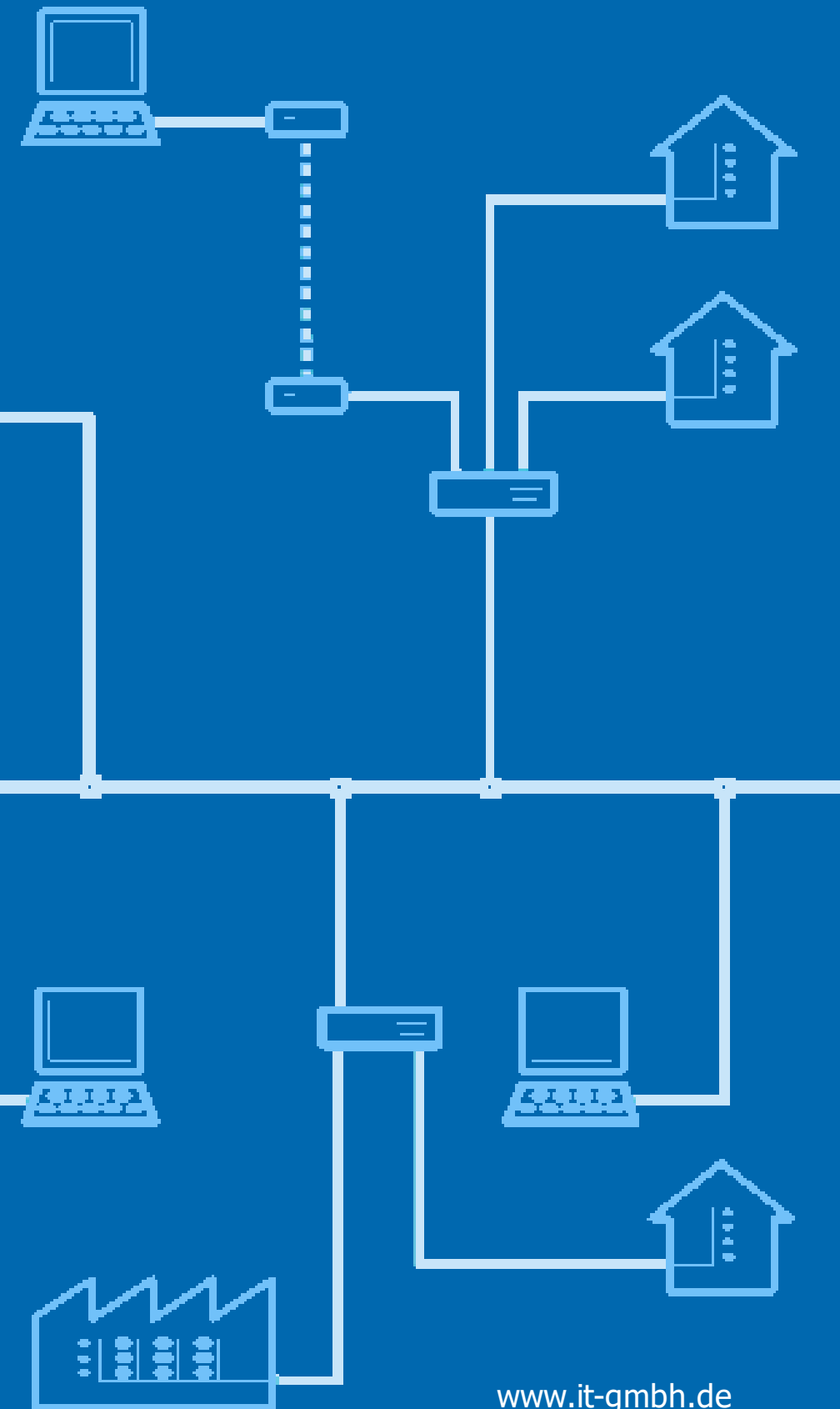




Gesellschaft für
Informationstechnik mbH



Manual Recorder

Recorder

Manual

Table of Contents

Chapter 1	Welcome	4
1.1	Product Overview.....	5
1.2	System Requirements.....	5
1.3	Licensing.....	6
Chapter 2	How to	8
2.1	Record.....	8
2.1.1	How it works	8
2.1.2	Configuration	8
2.1.2.1	Create the Configuration File.....	8
2.1.2.1.1	Interfaces	8
2.1.2.1.2	Output	9
2.1.2.2	Test	12
2.1.2.3	Unattended Operation.....	12
2.1.3	Data Maintenance	12
2.2	Convert and Process.....	14
2.2.1	Convert to different Format	14
2.2.2	Merge Files	15
2.2.3	Filter	15
2.2.4	Options for Text Export	15
2.2.5	Export to Database	16
2.3	Analyze.....	16
2.3.1	Display	17
2.3.2	Plugins	18
2.3.2.1	Plugin Itgmbh.Recorder.Visualizers.....	18
2.3.2.1.1	Value Chart	18
2.3.2.1.2	Statistics	18
2.3.2.2	Plugin Itgmbh.Recorder.ConsistencyChecker.....	18
Chapter 3	Reference	21
3.1	Record.....	21
3.1.1	Configuration File	21
3.1.2	Install as Service (Windows)	23
3.1.3	Install as Daemon (Linux)	24
3.2	Event Properties.....	24
3.3	Project data (Meta data).....	27
3.4	Filter Syntax	27
3.5	Powershell.....	29
3.5.1	Cmdlet Get-RecorderFrame	29
3.5.2	Cmdlet Export-RecorderFile	30
3.5.3	Cmdlet Get-RecorderProjectInfo	31
3.5.4	Class RecorderProjectInfo	33
3.6	Advanced Topics.....	33
3.6.1	Create Plugins	33
3.7	FAQ	34

Chapter 4 Imprint	36
Chapter 5 Contact	38
Chapter 6 Feedback	40
Chapter 7 Open Source Licenses	42
Index	43

Chapter



Welcome

1 Welcome

We congratulate to your purchase of the ETS App Recorder, and thank you for choosing one of our products.

[Product Overview](#)

How to ...

- [Record](#)
- [Convert and Process](#)
- [Analyze](#)

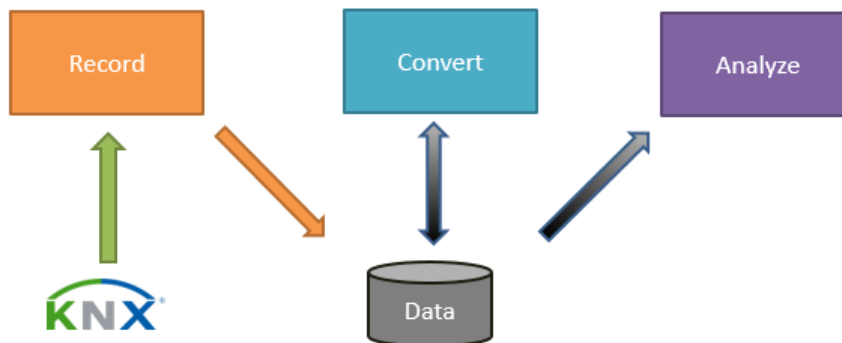
Additional Information

- [Contact \(Orders, Support\)](#)

1.1 Product Overview

Recorder is the Tool Suite around KNX Telegram recording:

- Continuous recording
 - With all supported KNX Interfaces (USB, IP, also with KNX Security)
 - Unattended operation in the background
 - Runs on Windows or Linux
- Conversion and Processing
 - Format conversion, merging, filtering, formatting
 - using a convenient user interface or automated via Powershell scripts
- Analysis
 - Telegram display
 - Powerful filtering
 - Advanced Analysis via Plugins



1.2 System Requirements

Recording function

Operating systems All operating systems supported by .NET Core 2.2, e.g.
 Windows from Windows 7 SP1
 macOS from Version 10.12
 Linux (various distributions)

[Complete List](#)

KNX Interfaces KNX USB Interface
 KNX IP Interface (with or without KNXnet/IP Security)
 KNX IP Backbone (with or without KNXnet/IP Security)

Available Depends on bus load and recording type.

storage space Typical value: with 4 telegrams per second and recording to text file approx. 20 MB per
 for the trace files interface per day

Processing and Analysis function (graphical UI)

Operating systems Windows from Windows 7 SP1

Additional Software Microsoft .NET Framework from Version 4.7.2

Processing and Analysis function (Powershell)

Operating systems All operating systems supported by Windows Powershell
[Complete List](#)

Additional Software Microsoft .NET Framework from Version 4.7.2
 Microsoft PowerShell from Version 5

1.3 Licensing

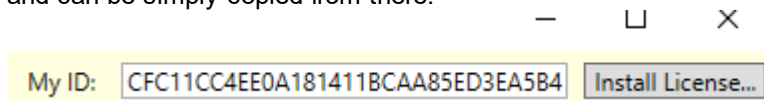
Without License, Recorder runs in Demo mode:

- Recording terminates after 30 minutes
- No installation as Service (Windows) / Daemon (Linux)
- Conversion, Processing and Analysis are restricted to a maximum of 10.000 telegrams.
- The Powershell Cmdlet processes only the first 10.000 telegrams

If you want to run Recorder without these restrictions, you need a license file. The license is bound to the computer, i.e. can only run on the computer for which it has been issued.

To license, proceed as follows:

1. Determine the Computer ID
 - a. This is most easily done by starting RecorderUI. The computer ID is displayed on the upper right and can be simply copied from there:



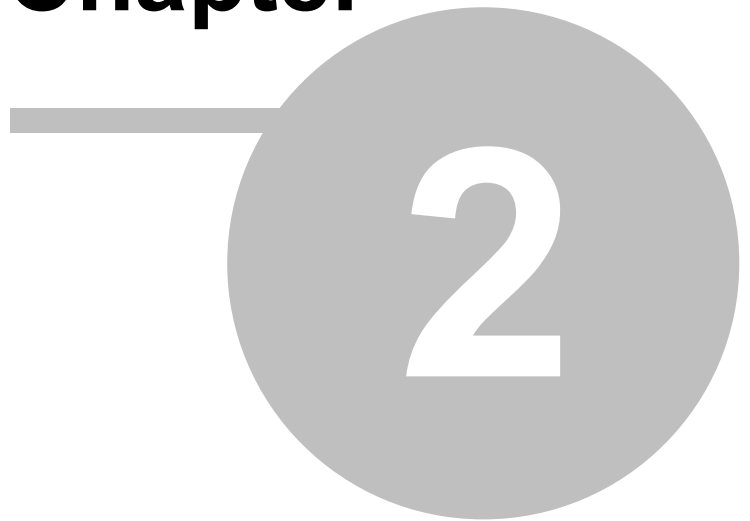
- b. On Linux or if RecorderUI is not installed, you may start the console application Recorder:

```
MachineID: 2CD896BAB33ABC49F96D1D1C61B6BD8D
Running in Demo Mode; terminating after 30 minutes
```

Important when using the recorder in the network: please generate the Machine-ID if there are no "temporary network adapters", e.g. no VPN, but only the ones that are always there!

2. In your order, please indicate:
 - a. the Computer ID
 - b. which license(s) you need:
 - only for recording (Recorder Run)
 - only for processing (Recorder)
 - for both
 If you want Recorder Run and Recorder to run on different computers, please tell us on which computer you want to license which tool.
3. You will get a file that you can activate using the "Install License" button. On Linux or if RecorderUI is not installed, you may also simply copy the file into the folder where Recorder is installed.

Chapter



How to

2 How to

2.1 Record

Recording is performed by a separate module, focused on the task to record and store KNX telegrams unattended, reliably and continuously.

[How it works](#)

[Configuration](#)

[Data Maintenance](#)

2.1.1 How it works

On Start it is attempted to connect to all configured interfaces. If this does not work for an interface, the connection attempt is retried regularly. This also happens if the connection is recognized as broken later. The following events are written to the configured outputs (files):

- Meta events:
 - Recording started / stopped
 - Connection to interface broken / re-established
 - Interface reported telegram loss
 - Telegrams lost when saving
- Telegram events: a KNX telegram has been received
- Busload events: optionally, the current busload is recorded

Usually the output is configured to write one file per day.

2.1.2 Configuration

The recording is configured via a configuration file.

This contains information about the interface(s) to be used and where to write the recorded data.

2.1.2.1 Create the Configuration File

Most easily you create the configuration file using the application RecorderUI.

For this, navigate to the **Record** page.

You may here create a new configuration file or open an existing one for editing.

In the left part, define the [Interfaces](#) used for recording.

In the right part, specify where the [Output](#) shall be written.

The Metadata in the lower part is not needed in the standard case (record as raw telegram data).



Then you may save the file.

Alternatively, the configuration file can be created and edited using any text editor. The format is documented [here](#).

2.1.2.1.1 Interfaces

Adding interfaces

Click on , to add all USB and Tunneling Interfaces accessible from the PC to the list.

Using  (USB),  (Tunneling) and  (Routing) you may also add interfaces manually.

Deleting interfaces

Select the interfaces to be deleted (multi-selection using Shift key) and press the Del key.

Properties

If an interface is selected, you may edit its properties on the right side:

General

Id	This serves to uniquely identify an interface if the recording is done using multiple interfaces. The Id may be an arbitrary string, but please avoid the special characters semicolon (;), asterisk (*) and question mark (?).
Enabled	You may disable individual interfaces without losing their configuration

Interface

Connection String	This string contains all relevant information to connect to the interface. See below.
Interface Mode	The recording can be done in Group monitor or Bus monitor mode. In Bus monitor mode, in addition to group telegrams also individually addressed and faulty telegrams are recorded, this mode is however not supported by all interfaces. "Prefer Bus Monitor" tried to open in Bus monitor mode and falls back to Group monitor mode if this fails.
Report Busload	If not only telegrams shall be recorded but in addition the bus load shall be determined, enter here how often the current bus load (as mean value over this period as well as as maximum second mean value within the period) shall be written to the output.

Connection string for USB

No additional USB settings are available using the user interface. These would only be necessary if multiple USB interfaces are connected. If this is the case, the configuration file needs to be adapted manually; please ask out Support for help.

Connection string for Tunneling

IP Address	The IP address of the tunneling interface
Use NAT	Uses "NAT mode" for the connection. This is necessary if PC and interface communicate via a router with "Network Address Translation", but works also in other cases (except with very old interfaces). Please keep in mind that for security reasons you should never expose a KNX IP Interface via Port Forwarding to the public internet!


Connection string for Routing


Multicast Address The used IP Multicast Address; usually 224.0.23.12

2.1.2.1.2 Output

Adding outputs

Click on  to configure the output into a file.

Click on  to configure sending to a [syslog](#) server.

For testing it is useful, to also configure output to the console window using ; this option has no effect if the recording is running in the background (as Windows service / Linux Daemon).

Deleting outputs

Select the outputs to be deleted (multi-selection using Shift key) and press the Del key.

Properties

If an output is selected, you may edit its properties on the right side:

General

Interfaces	Here you can specify if this output shall receive the data from all interfaces (*) or only specific interfaces. To output only specific interfaces, list their Ids separated with semicolon (;). You may also use the wildcards ? (any character) and * (any number of any characters). Examples: * All interfaces KNX* All interfaces starting with "KNX" KNX5;KNX7 The two interfaces "KNX5" and "KNX7"
Enabled	You may disable individual outputs without losing their configuration

Properties for Console output

No additional properties.

Properties for File output

Output

Base Directory	Enter the path to a directory where the output files shall be written. If empty, the directory "Documents\Recorder" (Windows) / \$HOME/Recorder (Linux) are used. If the recording is run in the background (as Windows service / Linux Daemon) it is advisable to enter a specific path here as otherwise the output will be written to the user profile of the service account. If the entered value contains a part "%name%", this will be replaced by the value of the environment variable <i>name</i> .
File Pattern	Specifies how the file name is constructed. Commonly one would derive this from e.g. the date. The pattern can contain variable parts as { <i>name</i> } or { <i>name:format</i> }. <i>name</i> determines which information from an event to be written shall be evaluated: L Locale Time T Time as UTC (world time) I Interface Id <i>format</i> (only for L and T) determines how the data/tie information shall be formatted (Standard formats and Custom formats). Examples: Trace-{L:yyyy-MM-dd}.txt Start a new file each day (according to local time). The file name for July 1 st 2019 is "Trace-2019-07-01.txt". {L:yyyy-MM} Puts the file into sub-folders according to month. The file name \Trace-{L:dd}.txt for July 1 st 2019 is "2019-07\Trace-01.txt"

File Format	File format
	Text One line per event, fields are separated by TAB (recommended format)
	Binary Very compact binary format (recommended if space is limited)
	XML Identical to the ETS recording format (biggest files)
	Automatic Determines the format from the file extension
Flush Period	Specifies how often data is written to the file. Hint: when writing to an SD card, do not choose the interval too short as this medium only supports a limited amount of write accessed. A sensible value is e.g. one hour (01:00:00).
Removable Drive	Set this checkbox if the target is a removable drives (e.g. USB stick, network).
Tasks	
Delete after	If you write a value > 0 here, files older than this number of days will be deleted. See also Data Maintenance .
Synchronize to	You may enter a location here, where the files will be copied in addition. Details see Data Maintenance . Examples: \\Server\RecorderArchiv (Windows only) ftp://user:password@server/archiv
Format (text files only). Hint: a tab character is represented as \$t	
Prolog	First line of every file; leave empty if not desired
Frame	Specifies how a received KNX frame is formatted
Meta	Specifies how a meta event (e.g. Start/Stop/Connection loss) is formatted; leave empty if not desired
Busload	Specifies how busload data is formatted (has to be enabled in the Interfaces); leave empty if not desired

Properties for Syslog output

Output

Transport	Local Only valid when running on Unix-style operating system: send to the local syslog
	UDP Send via USP
	TCP plain Send via TCP without encryption
	TCP Send via TCP using TLS 1.2 encrypted
Server	The server name or IP address
Port	The IP port (default is 514)
Format	RFC31 Only valid with Transport=Local 64 RFC31 The older BSD syslog format 64 RFC54 The current official syslog format 24
App Name	By default, the AppName field in the syslog messages is "Recorder"; here, you may specify a custom AppName.
Format	
Frame	Specifies how a received KNX frame is formatted

Meta	Specifies how a meta event (e.g. Start/Stop/Connection loss) is formatted; leave empty if not desired
Busload	Specifies how busload data is formatted (has to be enabled in the Interfaces); leave empty if not desired

2.1.2.2 Test

To use the created configuration immediately to start recording (e.g. to test the configuration), click on **Start**.

A console window opens.

Check if errors (in red) or warnings (in yellow) are displayed.

If you configured a Console [Output](#), you should also see the recorded telegrams.


You may also launch the recording module from Windows Explorer by dragging the configuration file onto the Recorder.exe executable.

If IP interfaces are used, you might need to add Recorder.exe to your **Firewall** exception rules.

2.1.2.3 Unattended Operation

For unattended permanent operation, the Recorder should be installed as Service (Windows) / Daemon (Linux).

On Windows this is especially easy if also the Recorder user interface is installed:

1. Switch to the **Record** page.
2. Create or open the configuration file
3. Click on  (Install Service).

Alternatively, you can also install manually: [Install as Service \(Windows\)](#) [Install as Daemon \(Linux/macOS\)](#)

2.1.3 Data Maintenance

Here some additional tips for long-term data maintenance

Synchronizing

Often it is desirable to synchronize the recorded files to an external server.

Built-in Synchronization

If the property "Synchronize to" in the configuration of the [Output](#) is set, the files will be copied there automatically. This of course will be done only if the recorder module is running.

On **Windows** you can specify:

- a Path like e.g. E:\Recorderfiles, where E: might be a local drive or network drive.
- a Network path as e.g. \\Server\Recorderfiles
- an FTP or FTPS-URL with or without login data like e.g. ftp://server/RecorderFiles, ftp://user:password@server/RecorderFiles or ftps://user:password@server/RecorderFiles

If a password is entered, be sure to protect the configuration file against unauthorized read access!

All files will be copied having the "Archive" attribute; this will be set by Windows automatically on each write access and re-set by the Synchronization function after copying.

In addition on Windows the following works:

- a EMail address in the form mailto:user@domain.com. The SMTP Server and the Sender address have to be configured in the Recorder.exe.config with a text editor.

On **Linux** you can specify:

- a path like e.g. /data/recorderfiles where the directory might reside on a local drive or be a network location mapped via mount
- an FTP or FTPS-URL with or without login data like e.g. ftp://server/RecorderFiles, ftp://user:password@server/RecorderFiles or ftps://user:password@server/RecorderFiles

Since Linux has no Archive attribute all files not yet present in the target or have an older timestamp in the target will be copied (caution: the later may lead to problems if the FTP server has a different time zone).

If this functionality is not sufficient, the task can also be implemented easily using operating system means, or with one of the any free or commercial synchronization tools. Here some examples:

Robocopy (Windows)

This powerful standard Windows tool is suitable for (almost) all automated copy operations. To copy all TXT output files from C:\RecorderFiles to the share \\Server\Recorderfiles you may use e.g.:

```
robocopy c:\RecorderFiles \\Server\Recorderfiles *.txt /s /m
```

Explanation:

- /s means: copy also sub-directories
- /m means: copy only files with the Archive bit set, then re-set the Archive bit. The effect is that already copied files will be skipped except if they were modified since then.

To execute this command regularly,

- in Windows Task Scheduler, add a "Basic task",
- give it a name, e.g. "Recorder Sync",
- specify that this shall be executed e.g. daily at 02:00,
- select "Run Program", enter robocopy as program and the arguments e.g. "c:\RecorderFiles \\\Server\Recorderfiles *.txt /s /m".

rsync (Linux)

rsync is a similar tool for Linux. The syntax here is e.g.

```
rsync -a -u /var/recorderfiles server:recorderfiles/
```

Explanation:

- -a means: copy sub-directories, attributes, timestamps etc.
- -u means: copy only files that are newer as existing target files


To execute this command regularly, create a corresponding system cron job. This is done by editing the file /etc/crontab with a text editor, e.g.

```
# m h dom mon dow user  command
0 2 * * * username    rsync -a -u /var/recorderfiles server:recorderfiles/
```

Depending on the distribution there might be a convenient UI for this task.

Delete

To delete recorder files after some time, you may use the property "Delete after" in the [Output](#) configuration. This of course will be done only if the recorder module is running.

Alternatively, you may use operating system features similar to the Synchronization case (see e.g. [here](#)) 

2.2 Convert and Process

The files created by the recording module or by other sources (e.g. ETS) can be process in various ways.

For this, navigate to the **Convert** page.

[Convert to different Format](#)

[Merge Files](#)

[Filter](#)

[Options for Text Export](#)

[Export to Database](#)

As alternative to performing these tasks in the user interface RecorderUI, you might consider using the [Powershell Module](#). This is especially suited for regularly performed automated tasks.

2.2.1 Convert to different Format

It is possible to convert recorder files from one file format to another.

The following formats are supported:

Text	One line per telegram, fields separated by TAB
Binary	Very compact
XML	Identical to ETS monitor file format

To convert one or more files:

1. Add the files to the left part of the window:
 - Use the button Add Files to select one or more files
 - Use the button Add Folder to add all recorder files in a folder and its sub-folders
 - You might also use drag & drop from a Windows Explorer window
 - Remove unwanted files using the **Del** key
 2. On the right, select **Convert all files separately** and specify the folder where the converted files should be put in
 3. Select the desired file format
 4. You might in addition specify a [Filter](#) to be applied. For text output, additional [Options](#) are available
 5. Click **Merge | Convert | Filter** to start the conversion
- If you do not want to convert the files individually, but merge them into one big file, have a look at [Merge Files](#).

2.2.2 Merge Files

It is possible to merge multiple recorder files into a single one. The output file might have the same or a different file format than the input files.


To merge files:

1. Add the files to the left part of the window:
 - Use the button Add Files to select one or more files
 - Use the button Add Folder to add all recorder files in a folder and its sub-folders
 - You might also use drag & drop from a Windows Explorer window
 - Remove unwanted files using the **Del** key
2. On the right, select **Merge into one file** and specify the output file name
3. You might in addition specify a [Filter](#) to be applied. For text output, additional [Options](#) are available
4. The option **Remove Duplicates** will remove duplicated telegrams. This happens e.g. if recording different lines of a KNX installation using multiple interfaces and telegrams are passed from one line to the other. Duplicates are recognize by the fact that they are identical except the routing counter and are received within one second.
The option also removes LinkLayer repetitions from Bus monitor traces.
5. Click **Merge | Convert | Filter** to start the merge process

2.2.3 Filter

A very powerful filter function is at your hands to evaluate events/telegrams according to many criteria. Only events fulfilling the filter condition will be written to the output file when converting or merging.

A filter condition is a string with a specific [Syntax](#) which you might enter directly (for experts) or build up with a filter editor). You access the filter editor using the **+** button.

The most recently used filter conditions are available via the  button.


Delete the filter condition using **x**.

2.2.4 Options for Text Export

If the output file has format "Text", additional options are available. These allow to prepare the output for further processing in other tools (like e.g. Excel).

Columns

As column separator, select tabulator, comma or semicolon.

Add new columns via the  button. All [Event Properties](#) are available.

To delete an entry, select it and press the **Del** key.

To re-order, the entries can be moved with the mouse (grab entry on row header).

Note that the recorder can only read in text files containing at least the default columns.

Project data (Meta data)

If you want to output one of the properties Source Name, Destination Name or Value, or if encrypted telegrams (KNX DataSecurity) occur, [Project data \(Meta data\)](#) is required. If available, just add the ETS project (knxproj); but other formats are also supported.

2.2.5 Export to Database

It is possible to export recorder files data into a database for storage or further evaluation.

The following data providers are supported:

SQL Server Compact A desktop SQL database operating on SDF files (no server installation)



MySQL/MariaDB A popular open source database

SQL Server Microsoft database often encountered in enterprise environments

OLE DB Allows to connect almost every database product using different OLE DB drivers

ODBC Allows to connect almost every database product using different ODBC drivers

To export one or more files:

1. Add the files to the left part of the window:
 - Use the button Add Files to select one or more files
 - Use the button Add Folder to add all recorder files in a folder and its sub-folders
 - You might also use drag & drop from a Windows Explorer window
 - Remove unwanted files using the **Del** key
2. On the right, select **Export to database**
3. Specify the data provider and the provider-specific connection string.
The  button shows all available connection string settings of the selected data provider; you can of course also enter the connection string manually according to the provider documentation. For fast access, the most recently used connection strings are available via the  button.
4. Specify the name of the database table; if the table does not yet exist, it will be created.
5. You might in addition specify a [Filter](#) to be applied and the columns to be exported (as in [Options for text export](#))
6. The option **Remove Duplicates** will remove duplicated telegrams. This happens e.g. if recording different lines of a KNX installation using multiple interfaces and telegrams are passed from one line to the other. Duplicates are recognized by the fact that they are identical except the routing counter and are received within one second.
The option also removes LinkLayer repetitions from Bus monitor traces.
7. Click **Merge | Convert | Filter** to start the export

2.3 Analyze

The Recorder user interface can display and analyze the recorded telegrams.

For this, navigate to the **Analysis** page.

[Display Telegrams](#)

[Plugins](#)

As alternative to analyzing in the user interface RecorderUI, you might consider using the [Powershell Module](#). This is especially suited for regularly performed automated tasks.

2.3.1 Display

The basic function is the display of the events contained in a recorder file.

Project data (Meta data)

The display gets much easier to read if [Project data \(Meta data\)](#) is available. Then also the names of devices and group addresses are displayed and values can be formatted according to their datapoint type.

Click on **Options** to show the MetaData list, then add new data using the  button.


If available, just add the ETS project (knxproj); but other formats are also supported.

If encrypted telegrams (KNX DataSecurity) occur, [Project data \(Meta data\)](#) is required either as ETS project or as ETS keyring file.


Filter


A very powerful filter function is at your hands to show or hide events/telegrams according to many criteria. Only events fulfilling the filter condition will be shown.

Click on **Filter** to show the Filter control.

A filter condition is a string with a specific [Syntax](#) which you might enter directly (for experts) or build up with a filter editor). You access the filter editor using the .

To apply the entered filter condition, click on .

The most recently used filter conditions are available via the .



Delete the filter condition using .


You can also use a displayed event to create a filter condition: click with the right mouse button on cell that should serve as basis for a filter and select the desired condition from the Filter context menu.

Search

You may search for an event/telegram in the list containing a specific text.


Click on **Search** to show the Search control.

Enter the search string and click on  or  (or use the keys F3 / Shift+F3) to search downwards/upwards from the currently selected event. All visible columns are searched. The search is case-independent.

The most recently used search texts are available via the .

Delete the search text using .

Bookmarks

You may bookmark  one or more events/telegrams in order to later jump quickly back to them. This function is available by the keyboard only:

Ctrl+F2 Toggle the bookmark at the selected event

Ctrl+Shift+ Clear all bookmarks

F2

F2 Jump to the next bookmark

Shift+F2 Jump to the previous bookmark

2.3.2 Plugins

As problem analysis tool, Recorder must be extensible in a flexible way. Only this guarantees that for specific problem cases millions of data records can be scanned quickly to extract the relevant information.

This is why Recorder allows adding Plugins.

All installed Plugins are listed in the **Plugins** menu (hidden if there are no plugins installed).

As an example, two of the available plugins will be discussed shortly. An up-to-date list can be found on out [Website](#).

Installation

Currently, you have to install plugins manually. Create a sub-directory Plugins in the Recorder program directory, if not yet existing.

It is recommended to create a sub-directory for each Plugin.

Extract the ZIP file into this sub-directory.

To uninstall, simply delete the sub-directory.

Invocation

Depending on the plugin, it is useful or required to restrict the data passed to the plugin by applying a suitable filter condition or by selecting events manually.

Invoke the plugin via the **Plugins** menu.

Close the plugin view using the  button.

2.3.2.1 Plugin Itgmbh.Recorder.Visualizers

This plugin provides a graphical visualization of values and statistics.

2.3.2.1.1 Value Chart

This plugin graphically displays values exchanged by group addresses. In addition, it can be used for the graphic display of bus load if any BusLoad events have been selected.

As it is not useful to display very many different group addresses or group values of different datapoint type, you should first apply a suitable filter condition or manually select the relevant telegrams.

The output graphics has the following features:

- Zoom in and out using the mouse scroll wheel or the Page Up/Down keys
- Move horizontally and vertically using the right mouse button or the cursor keys
- Go back to default view using the Home button

2.3.2.1.2 Statistics

This Plugin extracts some statistic information from the displayed telegrams

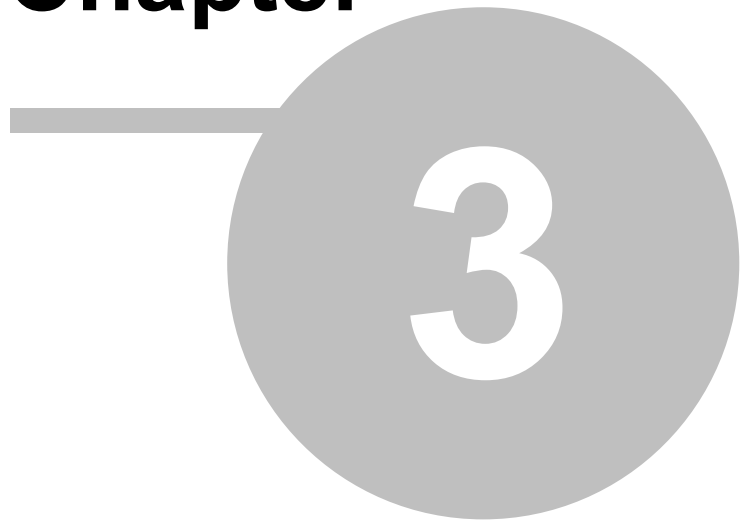
- the 10 devices most frequently sending
- the 10 most used group addresses

2.3.2.2 Plugin Itgmbh.Recorder.ConsistencyChecker

This plugin analyzes the recorded messages looking for inconsistencies. The following checks are performed:

-
- if there are invalid frames, the number of invalid frames is reported
 - if there are repeated frames, the number of repetitions is reported
 - if frames from a device is received with different hop counts on the same interface, this is reported with the individual address. The cause may be multiple devices with the same address or loops in the installation topology.
 - if meta data is available, any individual address or group addresses not found in the project is reported
 - if meta data is available, group values which do not fit to the datapoint type in the project are reported
 - receiving group values of different size on the same group address is reported
 - group value reads without corresponding response or with responses from multiple devices are reported

Chapter



Reference

3 Reference

3.1 Record

3.1.1 Configuration File

This is a JSON file containing:

- A Property "sources" containing an array of Sources (Interfaces), e.g.

```
"sources": [
  {
    "$type": "KnxSource",
    "id": "Knx1",
    "connectionString": "Type=KnxIpTunneling;HostAddress=192.168.1.60"
  }
]
```

- A Property "sinks" containing an array of sinks, e.g.

```
"sinks": [
  {
    "$type": "RollingFileSink"
  }
]
```

- Optionally, a property "metaData" containing one or more file names (separated by a semicolon), from which project data is read. See [Project data \(Meta data\)](#) for supported file formats.

For password-protected knxproj and knxkeys files, the password can be appended to the file name, separated by a colon. **Be sure to protect the configuration file against unauthorized read access in this case!**

Source

All source types have the following properties:

Property	Default value	Description
id		Identifies this source
isEnabled	true	Information whether or not this source is enabled or disabled

KnxSource

Uses a Falcon connection to record KNX telegrams.

Property	Default value	Description
connectionString		Falcon Connection String
interfaceMode	LinkLayer	LinkLayer: Group monitor Busmon: Bus monitor PreferBusmon: Prefers Bus monitor
retryOpen	00:01:00	If not 00:00:00, a failed connection attempt will be retried after this period of time.
reportBusload	false	If set, busload will be evaluated and reported in the given time period.

Sinks

All sink typed have the following properties:

Property	Default value	Description
isEnabled	True	Information whether or not this sink is enabled or disabled

Property	Default value	Description
sources	*	Specifies the sources - identified by their Id - that shall we written to this sink. Multiple Ids can be separated by semicolon. Wildcards ? and * are possible.
sourcesRegex		Alternatively to sources, the selection can be specified by a regular expression.

ConsoleSink

Writes the output to the console window (when running interactively)

Property	Default value	Description
config		Output format (see RecorderTextWriterConfig)

RollingFileSink

Writes the output to files

Property	Default value	Description
baseDir	%mydocuments%\Recorder	Base folder for the files to be created
fileFormat	Auto	File format: <ul style="list-style-type: none"> • Auto: determined by file extension • Binary: very compact binary format • Text: Text (CSV) • XML: Compatible with ETS monitor
config		Only for Text: Output format (see RecorderTextWriterConfig)
flushPeriod	00:00:05	Specifies the minimum time interval in which the file will be written.
isRemovableDrive	false	If set, the file will be closed after every write access; failed writes will be retried. Useful for removable drives (e.g. USB stick, network)
rollingStrategy	RollByEventData	Specifies how the file name is constructed
rollingStrategy.filePattern	Trace-{L:yyyy-MM-dd}.txt	Only for RollingStrategy=RollByEventData: file name, or sub-folder with file name (placeholders see below s.u.).
deleteAfterDays	0	If > 0, files older than this number of days will be deleted
synchronizeTo		Optional synchronization target; see Data Maintenance

SyslogSink

Writes the output to a syslog server

Property	Default value	Description
transport	Udp	One of: Local, Udp, Tcp, EncryptedTcp
syslogServer	localhost	Host name or IP Address
syslogPort	514	IP Port
serializer	Rfc5424	One of: Local, Rfc3164, Rfc5424
appName	Recorder	The AppName to use
config		Output format (see RecorderTextWriterConfig)

RecorderTextWriterConfig

Specifies how text output is constructed

Property	Default value	Description
prolog		Optional first row
frameReceivedEventFormat		Format for output of recorded telegrams (placeholders see below s.u.).
busloadEventFormat		Format for output of busload information (placeholders see below s.u.).
metaEventFormat		Format for output of meta events (Start/Stop/Connection loss etc.)

Placeholder

Name	Abbreviation	Description
LocalTimeStamp	L	Local time. Separated by a colon, the usual .NET DateTime format strings can be given (see Standard formats and Custom formats)
Timestamp	T	Time as UTC (world time), otherwise as LocalTimeStamp
Interface	I	ID of the interface
Data	D	Data
...		Additional properties see Event Properties . Note that some properties require that metaData is available.

3.1.2 Install as Service (Windows)

You may easily configure recording as Windows service using the Recorder user interface as described in [Unattended Operation](#).

If this is not possible (e.g. user interface not installed) or desired (e.g. automated installation), you may use the manual installation as described in this chapter.

1. Create a [Configuration file](#) and copy it to a suitable folder, e.g. in the Recorder program folder.
2. Open an administrative command prompt (e.g. by entering "cmd" in the start menu search field and then selecting "Run as Administrator" in the context menu of the appearing entry)
3. Navigate to the Recorder program folder
4. Enter: "Recorder -install *path_to_configuration_file*" (without the quotes)
This creates a service with standard settings (Service account: LocalService, Service name: Recorder).
A different service name can be specified using the "-n *name*" option.
A different service account can be selected after installation using the Windows Service Manager.

To remove the service, use the command: "sudo Recorder -uninstall" or "sudo Recorder -uninstall -n *name*".

Access rights

The -install invocation also adjusts permissions for the service account:

- Full access to the BaseDir of all RollingFileSinks
 - Write access to the Log folder %LOCALAPPDATA%\IT GmbH\Recorder\Log
- If you configure a different service account, you have to adjust the access permissions yourself.
Note that deinstalling the service does not remove the granted permissions.

Firewall

If IP interfaces are used, you might need to add Recorder.exe to your **Firewall** exception rules.

3.1.3 Install as Daemon (Linux)

The installation as Daemon is described here for **systemd**. If your Linux distribution has a different service infrastructure, please ask our support (please be aware that we cannot support every distribution and that this will be a charged service).

1. Create a [Configuration file](#) and copy it to a suitable folder, e.g. in the Recorder program folder.
2. Open a terminal window
3. Navigate to the Recorder program folder
4. Enter: "sudo Recorder -install *path_to_configuration_file* -user *user*" (without the quotes)
The given user is created if not yet existing.
This creates a service with name "Recorder". A different service name can be specified using the "-n *name*" option.

To remove the service, use the command: "sudo Recorder -uninstall" or "sudo Recorder -uninstall -n *name*". Zum Entfernen des Diensts gehen Sie genau so vor,

Access rights

The -install invocation also adjusts permissions for the service account:

- At least read access to the program directory and the configuration file
- Full access to the BaseDir of all RollingFileSinks
- If a KNX USB interface is used, read/write access to the raw HID device (e.g. /dev/hidraw0) is required

Note that deinstalling the service does not remove the granted permissions.

Firewall

If IP interfaces are used, you might need to add Recorder to your **Firewall** exception rules.

Raspberry Pi

We offer a pre-configured Raspberry Pi which is ready to use with a few simple steps. For details see the accompanying documentation.

3.2 Event Properties

Here we describe the individual properties that may be used in [Filters](#) and for column definitions for [Text Export](#).

An event has the following properties:

Property	Type	Description
Timestamp	DateTime	Timestamp (UTC)
LocalTimestamp	DateTime	Timestamp (local time)
Interface	String	Interface Id
EventType	Enum	Event Type (Meta, Frame, Busload)
Meta	see below	This object contains the information on Meta events, for other event types this object is Null.
Frame	see below	This object contains the information on Frame (telegram) events, for other event types this object is Null.
Busload	see below	This object contains the information on Busload events, for other event types this object is Null.

The **Meta** object has the following properties:

Property	Type	Description
Type	Enum	RecodingStarted Recording has started RecordingStoppedRecording has ended ConnectionLost Connection to the Interface has been lost ConnectionEstabliConnection to the Interface has been (re-)established shed InterfaceFrameLo The interface reported a frame loss ss SerializationFram Some events could not be written to the output eLoss

The **Frame** object has the following properties (properties marked with * require [Project data](#)):

Property	Type	Description
IsValidMessage	Boolean	The message is valid
RawData	Byte[]	Raw data
MessageCode	Enum	cEMI message code
IsExtendedFrame	Boolean	True for extended Frames, False for Standard Frames
IsStandardFrame	Boolean	True for Standard Frames, False for extended Frames
IsRepeated	Boolean	True for repeated Frames
IsSystemBroadcast	Boolean	True for System Broadcast (not for TP)
Priority	Enum	Message Priority (System, Alarm, High, Low)
IsDestinationIndividual	Boolean	True, if the destination address is an individual address (device address)
IsDestinationGroup	Boolean	True, if the destination address is an group address
HopCount	Byte	Routing counter
ExtendedFrameFormat	Byte	For extended Frames, the frame format (LTE only)
SourceAddress	KnxAddress	Source address
SourceName (*)	String	Name of the Source according to project information

Property	Type	Description
DestinationAddresses	KnxAddresses	Destination address
DestinationName (*)	String	Name of the Destination according to project information
TpduLength	Byte	Length information (TPDU Length)
Tpci	Enum	Transport Layer message code (TConnect, TDisconnect, TAck, TNak, TDataConnected, TData)
TSequenceNumber	Byte	Transport Layer sequence number
Apci	Enum	Application Layer message code (e.g. AGroupValueRead, AGroupValueWrite, AGroupValueResponse)
RawAsdu	Byte[]	Application Layer raw data For "short" (4-bit) APCIs this start at the incomplete byte where to upper two bits contain part of the APCI; for "long" (10-bit) APCIs it starts with the next byte.
Value (*)	Object	Group value interpreted according to project information (Datapoint Type). This is mapped as follows: DPT1 Boolean DPT2 Boolean? DPT3 Byte DPT4 Char DPT5 5.1 and 5.3: Float other: Byte DPT6 SByte DPT7 7.2 to 7.7: TimeSpan other: UInt16 DPT8 8.2 to 8.7: TimeSpan other: Int16 DPT9 Float DPT10 KnxTime DPT11 DateTime DPT12 UInt32 DPT13 Int32 DPT14 Float DPT16 String DPT17 Byte DPT18 KnxSceneControl DPT19 KnxDateTime DPT20 Byte DPT21 Byte DPT26 KnxSceneInfo DPT29 Int64 DPT23 Color 2
FormattedValue (*)	String	Group value formatted according to project information (Datapoint Type).

The **Busload** object has the following properties

Property	Type	Description
Maximum	Byte	Maximum Bus load (registered over a second) in % since the last report
Mean	Byte	Mean Bus load in % since the last report

3.3 Project data (Meta data)

For the following data, information from the ETS project is required:

- Device name
- Group address name
- Datapoint type
- Decryption of KNX DataSecure telegrams

Recorder can read this data from several file formats

ETS Project Export

This is the easiest option as the file contains all required data.

All knxproj formats of ETS4 and ETS5 are supported.

If the project is password protected, you will be asked for the password after opening.

ETS Keyring

ETS can export KNX DataSecurity keys into a "Keyring File" (knxkeys). Also here, you will be asked for the password. The keyring file is not necessary if also the knxproj file has been imported.

XML Files

XML files in the following formats can be imported:

- ETS4/ETS5 Group address export
- Elvis2 XML Export

Text/CSV Files

Text/CSV files with the following properties can be imported:

- Character set UTF8
- Separator Tabulator, Comma or Semikolon
- With caption row
- Group addresses (2- or 3-level) / individual addresses in a column with caption "Address"
- Names in a column with caption "Sub" or with a caption containing "name" or "Name"
- Datapoint type in column with caption "DatapointType" or "DPT"

3.4 Filter Syntax

Constants

Numerical values can be entered as decimal (e.g. 20) or hexadecimal (e.g. 0x14).

Strings are enclosed in single (e.g. 'Text') or double (e.g. "Text") quotes.

Date/Time values are written as strings in the format "yyyy-MM-dd HH:mm:ss.fff" (e.g. "2020-09-10 10:30:15.000") (the millisecond and second parts can be omitted).

Group addresses can be entered as numerical value, in two-level (e.g. 2/300) or three-level (e.g. 2/1/44) notation.

Individual addresses can be entered as numerical value or in the usual ETS notation (e.g. 10.2.101).

Lists are enclosed in [] and may contain values separated by comma (e.g. [1,2]), ranges (e.g. [1-9,11-19]) or addresses with wildcards (e.g. [1/2/*,2/0/*] - also mixed).

Basic conditions

For [Properties](#) of type Boolean (truth value) you may write:

Syntax	Erklärung	Beispiel
<i>property</i>	fulfilled if this property has the value true.	Frame.Message.IsDestinationGroup
! <i>property</i>	fulfilled if this property has the value false	!Frame.Message.IsDestinationGroup

For numerical and date/time [Properties](#) the following conditions are available:

<i>property = value</i>	fulfilled if this property has the specified value	Frame.Message.HopCount = 3
<i>property != value</i>	fulfilled if this property does not have the specified value	Frame.Message.HopCount != 3
<i>property < value</i>	fulfilled if the property value is less than the specified value	Frame.Message.HopCount < 3
<i>property <= value</i>	fulfilled if the property value is less than or equal to the specified value	Frame.Message.HopCount <= 3
<i>property >= value</i>	fulfilled if the property value is greater than or equal to the specified value	Frame.Message.HopCount >= 3
<i>property > value</i>	fulfilled if the property value is greater than the specified value	Frame.Message.HopCount > 3

For enumerated [Properties](#) (Enum) the following conditions are available:

<i>property = value</i>	fulfilled if this property has the specified value	Frame.Message.Priority = Alarm
<i>property != value</i>	fulfilled if this property does not have the specified value	Frame.Message.Priority != Alarm
<i>property : [list]</i>	fulfilled if the property value is in the list	Frame.Message.Priority : [Alarm,High]
<i>property !: [list]</i>	fulfilled if the property value is not in the list	Frame.Message.Priority !: [Alarm,High]

For string [Properties](#) the following conditions are available:

<i>property = value</i>	fulfilled if this property has the specified value	Interface = "KNX1"
<i>property != value</i>	fulfilled if this property does not have the specified value	Interface != "KNX1"
<i>property ~ pattern</i>	fulfilled if the property value matches the given pattern (regular expression)	Interface ~ "^KNXd\$"
<i>property !~ pattern</i>	fulfilled if the property value does not match the given pattern (regular expression)	Interface !~ "^KNXd\$"
<i>property ? value</i>	fulfilled if this property contains the specified value	Interface ? "KNX"
<i>property !? value</i>	fulfilled if this property does not contain the specified value	Interface !? "KNX"

For KNX addresses the following conditions are available:

<code>property = value</code>	fulfilled if this property has the specified value	<code>Frame.Message.DestinationAddress = 2/0/1</code>
<code>property != value</code>	fulfilled if this property does not have the specified value	<code>Frame.Message.DestinationAddress != 2/0/1</code>
<code>property : [list]</code>	fulfilled if the property value is in the list	<code>Frame.Message.DestinationAddress : [2/0/*]</code>
<code>property !: [list]</code>	fulfilled if the property value is not in the list	<code>Frame.Message.DestinationAddress !: [2/0/*]</code>

For the tag list the following conditions are available:

<code>property ? value</code>	fulfilled if this property contains the specified tag	<code>TagList ? "tag1"</code>
<code>property !? value</code>	fulfilled if this property does not contains the specified tag	<code>TagList !? "tag1"</code>

Composite conditions

The basic conditions can be combined using AND (or equivalently &&), OR (or ||) and NOT (or !). The rule "AND before OR" (like "multiplication before addition") applies, as far as not a different evaluation order is specified by parentheses.

3.5 Powershell

Before you can use the Powershell module in a script, it has to be imported:

```
Import-Module pfad\Itgmbh.Recorder.Powershell.dll
```

After this step, the two Cmdlets Get-RecorderFrame and Get-RecorderProjectInfo are available.

3.5.1 Cmdlet Get-RecorderFrame

Reads Recorder events from one or more files.

```
Get-RecorderFrame
  [-Path] <String[]>
  [<Common Parameters>]
```

```
Get-RecorderFrame
  -LiteralPath <String[]>
  [<Common Parameters>]
```

Description

The Get-RecorderFrame cmdlet reads events from the input files given in the -Path or -LiteralPath parameters, merging them to a sequence of event objects (properties see [Event Properties](#)). The input files must be valid recorder files.

Examples


Example 1: Merge recorder files and write them to a CSV file

This example merges all recorder files in a directory F:\RawFiles and writes all group messages as CSV file F:\Processed\All.txt. Use project F:\Project.knxproj to determine the datapoint type.

```
$proj = Get-RecorderProjectInfo F:\Project.knxproj

Get-RecorderFrame F:\RawFiles -ProjectInformation $proj |
  Where-Object IsDestinationGroup |
  Select-Object LocalTimestamp, DestinationAddress, GroupValue |
  Export-Csv -Path F:\Processed\All.txt
```

Parameter

Property	Type	Default Value	Description
-Path	String[]		Path name of the recorder file(s). Wildcards are allowed. You may also include directory paths, in this case all recorder files in these directories will be read The value can also be passed via a Pipe. Either -Path or -LiteralPath has to be given.
-LiteralPath	String[]		As -Path but without wildcards
-Filter	String		Optional Filter, see Filter Syntax . Alternatively, of course filtering can also be done with Powershell means ( Where-Object).
-ProjectInformation	RecorderProjectInfo		Optional Meta data (Project data). Only needed if Source or Destination Names or interpreted group values shall be accessed; in addition for decrypting KNX DataSecurity telegrams.
-TimeFormat	String		When reading text files, the time format used
-BusmonFrameFormat		Unknown	Only for Bus monitor traces: specifies the frame format Unknown: determined according to some heuristics Tp: Twisted Pair Rf: KNX RF
-Culture	CultureInfo	(user setting)	For culture-dependent formatting
-RemoveDuplicates	Boolean	False	Remove Duplicates (several occurrences of the same telegram within a short time)

Input

Via command chaining (Pipe) the output of another command can be interpreted as Path parameter.

Output

The cmdlet produces a sequence of event objects.

3.5.2 Cmdlet Export-RecorderFile

Writes Recorder events to a file.


```
$events | Export-RecorderFile
  [-Path] <String[]>
  [<Common Parameters>]
```

Description

The Export-RecorderFile cmdlet writes the events from the input pipe to a file given in the -Path parameter.

Examples

Example 1: Merge recorder files and write them to a Recorder XML file

This example merges all recorder files in a directory F:\RawFiles and writes the output as XML file F:\Processed\All.xml.

```
Get-RecorderFrame F:\RawFiles |
  Export-RecorderFile -Path F:\Processed\All.xml
```

Parameter

Property	Type	Default Value	Description
-Path	String		Path name of the output recorder file
-FileFormat	FileFormat	Auto	The output file format
-ProjectInformation	RecorderProjectInfo	Unknown	Optional Meta data (Project data). Only needed if the FileFormat is Text and Source or Destination Names or interpreted group values shall be accessed.
-Culture	CultureInfo	(user setting)	For culture-dependent formatting

Input

The events are passed via command chaining.

Output

None.

3.5.3 Cmdlet Get-RecorderProjectInfo

Reads project information (meta data).

```
Get-RecorderProjectInfo
  [-Path] <String[]>
  [<Common Parameters>]
```

```
Get-RecorderRecorderProjectInfo
  -LiteralPath <String[]>
  [<Common Parameters>]
```

Description

The Get-RecorderProjectInfo cmdlet reads the relevant data from any of the supported files - see [Project data \(Meta data\)](#).

Examples

Example1: Reading a project file

```
$proj = Get-RecorderProjectInfo myproject.knxproj
with Password
$password = Read-Host "Password" -asSecureString
$proj = Get-RecorderProjectInfo myproject.knxproj -Password $password
```

Example2: Reading CSV files

```
$proj = Get-RecorderProjectInfo groupinfo.csv deviceinfo.csv -Encoding UTF8
```

Parameter

Property	Type	Default Value	Description
-Path	String[]		Path name of the meta data file(s). Wildcards are allowed. You may also include directory paths, in this case all recorder files in these directories will be read The value can also be passed via a Pipe. Either -Path or -LiteralPath has to be given.
-LiteralPath	String[]		As -Path but without wildcards
-Password	SecureString		Optional password for protected project files (knxproj) and keyring files (knxkeys)
-GroupNameFormat	String	Name	Determines how the group address text is constructed. Given as text with placeholders <ul style="list-style-type: none"> • {Name} or {Description} for the Name/Description of the group address • {^Name} or {^Description} for the Name/Description of the parent group range (2-level: main group, 3-level: middle group) • {^^Name} or {^^Description} for the Name/Description of the grand-parent group range (3-level: main group)
-DeviceNameFormat	String		Determines how the device text is constructed. Given as text with placeholders <ul style="list-style-type: none"> • {Name} or {Description} for the Name/Description of the device If not given, the Name is used if entered by the user, otherwise the Description, if it is empty, the product name
-Encoding	Encoding	UTF8	For Text (CSV) files the text encoding
-Culture	CultureInfo	(user setting)	For culture-dependent formatting

Input

Via command chaining (Pipe) the output of another command can be interpreted as Path parameter.

Output

The cmdlet returns a [RecorderProjectInfo](#) object. This can be passed as parameter to the [Get-RecorderFrame](#) Cmdlet.

3.5.4 Class RecorderProjectInfo

Description

Examples

Properties

Property	Type	Description
GroupNames	Hashtable	Maps group addresses to texts
DeviceNames	Hashtable	Maps individual addresses to texts
DatapointTypes	Hashtable	Maps group addresses to datapoint types

3.6 Advanced Topics

3.6.1 Create Plugins

Important Hint

For further questions about the creation of plugins please contact our support.
We reserve the right to modify the interfaces described here at any time.

Create a Plugin

To create a plugin, you need a development environment capable of creating .NET Assemblies for .NET Framework 4.6.1.

The Plugin has to be created as Class Library and must reference the following assemblies:

- Itgmbh.Recorder.Core
- Itgmbh.Recorder.UIPlugin

It has to contain one or more plugin classes implementing the interface `Itgmbh.Recorder.IUIPlugin`:

- `MenuTitle` must return a Text used for display in the Plugin menu
- `MenuIcon` can optionally return the URI (e.g. a Pack-URI) of an image for the Plugin menu.
- `CreateControl` must create the WPF Control to be displayed. The following is passed as parameters:
 - `displayedEvents`: the list of all events after applying any filter
 - `selectedEvents`: the list of currently selected events
 - `serviceProvider`: can be used to access services like `IProjectInformation` or `IGroupValueFormatter`.

In addition, a `pluginmanifest` file has to be created describing the plugin. In a simple case this might look like that:

```
<?xml version="1.0" encoding="utf-8" ?>
<PluginManifest AssemblyName="MyCompany.MyAssembly">
  <Plugin PluginType="UIPlugin" TypeName="MyCompany.MyAssembly.MyPluginClass1"/>
  <Plugin PluginType="UIPlugin" TypeName="MyCompany.MyAssembly.MyPluginClass2"/>
</PluginManifest>
```

Create a ZIP file from the `pluginmanifest` file, the Plugin assembly and possibly other files. This ZIP file can then be installed as described in [Plugins](#).

3.7 FAQ

▣ Where do I find log files for troubleshooting?

By default, the Recorder tools write log files to help you and our support to analyze and problems at these locations:

Windows Applications: `%localappdata%\IT GmbH\Recorder\Log`

Windows Services: `%programdata%\IT GmbH\Recorder\Log`

Linux: as configured in `Recorder.config`

Chapter



Imprint

4 Imprint

The information and data contained in this document are subject to change without prior notice. The names and data used in examples are fictitious if not noted otherwise. You may not reproduce or copy this document, or any portion thereof, for any purpose without the explicit written consent of IT GmbH, regardless of the mode and means, electronically or mechanically.

© 2018-2021 IT Gesellschaft für Informationstechnik mbH

IT Gesellschaft für Informationstechnik mbH
An der Kaufleite 12
D-90562 Kalchreuth
Germany

All rights reserved.

Windows is a trademark of the Microsoft Corporation.

ETS is a registered trademark of KNX Association c.v.b.a.

Chapter



Contact

5 Contact

Orders

For orders, please contact our Sales.

Tel: +49 (0) 911 518349-0 (Mo-Fr 9 a.m. to 4 p.m.)

Fax: +49 (0) 911 5183688

Email: vertrieb@it-gmbh.de

Support

If you have questions or problems, you may contact our Support:

Tel: +49 (0) 911 518349-10 (Mo-Fr 9 a.m. to 4 p.m.)

Fax: +49 (0) 911 5183688

Email: support@it-gmbh.de

WEB:  [Problem Report](#)

Newsletter

We would like to inform you about useful additional products and news. If you wish, please register for our newsletter. Click on the link below to access our newsletter registration page. You can cancel your registration at any time without giving any reasons.

 [Subscribe to the Newsletter](#)



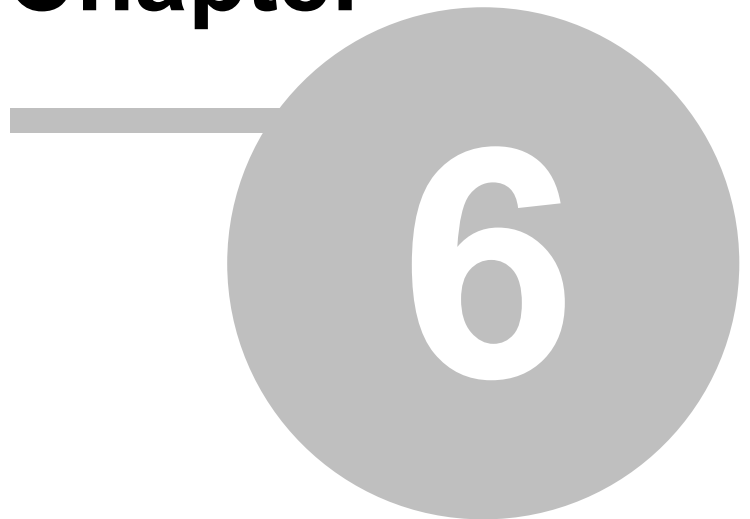
IT GmbH

An der Kaufleite 12

D-90562 Kalchreuth

 <https://www.it-gmbh.de>

Chapter



Feedback

6 Feedback

Please help us to improve our ETS Apps. Your feedback may influence further development so that in subsequent versions and manual editions your wishes and requirements may be taken into consideration.

We look forward to receiving your comments and wishes on the content, representation of associations as well as comprehensibility regarding the program parts or the documentation. Please also let us know if you have any improvement suggestions with regard to support, training or sales.

 [Feedback via IT Website](#)

Chapter



Open Source Licenses

7 Open Source Licenses

The following Open Source components are used in Recorder.

Name	Link	License
Autofac	 https://github.com/autofac/Autofac	MIT License Web Link:  https://github.com/autofac/Autofac/blob/develop/LICENSE
BouncyCastle	 https://www.bouncycastle.org/csharp/index.html	MIT License Web Link:  https://www.bouncycastle.org/licence.html
CommandLineParser	 https://github.com/commandlineparser/commandline	MIT License Web Link:  https://github.com/commandlineparser/commandline/blob/master/License.md
ServiceLocator	 https://servicelocation.codeplex.com/	Microsoft Reciprocal License (Ms-RL) Web Link:  https://servicelocation.codeplex.com/license
DotNetZip	 https://dotnetzip.codeplex.com/	Microsoft Reciprocal License (Ms-RL) Web Link:  https://dotnetzip.codeplex.com/license
Extended WPF Toolkit	 https://wpftoolkit.codeplex.com/	Microsoft Reciprocal License (Ms-RL) Web Link:  https://wpftoolkit.codeplex.com/license
Google.ProtocolBuf	 https://github.com/protocolbuffers/protobuf	Web Link:  https://github.com/protocolbuffers/protobuf/blob/master/LICENSE
Log4net	 http://logging.apache.org/log4net/	Apache License Version 2.0, January 2004 Web Link:  http://logging.apache.org/log4net/license.html
MVVM Light	 http://www.mvmlight.net/	MIT License Web Link:  http://mvmlight.codeplex.com/license
Newtonsoft.Json	 http://james.newtonking.com/json	MIT License Web Link:  https://github.com/JamesNK/Newtonsoft.Json/blob/master/LICENSE.md
Ooki Dialogs	 http://www.ookii.org/software/dialogs	Ookii.Dialogs License
OxyPlot	 http://www.oxyplot.org/	MIT License Web Link:  http://opensource.org/licenses/MIT

Index

	Service	12, 23
	Support	38
	System Requirements	5
- B -	- T -	
Bookmark	Tunneling	8
- C -	- U -	
Configuration File	USB	8
Contact		
- F -		
Feedback		40
File		9
Filter		17
- H -		
Hotline		38
- I -		
Imprint		36
Interface		8
- J -		
json		21
- L -		
Licensing		6
- O -		
Operating system		5
Output		9
- R -		
Routing		8
- S -		
Search		17

