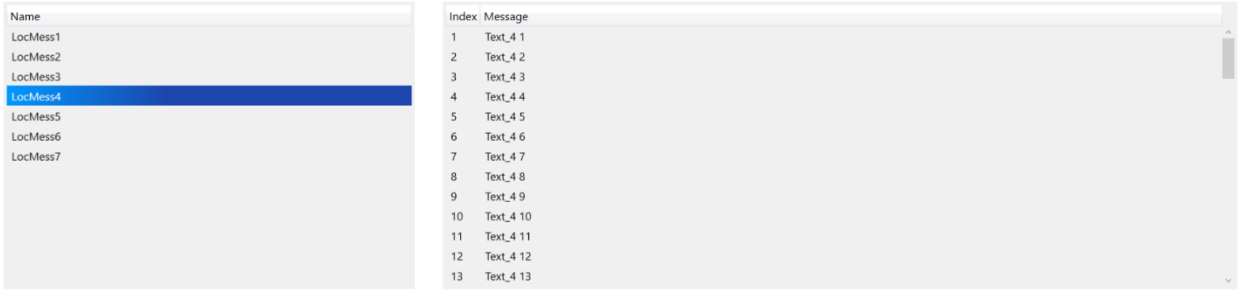
Overview



Just like in the XML Editor, the Local Messages tool includes the same input fields: **Excel File Path**, **Target Path**, **Sheet Name**, and **Header Row**. After pressing **Load Data**, the application reads the Excel file and displays the results in two DataGrids:

- **Left DataGrid** – Shows the list of Local Message names.
- **Right DataGrid** – Shows the items belonging to the selected Local Message, including the index and the message text.

When you select a Local Message from the left side, the corresponding items are automatically displayed on the right.



When you press the **Generate LM** button, the application generates the corresponding Local Message files and saves them in the selected target folder.

Excel Data Structure

XML Editor

For the XML Editor, the application expects the Excel file to contain a set of parameters represented as columns. Each column name becomes a parameter that can be inserted into the XML template, and each row represents a single instance used during generation or updating.

All columns you define will be treated as parameters and made available in the parameter list. Naturally, this works best with a reasonable number of columns—adding hundreds of thousands (or an entire data warehouse) may not only slow down the application but could also make your laptop question its life choices.

	A	B	C	D	E	F
1	**Param1**	**Param2**	**Param3**	**Param4**	**Param5**	**Param6**
2	Param1_1	Param2_1	Param3_1	Param4_1	Param5_1	1
3	Param1_2	Param2_2	Param3_2	Param4_2	Param5_2	2
4	Param1_3	Param2_3	Param3_3	Param4_3	Param5_3	3
5	Param1_4	Param2_4	Param3_4	Param4_4	Param5_4	4
6	Param1_5	Param2_5	Param3_5	Param4_5	Param5_5	5
7	Param1_6	Param2_6	Param3_6	Param4_6	Param5_6	6
8	Param1_7	Param2_7	Param3_7	Param4_7	Param5_7	7
9	Param1_8	Param2_8	Param3_8	Param4_8	Param5_8	8
10	Param1_9	Param2_9	Param3_9	Param4_9	Param5_9	9
11	Param1_10	Param2_10	Param3_10	Param4_10	Param5_10	10
12	Param1_11	Param2_11	Param3_11	Param4_11	Param5_11	11
13	Param1_12	Param2_12	Param3_12	Param4_12	Param5_12	12
14	Param1_13	Param2_13	Param3_13	Param4_13	Param5_13	13
15	Param1_14	Param2_14	Param3_14	Param4_14	Param5_14	14
16	Param1_15	Param2_15	Param3_15	Param4_15	Param5_15	15
17	Param1_16	Param2_16	Param3_16	Param4_16	Param5_16	16
18	Param1_17	Param2_17	Param3_17	Param4_17	Param5_17	17
19	Param1_18	Param2_18	Param3_18	Param4_18	Param5_18	18
20	Param1_19	Param2_19	Param3_19	Param4_19	Param5_19	19
21	Param1_20	Param2_20	Param3_20	Param4_20	Param5_20	20

For the editing option, the required Excel structure is the same as described above. The application uses the column names as parameters, and each row represents one instance that will be matched to the XML nodes based on the selected key.

	A	B	C	D	E	F
1	**Param1**	**Param2**	**Param3**	**Param4**	**Param5**	**Param6**
2	Param1_1	NewParam2_1	NewParam3_1	NewParam4_1	NewParam5_1	1
3	Param1_2	NewParam2_2	NewParam3_2	NewParam4_2	NewParam5_2	2
4	Param1_3	NewParam2_3	NewParam3_3	NewParam4_3	NewParam5_3	3
5	Param1_4	NewParam2_4	NewParam3_4	NewParam4_4	NewParam5_4	4
6	Param1_5	NewParam2_5	NewParam3_5	NewParam4_5	NewParam5_5	5
7	Param1_6	NewParam2_6	NewParam3_6	NewParam4_6	NewParam5_6	6
8	Param1_7	NewParam2_7	NewParam3_7	NewParam4_7	NewParam5_7	7
9	Param1_8	NewParam2_8	NewParam3_8	NewParam4_8	NewParam5_8	8
10	Param1_9	NewParam2_9	NewParam3_9	NewParam4_9	NewParam5_9	9
11	Param1_10	NewParam2_10	NewParam3_10	NewParam4_10	NewParam5_10	10
12	Param1_11	NewParam2_11	NewParam3_11	NewParam4_11	NewParam5_11	11
13	Param1_12	NewParam2_12	NewParam3_12	NewParam4_12	NewParam5_12	12
14	Param1_13	NewParam2_13	NewParam3_13	NewParam4_13	NewParam5_13	13
15	Param1_14	NewParam2_14	NewParam3_14	NewParam4_14	NewParam5_14	14
16	Param1_15	NewParam2_15	NewParam3_15	NewParam4_15	NewParam5_15	15
17	Param1_16	NewParam2_16	NewParam3_16	NewParam4_16	NewParam5_16	16
18	Param1_17	NewParam2_17	NewParam3_17	NewParam4_17	NewParam5_17	17
19	Param1_18	NewParam2_18	NewParam3_18	NewParam4_18	NewParam5_18	18
20	Param1_19	NewParam2_19	NewParam3_19	NewParam4_19	NewParam5_19	19
21	Param1_20	NewParam2_20	NewParam3_20	NewParam4_20	NewParam5_20	20

Local Messages

For Local Messages, the application expects the Excel file to contain the following headers: **Name**, **Index**, and **Text**.
Each row should specify the Local Message name, followed by the corresponding index and message text. All rows with the same *Name* will be grouped together and form a single Local Message with its associated items.

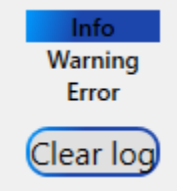
	A	B	C
1	Name	Index	Text
2	LocMess1	1	Text_1 1
3	LocMess1	2	Text_1 2
4	LocMess1	3	Text_1 3
5	LocMess1	4	Text_1 4
6	LocMess1	5	Text_1 5
7	LocMess1	6	Text_1 6

Diagnostics

The diagnostics panel is located at the bottom of the application, as mentioned earlier. It provides real-time feedback about the actions performed by the application. Messages appear as **Info**, **Warning**, or **Error**, each displayed with its own color for easy identification. This helps you understand what is happening internally and quickly spot any issues that may require attention.

The diagnostics panel can be filtered using the checkboxes on the right side. These allow you to show or hide **Info**, **Warning**, and **Error** messages depending on what you want to focus on.

You can also use the **Clear Log** button to remove all displayed messages and start with a clean diagnostic view.

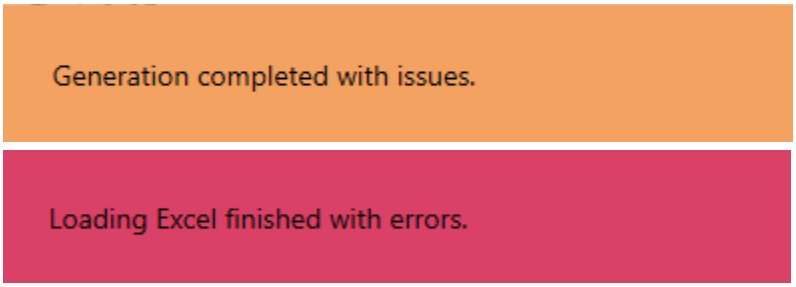


Snackbar Notifications

The application also displays quick notifications through a **Snackbar** popup. Whenever a file is generated or an action is completed, a short message appears at the bottom of the screen. Each Snackbar notification is color-coded according to its type:

- **Green** for successful operations
- **Orange** for warnings
- **Red** for errors

These popups provide immediate feedback and help you quickly understand the result of the action, even if you're working in another part of the interface.



Disclaimer

ToolBoxV2 is provided as a utility to support engineering workflows and automate repetitive tasks. While the application is designed with care and tested on a variety of scenarios, it may still contain limitations or unexpected behaviors. Users are responsible for verifying all generated outputs, including XML files, Local Messages, and any other produced data, before applying them in real projects or production environments.

The author does not assume liability for incorrect configurations, data loss, project issues, or any consequences resulting from the misuse of this application or from unverified outputs. Always maintain backups of your data and review results carefully.

By using ToolBoxV2, you acknowledge that the software is delivered “as is” and may continue to evolve over time. Feedback, bug reports, and suggestions are appreciated and help improve the tool for future versions.