Datalogger tool

TXV 003 30.01
second eddition
July 2011
subject to changes

History of changes

Date	Eddition	Change description
May 2011	1	First edition (reffers to the state of the tool in version 1.0.29)
July 2011	2	Added chapter 3. Datalogger in the PLC

Obsah

1 Introduction

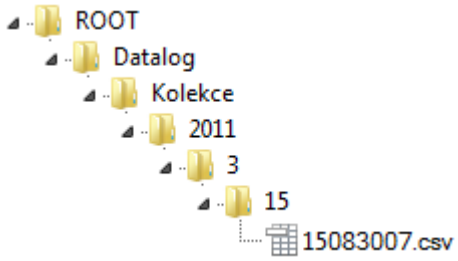
Datalogger is a tool designed for creation of automatic system for data and events saving within the PLC Tecomat systems. The record is saved onto the memory card which must be a part of the relevant systém. The final file is saved in the standard CSV format (*Comma-Separated Values*) to enable easy processing in some of the commonly used tab editors and databasis.

1.1 Function principle

The tool was designed so, that it enables the user to design sofisticated tools for the text editting of the events within PLC systems. The structure of the Datalogger can be devided into four hierarchicaly sequenced parts.

First, root, part is the *Datalogger* itself. This data structure is the main structure gathering all information about the Datalogger configuration. Each *Datalogger* configuration has always assigned its own unique name and generated CSV file with the record of the events is saved into the directory structure initiated by the directory of this name. The directory structure is standardly set up in the main work directory *ROOT*, or alternatively, in the directory *WWW* for accession of the created files via the web browser.

Another group of data structure within the Datalogger hierarchy is *Collection*. Under the Collection can be realized a sum of controlled parameters (Signals) which values should be within the specified time recorded into the CSV file. Together with the creation of each Collection, there is , in the automatically generated directory structure, created another directory of the similar name that was assigned to the Collection. The whole directory path to the CSV record is also completed for a date of the relevant record in the sequence *year* → month → *day* where the last of the named directories is optional. The name of the CSV file is composed of the day and the time of the file creation. The final structure then can look as follows:



The configuration set Collection also bears information about default format of entering and creation of CSV files, simplified specification of commonly used Signals such as time or date of the entry and the definition of the Collection type. The Collection type is an important feature deciding about events that will initiate the entry into the CSV file. Each Collection can consist of three types:

- *periodic*
- *events*
- *signals*

The first Collection type is *periodic* that does periodic synchronic entry of all defined Signals, in another words, after the expiration of the specified time interval, the whole picture of status of all Signals is entered into the CSV file which the relevant Collection owns. This type of the Collection can have a stable period or two-stated period controlled via the selected system variable. It is also possible to select a function of asynchronic reset of the period counter undertaken on the basis of the event generated by some of the owned Signals defined in the Collection.

Second type is the Collection of *events*. In this case, the synchronic entry is undertaken in the same way as in the previous case, but, now on the basis of the change of the master variable of the Collection. Asynchronic entry of all signals can be, in case of need, also initiated by the change of any of them.

Within the Collection of the *signal* type are individual Signals entered into the file only asynchronicaly. This means that the entry is undertaken only on the basis of events that related Signal defines. On the contrary to the previous two types, there is only entry of such signal undertaken that initiated the requirement for entry.

The third item of the Datalogger hierarchy is already mentioned Signal that is always in possession of the Collection. The main parameter of this data structure is selected variable of the user program which value is controlled and entered into the CSV file.The part of it is also a specification of the entry format of the value of the specified variable. Optionally, it is also possible to specify the event that will initiate the asynchronic entry of signals into the file. This event can be the change of the variable value or the transit of the value over the specified level. If the Signal is a part of the Collection of the signal type then this event should be always specified. The signal has always assigned its own unique name.

The last optionable item of the Datalogger structure is *Data*. These items can be created hierarchicaly under the nod of the *Signal* type but only in a case when mother Collection is of a signal type. Data items enable the user to add to the entry additional values. In another words, the asynchronic entry of the Signal value initiated on the basis of its change is accompanied by the entry of the set of other optionable varibale values. The part of the content of the data structure is, apart from the description of the selected variable itself, also a format in which the values of Data items should be entered. Each Data item is identified by its assigned name.

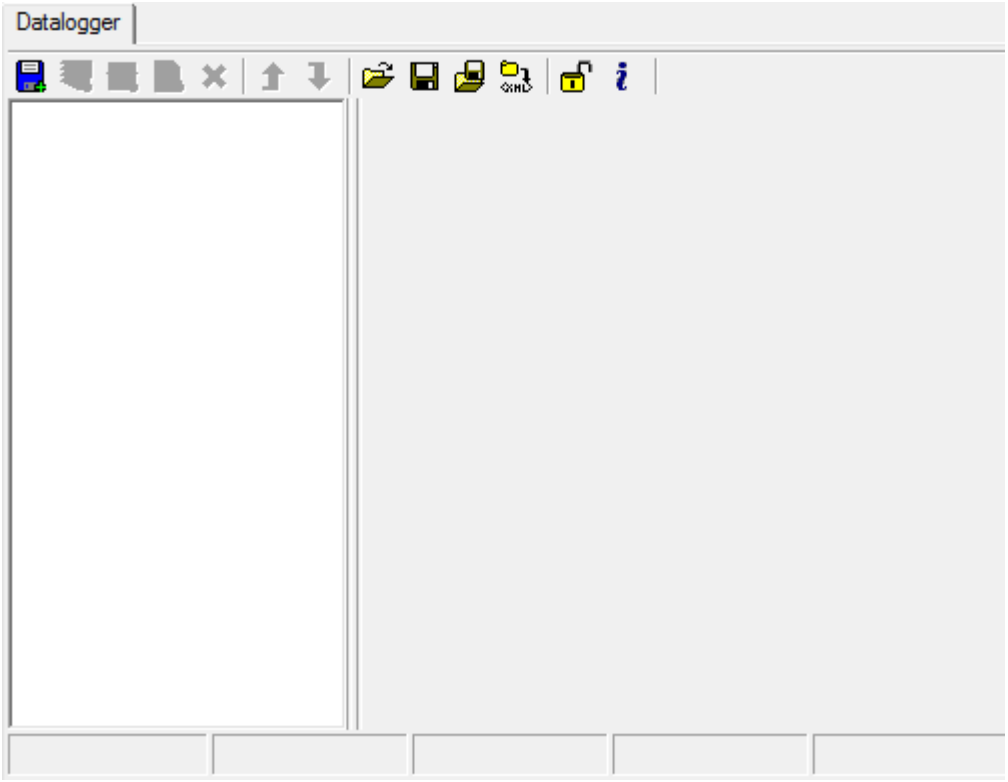
The whole Datalogger configuration is generated in the XML file form which is then taken over by the PLC system. The content of the file can be modified by hand, however, only experienced users should do so. To ensure the right functionality, it is recommended to use the tool within the Mosaic environment described in the following part of this document.

2 Datalogger tool operation

2.1 Tool start up

The tool is activated in the Mosaic environment either from the main menu ***Tools*** → *Datalogger*, or from the tool bar via the icon .

After the start up the tool window will open:



2.2 Tool window

The window consists of four parts:

1) Upper tool bar
2) Left panel of the description structure
3) Editing area
4) Status line

2.2.1 Upper tool bar

Tool bar buttons

Add *Datalogger* – Creates a new section of the *Datalogger* type in the description structure. Simultaneously, the editing area of this section is displayed.

Add *Collection* – Creates the new section of the *Collection* type in the description structure under the selected master nod of the *Datalogger* type. Simultaneously, the editing area of this section is displayed.

Add *Signal* – Creates the new section of the *Signal* type in the description structure under the selected master nod of the *Collection* type. Simultaneously, the editing area of this section is displayed.

Add *Data* – Creates the new section of the *Data* type in the description structure under the selected master nod of the *Signal* type. Simultaneously, the editing area of this section is displayed. *Data* of the item can be added only in such case when the master Collection in the related hierarchy is of the *signal* type (see chapter Chyba: zdroj odkazu nenalezen).

Delete – Delete the selected item in the description structure.

Shift up – Shift the selected item of one possition up.

Shift down – Shift the selected item of one possition down.

Open – Displays a dialog for file opening (**.dlog, *.xml*) with the saved configuration. After the file selection the current configuration is deleted and the configuration from the file is uploaded.

Save – Save actual configuration into the default directiory of the tool as a file **.dlog*.

Save as – Displays a dialog for saving of actual configuration into the selected directory. The configuration can be saved as a file **.dlog* or **.xml.*

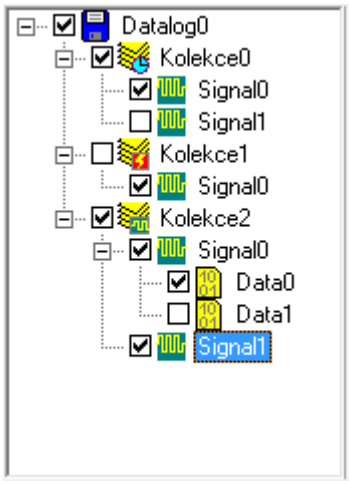
Compile the Datalogger – The generation of the XML code for the PLC. The generated XML code is saved in the project into the subdirectory *SendRoot.* This directory is automaticaly synchronized with the PLC during the program code transmission (PLC must support a file system, otherwise, the synchronisation does not run).

Unlock/Lock – Unlock/Lock the tool for editing. After the locking, it is possible to browse the actually uploaded configuration but it can not be edited.

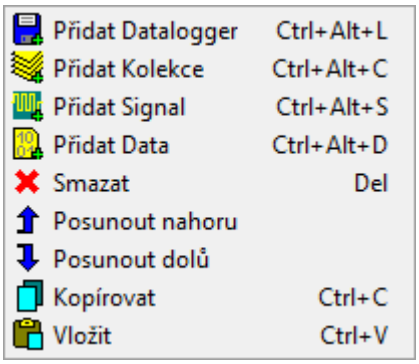
About the tool – Information about the tool version.

2.2.2 Left panel of the description structure

The panel serves for visualisation of the hierarchic tree structure of the Datalogger and for selection of individual items for editing. Also, it is possible, using the thick boxes located at each nod, to easily enable or forbid the relevant item (nod) in the overall configuration.



The root of the tree structure is always the section of the *Datalogger* type that contains subsections describing the whole configuration. The following items of the tree hierarchy are items of the *Collection* type into which it is possible to save nods of the *Signal* type. In the case when the mother Collection is of the *signal* type (see chapter Chyba: zdroj odkazu nenalezen), the tree structure can be ended by items of the *Data* type which are subordinated to partial Signals. The meaning of individual item types for the whole configuration was explained in the chapter Chyba: zdroj odkazu nenalezen. By right mouse button, it is possible to evoke local menu in the selected item of the tree.



Add *Datalogger* – Duplicates the button of the tool bar described in the chapter Chyba: zdroj odkazu nenalezen.

Add *Collections* – Duplicates the button of the tool bar described in the chapter Chyba: zdroj odkazu nenalezen.

Add *Signal* – Duplicates the button of the tool bar described in the chapter Chyba: zdroj odkazu nenalezen.
Add *Data* – Duplicates the button of the tool bar described in the chapter Chyba: zdroj odkazu nenalezen.
Delete – Duplicates the button of the tool bar described in the chapter Chyba: zdroj odkazu nenalezen.
Shift up – Duplicates the button of the tool bar described in the chapter Chyba: zdroj odkazu nenalezen.
Shift down – Duplicates the button of the tool bar described in the chapter Chyba: zdroj odkazu nenalezen.
Copy – Copy the selected item into the box.
Insert – Insert the coppied item from the box.

2.2.3 Editing area

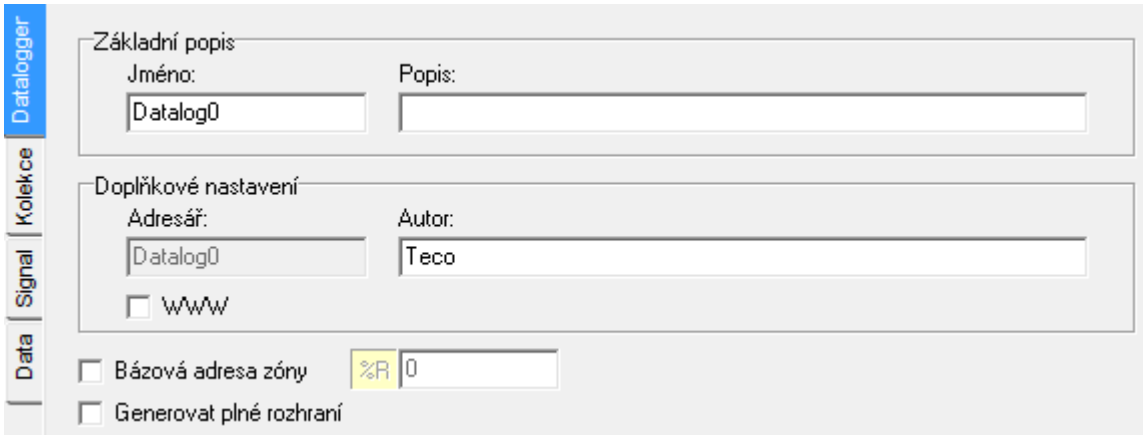
Editing area is blank in the default status. After selection of one of tree nods in the panel of the description structure, the relevant editting area is displayed.

Editing area can be according to the selected item of four types:

1) Editing area of the *Datalogger* section
2) Editing area of the *Collection* section
3) Editing area of the *Signal* section
4) Editing area of the *Data* section

Editing area of the *Datalogger* section

When selecting the nod of the *Datalogger* type in the tree description structure, the relevant editing area is displayed. The identification which editor we are in is indicated on the left side of the relevant tab.



Datalogger editor consists of following parts:

- ***Basic description***
  - *Name* – Representative name of the whole configuration of the Datalogger. The lenght is limited to 8 characters.
  - *Description* – Text description of the Datalogger configuration enabling the user to distinguish accurately the relevant configuration from other instances. The lenght is limited to 59 characters.

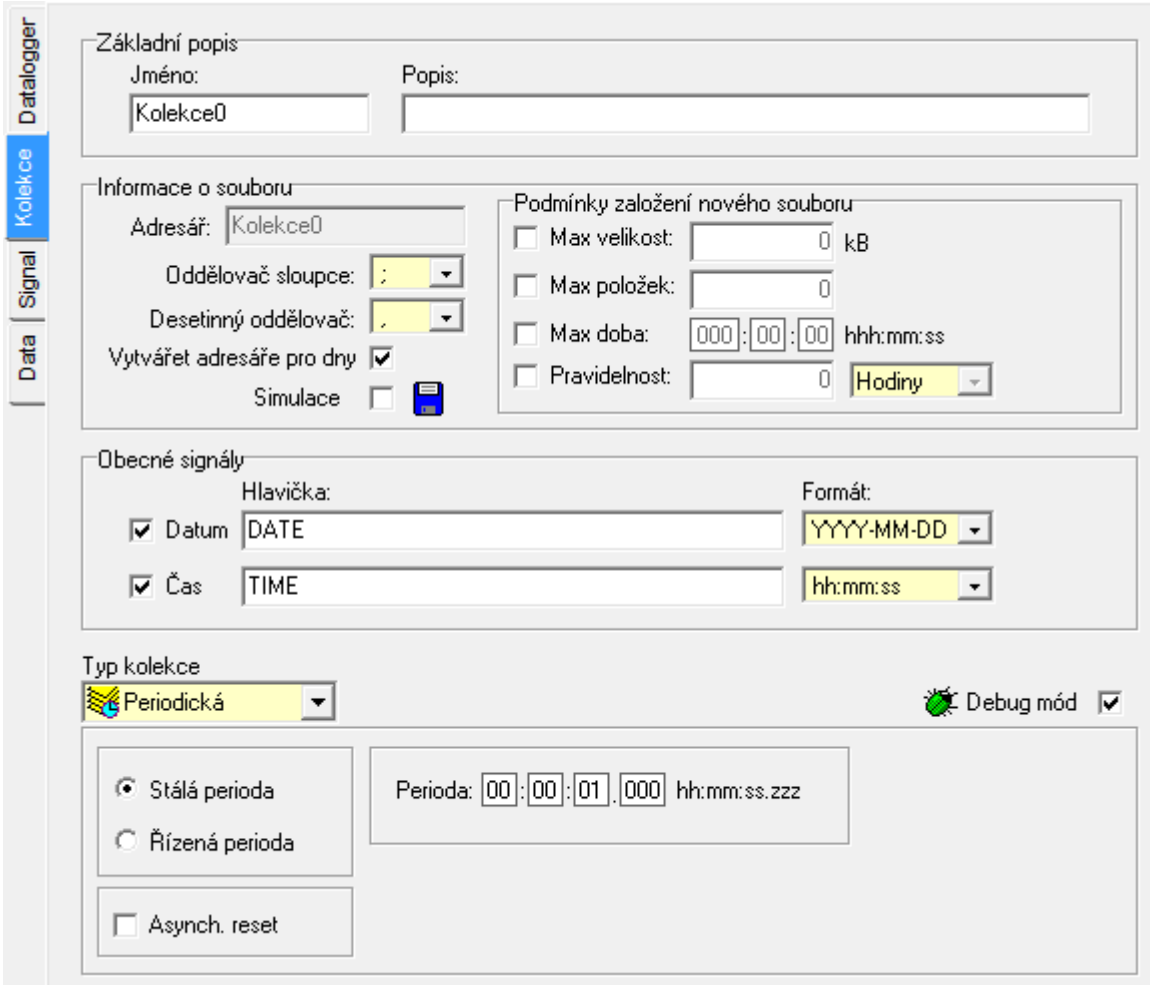
- ***Additional settings***
  - *Directory* – This element can not be editted. It serves to identification of the default directory into which the files are saved. The name of the directory corresponds to the name of the Datalogger configuration.
  - *Author* – The possibility of configuration author entry. The lenght is limited to 59 characters.
  - *WWW* – By thicking this option, the directory structure of the Datalogger will be saved under the directory of the web server. Reversely, the structure is set up in the default root directory of the unit.

- ***Zone base address*** – By thicking of this fiels and entry of the numeric value, the specified default base address of the memory (PLC notepad, area %R) is used into which the interface between Datalogger and the user program will be located. If the zone address is not specified, the interface will be saved to the first free space in the memory.

- ***Full interface generation*** – After thicking this option, the full interface for program control by Datalogger in the PLC will be generated. Reversely, only the restricted interface is enabled which enables the user to control and monitor the Datalogger program only partially. Simultaneously, also memory requirements are restricted. Datalogger program control is described in detail in chapter 3.

Editing area of the *Collection* section

When selection the nod *Collection* in the tree of the description structure, the relevant editing area is displayed. The identification which editor we are in is indicated on the left side of the relevant tab.



Collection editor consists of following parts:

- ***Basic description***
  - *Name* – Representative name of the whole configuration of the Collection. The lenght is limited to 8 characters.
  - Description – Text description of the Collection configuration enabling the user to distinguish accurately the relevant configuration from other instances. The lenght is limited to 59 characters.

- ***File information*** – Gathers the settings for directory structure and files generation.
  - *Directory* – This feature can not be edited. It serves for indication of the name of the Collection directory that is created in the default Datalogger directory into which individual records are entered. The name of the directory corresponds to the name of the Collection .
  - *Column separator* – Character that will be used as a column separator within the entry.
  - *Decimal separator* – Character that will be used as a decimal separator of the numeral

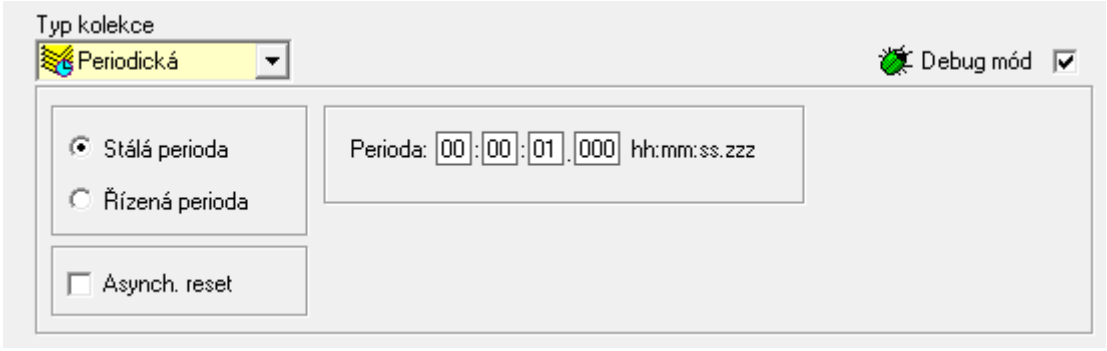
values within the entry.

- *Create directories for days* – By thicking this option the directory structure will create an individual directory for the day of the creation. Conversely, the directory structure will be terminated by the number of the month. The day of the entry is a part of the file name.
- Simulation – By thicking this option the Datalogger will work in the simulation mode. The interface of the logging system will work in a standard way as during the real entry to the file, however, the real entry into the memory will not be undertaken. This mode is suitable especially for debugging purposes.
- *Conditions for new file setup* – The user have the possibility to specify conditions for new files setup.
  - *Max size* – By thicking this option new file will be always created when the specified size is exceeded (in kB).
  - *Max items* – New file will be always created when the defined number of entries to the file is exceeded.
  - *Max time* – New file will be always created when the maximum file age is exceeded.
  - *Regularity* – By thicking this option new file will be created in regular intervals. Interval units can be *Hours*, *Days*, *Weeks* or *Months*.

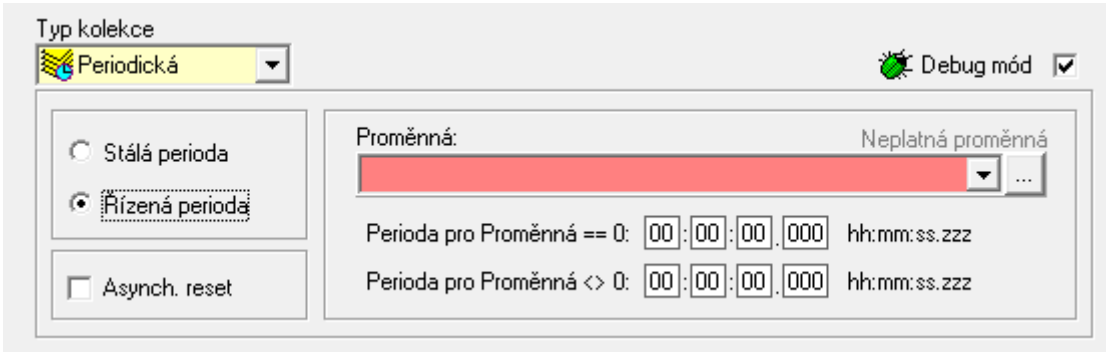
- ***General signals*** – Enables the use of Signals date and time within the Collection.
  - *Date* – each entry in the file will contain a column with the actual entry date.
  - *Time* – each entry in the file will contain a column with the actual entry time.
  - *Header* – specify the header text of the column *Date*/*Time* in the file.
  - *Format* – specify the format of the item entry *Date*/*Time* in the file.

- ***Debug mode*** – Switch on/off the debugging mode. By thicking this option the entry in the file will contain debugging texts (see item Debug text in the signal definition).

- ***Collection type*** – Selection of Collection type. There are three types of Collections available:
  -  ***Periodic*** – Data record is undertaken synchronously within the regular intervals. The first entry is done during the file setup. Then, values of all Signals in the Collection are saved periodicaly. Entries can be saved also asynchronously (out of the selected period) based on the change of the value or change of the level of individual signals in the Collection. Periodic Collection can be with stable or controlled period.
    - *Stable period* – The period is firmly set by the value of the time interval *Period*.



- *Controlled period* – The value of the time interval of the period can be controlled by the selected variable *Variable*. The variable name can be set by hand or it can be selected using the standard dialogue that can be initiated by clicking the button ... The editor always validates the selected variable. Parameters *Period for Variable == 0* and *Period for Variable <> 0* determines values of the period for a case of the zero/non-zero variable.

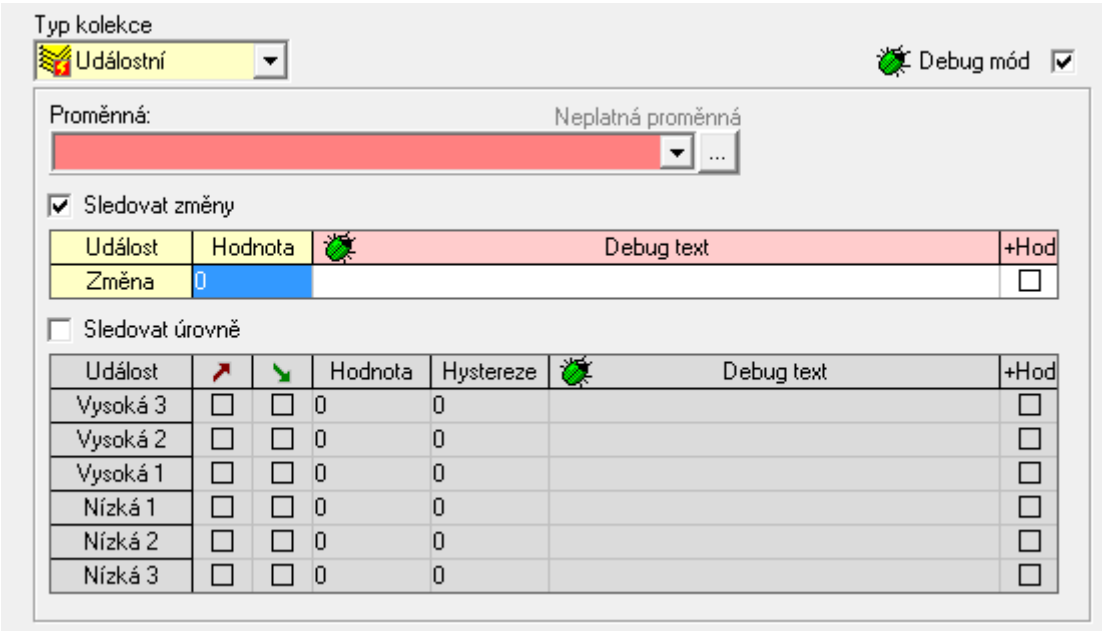


- *Asynch. reset* – By thicking this option the timer of the period will be automatically reset in case of variation of the asynchronous event specified by one of the Signals of the relevant Collection (see chapter).

- ***Events*** – Synchronous data record is undertaken on the basis of the value change or the change of the variable of the Collection. All values of all Signals in the Collection are always saved into the entry. The first data entry is done during the file setup. Entries can be saved also asynchronously (apart from the synchronous entry) based on the change of the value or the change of the level of the individual signals in the Collection.
  - *Variable* – The control variable that will be used for synchronous entry initialization. The variable name can be set by hand or it can be selected using the standard dialogue that is initiated by clicking the button ... The editor will always validate the selected variable.
  - *Track changes* – By thicking this option the size of the value change of the control variable can be set. When this size is reached or exceeded, the synchronous entry into the Collection is undertaken. The change is evaluated in absolute values. The specification of the change is done according to the definition table:
    - *Value* – Value of the size of the variable change.

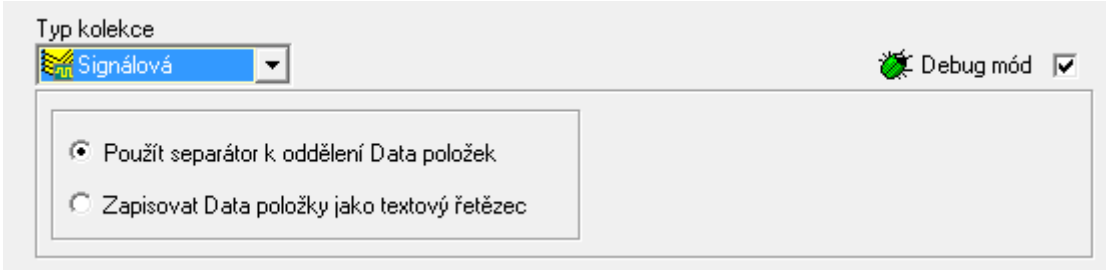
- *Debug text* – see *Debug mode* (page 11).
- *+Value* – By thicking this option the value of the control variable will be added to the debugging text.

- *Track levels* – By thicking this option up to six levels of the control variable values can be specified. When these values are reached or exceeded, the entry of the Collection will take place. Specification of the levels is determined according to the definition table:
  - ↗ - By thicking this field the increase of the variable value to the specified level will be tracked. If the *hysteresis <> 0*, the event of the entry will begin if the value of the control variable will exceed the value *(level + hysteresis)*.
  - ↘ - By thicking this field the decrease of the variable value to the specified level will be tracked. If the *hysteresis <> 0*, the event of the entry will begin if the value of the control variable will fall bellow the value *(level + hysteresis)*.
  - *Value* – The value of the variable level.
  - *Hysteresis* – The width value of the symetric hysteresis field around the set level that will increase the insensitivity of the entry of the Collection in the case when the value of the selected variable is for example loaded with the signal noise.
  - *Debug text* – see *Debug mode* (page 11).
  - *+Value* – By thicking this field the value of the control variable will be added to the debug text.



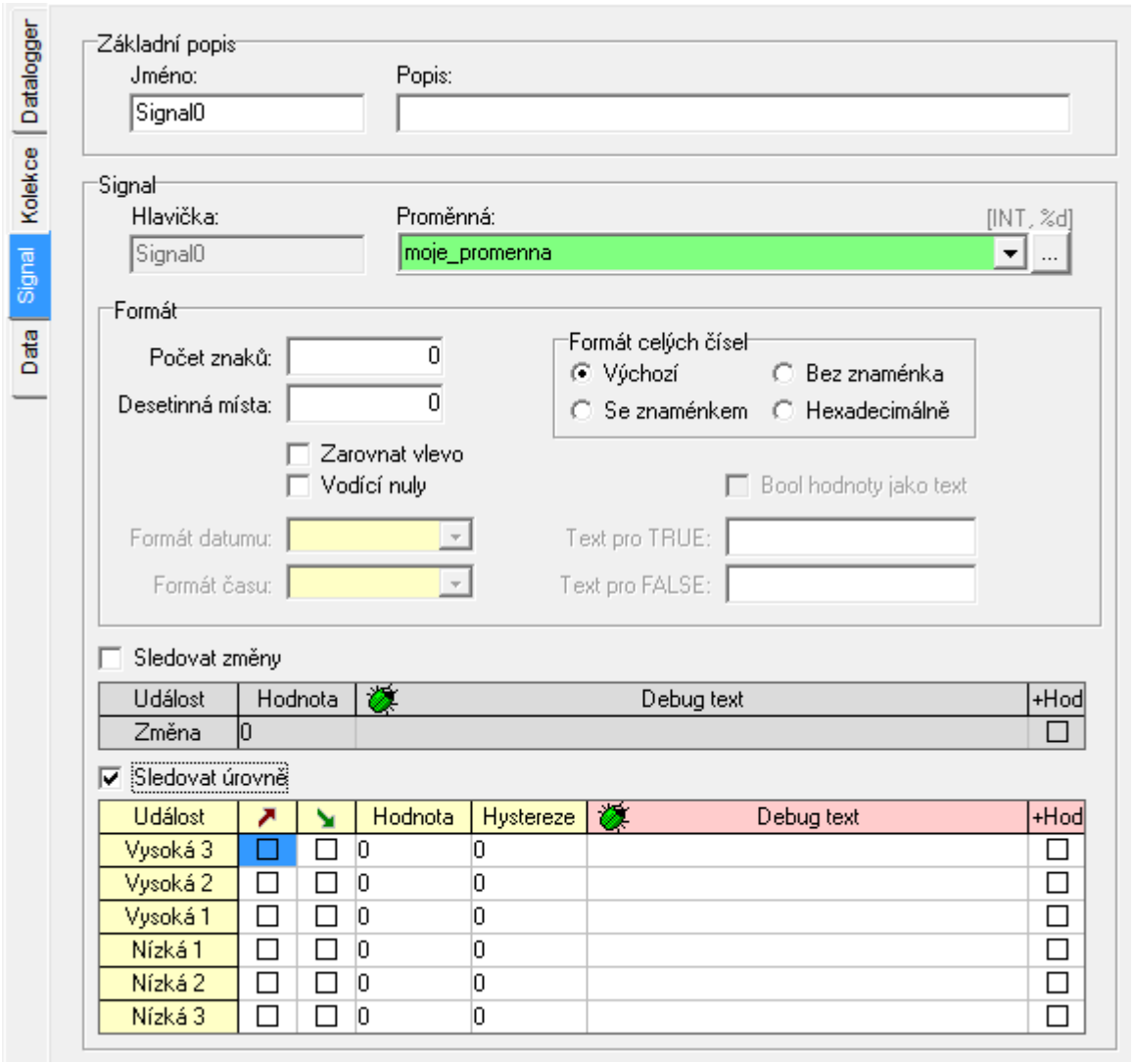
- ***Signal*** – Data entry is undertaken asynchronously on the basis of the Signal definition in the Collection. In this case the entry of all Signals is not done but each of them is entered on the basis of the asynchronous event that it itself defines. To each Signal in this Collection up to 4 items of the *Data* type can be added.

- *Use the separator to separate Data from items* – Items of *Data* type associated to the particular Signals will be, within the entry, separated by a selected sign *Column separator*.
- *Enter the Data of the item as a text string* – Items of *Data* type associated to the particular Signals will be entered as a coherent text.



Editing area of the *Signal* section

When selecting the nod of the *Signal* type within the tree structure, the relevant editing area is displayed. The type of the editor can be found on the left side (coloured).



Signal editor consists of the following parts:

- ***Basic description***
  - *Name* – Alias of the selected Signal. The lenght is limited to 15 characters.
  - *Description* – Text description of the Signal enabling the user to distinguish relevant configuration from other instances. The lenght is limited to the 59 characters.

- ***Signal***
  - *Header* – This item is not editable. It is used for identification of the name of the column

header within the Signal entry. The header name corresponds to the Signal name.

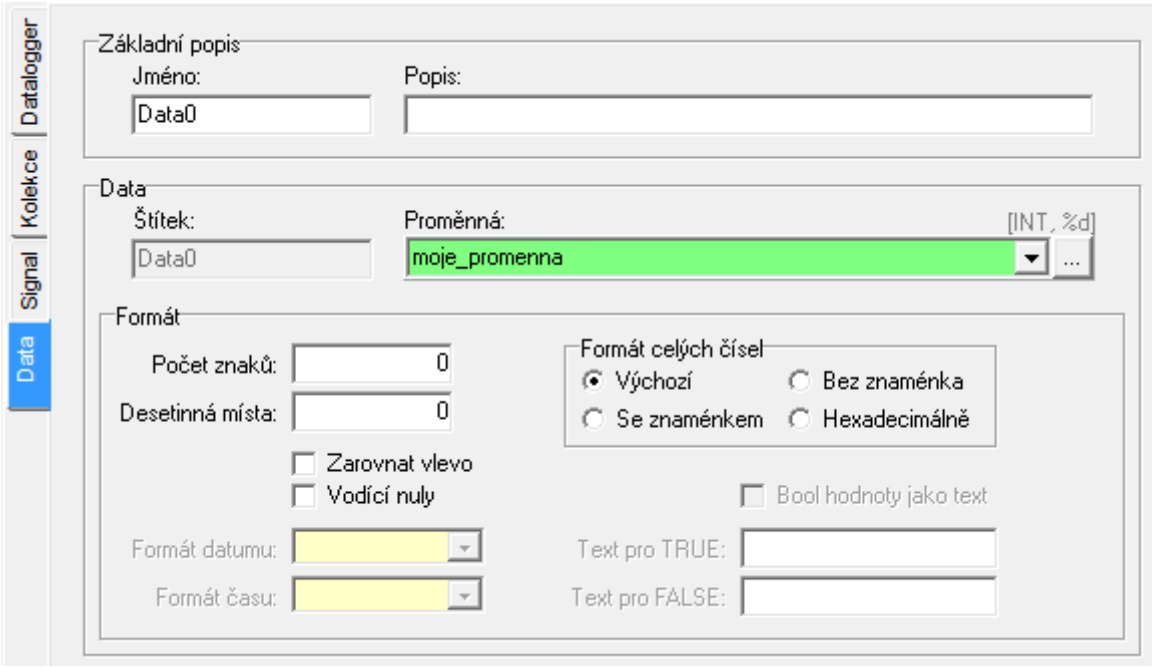
- *Variable* – Variable which value is controlled and saved into the entry. The name of the variable can be set manually or it can be selected using a standard dialogue that can be initiated by clicking the button ... . The editor will always validate the selected variable.
- *Format* – The format specification of the entered value of the selected variable within the entry.
  - *Number of characters* – The total number of characters of the numerical value.
  - *Decimal place* – The number of decimal places of the numerical value.
  - *Left alignment* – After clicking this field, the text entry of the numerical value will be aligned to the left in case it will not cover the defined number of characters. This option is exclusive to the option *Guiding zero*.
  - *Guiding zero* – By clicking this field, the text entry of the numerical value will be added with guiding zeros in case it will not cover the defined number of characters. This option is exclusive to the option *Left alignment*.
  - *Integers format* – Integers format selection (*integer*).
  - *Date format* – Format selection for *Date* type variable entry.
  - *Time format* – Format selection for *Time* type variable entry.
  - *Bool values as a text* – By ticking this field, one of the texts specified in the field *Text for TRUE* or *Text for FALSE* will be saved into the entry instead of bool value variable.
  - *Text for TRUE* – Text that will be entered into the record instead of the value 1 in case that the option *Bool values as a text is* ticked. The lenght is restricted to 15 characters.
  - *Text for FALSE* – Text that will be entered into the record instead of the value 0 in case that the option *Bool values as a text* is ticked. The lenght is restricted to 15 characters.
- *Watch changes* – By ticking this option, the size of the change of the selected variable can be set. When this size is achieved or exceeded, the asynchronous entry of the Collection will take place. The change is evaluated in absolute values. The change specification is done using defining table:
  - *Value* – The value of the change size of the variable.
  - *Debug text* – see *Debug mode* (chapter , page 11).
  - +Val – After ticking this field, the variable value will be also added to the debug text
- *Watch levels* – By ticking this option, up to six levels of the variable value can be specified. After reaching or exceeding this levels, the asynchronous entry is proceeded. The level specification is done using the defining table:
  - - By ticking this button, the variable value increase will be watched up to the specified level. If the *hysteresis <> 0*, the event of the entry will occur if the variable value exceeds the value *(level + hysteresis)*.
  - - By ticking this button, the variable value decrease will be watched up to the

specified level. If the *hysteresis <> 0*, the event of the entry will occur if the variable value drops under the value *(level + hysteresis)*.

- *Value* – The value of the variable level.
- *Hysteresis* – The value of the width of the symetric hysteresis range round the set level which will increase the non-sensitivity of the entry of the Collection in case that the value of the selected variable is, for example, shaded by noise.
- *Debug text – see Debug mode* (page 11).
- *+Val* – By ticking this field, the variable value will be added to the debug text.

Editing area of the section *Data*

When the nod of the *Data* type is selected in the tree of the description structure, the relevant editing area is displayed. výběru uzlu typu *Data* ve stromu popisné struktury se zobrazí příslušná editační plocha. That it is the Data section editor can be recognised on the left-hand side of the area thanks to the colouring of the relevant tag.



Data section editor consists of the following parts:

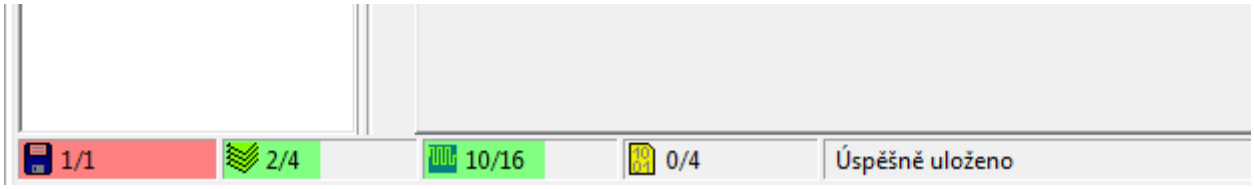
- ***Basic description***
  - *Name* – The alias of the selected Data item. The lenght is restricted to 15 characters.
  - *Description* – Text description of the relevant Data item enabling the user clear distinction of the configuration from other instances. The lenght is restricted to 59 characters.

- ***Data***

- *Label* – This item is uneditable. It is used for indication of the description label of the Data item used during the entry into the record. The label corresponds to the name of the Data item.
- *Variable* – The variable which value will be a part of additional parameters in the Signal entry. The name of the variable can be set manually or it can be selected using the standard dialogue that can be initiated by clicking the button . Editor will always validate the selected variable.
- *Format* – The format specification of the value of the selected variable in the entry.
  - *Number of characters* – The total number of characters of the numerical value.
  - *Decimal places* – The number of decimal places of the numerical value.
  - *Left alignment* – After clicking this field, the text entry of the numerical value will be aligned to the left in case it will not cover the defined number of characters. This option is exclusive to the option *Guiding zero*.
  - *Guiding zero* – By clicking this field, the text entry of the numerical value will be added with guiding zeros in case it will not cover the defined number of characters. This option is exclusive to the option *Left alignment*.
  - *Integers format* – Integers format selection (*integer*).
  - *Date format* – Format selection for *Date* type variable entry.
  - *Time format* – Format selection for Time type variable entry.
  - *Bool values as a text* – By ticking this field, one of the texts specified in the field *Text for TRUE* or *Text for FALSE* will be saved into the entry instead of bool value variable.
  - *Text for TRUE* – Text that will be entered into the record instead of the value 1 in case that the option *Bool values as a text is* ticked. The lenght is restricted to 15 characters.
  - *Text for FALSE* – Text that will be entered into the record instead of the value 0 in case that the option *Bool values as a text* is ticked. The lenght is restricted to 15 characters.

2.2.4 Status line

The status line informs the user about the filling of individual section of the Datalogger and about proceeded actions.



Indicators of filling displays actual number of particular section types from the maximum

possible number. If the indicator is red, it is not possible to add any other section instances. Contrary, the inticator is green. Maximum numbe of particular sections is, for version 1.0, as follows:

- Max 1 instance of the section *Datalogger*
- Max 4 instances of the section *Collection* in one section *Datalogger*
- Max 16 instances of the section *Signal* in one section *Collection*
- Max 4 instances of the section *Data* in one section *Signal*

Data about maximum number of the section displayed by indicators of filling can vary dependant on actualy selected item of the tree of the description structure. For example, if within one instance of the section *Datalogger* will be 4 instances of the section *Collection* and on the panel of the description structure will be selected the node of the section *Datalogger,* then the indicator of the filling for the section *Signal* will display as a maximum possible number of Signals the value 64 (4 *Collections* x 16 *Signals* = Max possible number of instances *Signal* in one instance *Datalogger*). It is recomended to clarify this fact because it can be perplexing from the beginning.

2.3 Datalogger configuration saving

Datalogger configuration is saved automatically into the same named default directory during every compilation or after closing the program Mosaic. It enables the possibility of automatical upload after repeated start of this environment. The configuration can be of course also saved during the editing by clicking the icon  on the tool bar. For saving the configuration into a different directory, the icon  is available that will initiate the standard dialogue for selection of saving path. The configuration can be saved as a file **.xml or *.dlog*. The content of both file types is similar.

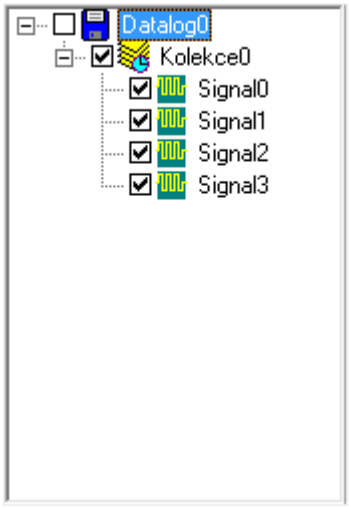
The Datalogger tool is equiped with the function of automatic saving to prevent unintentional data loss in case of incorrect operation ending. In such a case, the unsaved configuration is after repeated start of Mosaic automatically restored.

2.4 Dataloggeru addition into the project

To set up the Datalogger in the project only the Datalogger tool has to be initiated (see chapter Chyba: zdroj odkazu nenalezen) and create the first instance of the section *Datalogger*.At this moment configuration files are added into the project (*Datalogger.ST* and *Datalogger.mos*). Simultaneously, in the synchronous subdirectory *SendRoot* is after each compilation the file *DATALOG.xml* generated, containing the code for the PLC.

2.5 Datalogger removal from the project

To remove Dataloggeru from the project completely (i.e. Removal of all configuration files) it is neccessary to delete all instances in the description tree structure of the Datalogger tool (see chapter Chyba: zdroj odkazu nenalezen). To avoid unintentional data loss, it is recomended before deleting to save the whole configuration into a different, not project, directory using the option *Save as* (see chapter Chyba: zdroj odkazu nenalezen).



The second option how to prevent the Datalogger operation without the necessity to delete its configuration is its disabling by ticking the field at the relevant root nod of the description tree structure. This way all configuration files are still part of the project but the Datalogger is not active.

2.6 Errors and warnings

Translators error messages:

„Entered name 'X' is too long. Max allowed lenght is Y characters (PATH)“
The name X is longer than the max allowed text lenght of the particular item Y. PATH shows the path to the configuration section in the description tree structure that this error reffers to.

„Item name X must be set (PATH)“
The section name X was not set. Each section must have assigned its own unique name. PATH shows the path to the configuration section in the description tree structure that this error reffers to.

„ Item name X already exists (PATH)“
The section name X colides with the same name of another section of the same type. Each section must have assigned its own unique name. PATH shows the path to the configuration section in the description tree structure that this error reffers to.

„Item directory X already exists (PATH)“
The directory name X colides with the same directory name of another section of the same type. PATH shows the path to the configuration section in the description tree structure that this error reffers to.

„Column separator characters and decimal points must not be identical (PATH)“
Within *Collection* type section are similar column separator characters to the decimal points. This fact would cause error entry format. PATH shows the path to the configuration section in the description tree structure that this error reffers to.

„Invalid variable 'X' (PATH)“
The invalid system variable X was selected. PATH shows the path to the configuration section in the description tree structure that this error reffers to.

„Unauthorised data type of variable 'X'. For event control only numerical types are allowed. (PATH)
For events control the unsuitable systém variable X was selected. For such cases, it is possible to select only system variables that are numerical. PATH shows the path to the configuration section in the description tree structure that this error reffers to.

„Invalid value of the parameter 'Y' (PATH)
The invalid value was entered to the parameter Y. PATH shows the path to the configuration section in the description tree structure that this error reffers to.

„Unknown error“
Unidentifiable error.

Translators warning messages:

„Asynchronous event was not specified (PATH)“
This warning is generated for sections *Signal* that are part of Collection of the signal type in case that they do not have any defined event that would generate their asynchronous entry. PATH shows the path to the configuration section in the description tree structure that this error reffers to.

„Unknown warning“
Unidentifiable warning.

3 Datalogger in the PLC

3.1 Introduction

To ensure the correct Datalogger functioning in the PLC, it is necessary that the firmware of the central unit of the PLC contains the Datalogger support. The neccessary support is incorporated in central units of ranks K and L (all Foxtrot systems, central units CP-7004 and CP-7007 of the TC700 system) from the firmware version v6.6. Further, it is neccessary that the central unit is equipped with a memory card where CSV file will be stored.

The configuration tool for Datalogger within Mosaic generates a set of configuration files. During the compilation of the projest the file *DATALOG.XML* is created with the definition of actual configuration of the Datalogger. When setting up the Datalogger instance by the configuration tool in Mosaic environment, the file *Datalogger.ST* is automatically created with declarations of variables for possible control and operation of Datalogger from the user program and also is created the file *Datalogger.mos* with directives that are neccessary for Datalogger activation in the central unit. Files *Datalogger.ST* and *Datalogger.mos* are automatically placed into the project (refer to the list of files in the project) and their content is compiled together with other source files of the project. The *DATALOG.XML* file controls the saving of Collection of Signals into the CSV files in the central unit and is automatically sent into it during the user program loading.

Dataloggeru function principle in the central unit PLC

During transition of the PLC into the RUN mode, the central unit will upload the content of the file *DATALOG.XML*, checks its correctness and sets requirements for data saving into the CSV file. Further, it creates neccessary directories on the memory card where CSV files will be saved and for each collection sets up the first CSV file where the header is written. Afterwards, in each PLC cycle evaluates requirements for data saving and in case that the conditions defined by XML file are met, it undetakes the data storage. Simultaneously, in each PLC cycle, conditions for creation of a new CSV file are evaluated and if fulfilled, the actual file is closed and a new one is created. The last action that is undertaken by the Datalogger support in each PLC cycle is restoring of variables that are used for control of the Datalogger operation from the user program. This enables, for example, to control into which file data are stored and how many data was already stored into the file, etc.

During transition of the PLC into the HALT mode, the data saving is ended and all open CSV files are automatically closed. The same will happen if any fatal error will occur during the user program run and the program is stopped immediately.

Given that the number of entries onto the memory card is limited (mostly, it is recomended not to exceed 100 000 entries into one sector), the Datalogger in the PLC restores data in the CSV files every 10 minutes (if within those 10 mins was detected any requirement for entry). If this algorythm is not suitable, it is possible to control the frequency of date restoration from the user program. The programmer then must choose such an algorythm to ensure that the memory card will not be destroyed by the excessive number of entries during assumed period of application lifetime or Programátor pak musí zvolit takový algoritmus, aby se paměťová karta nezničila nadměrným

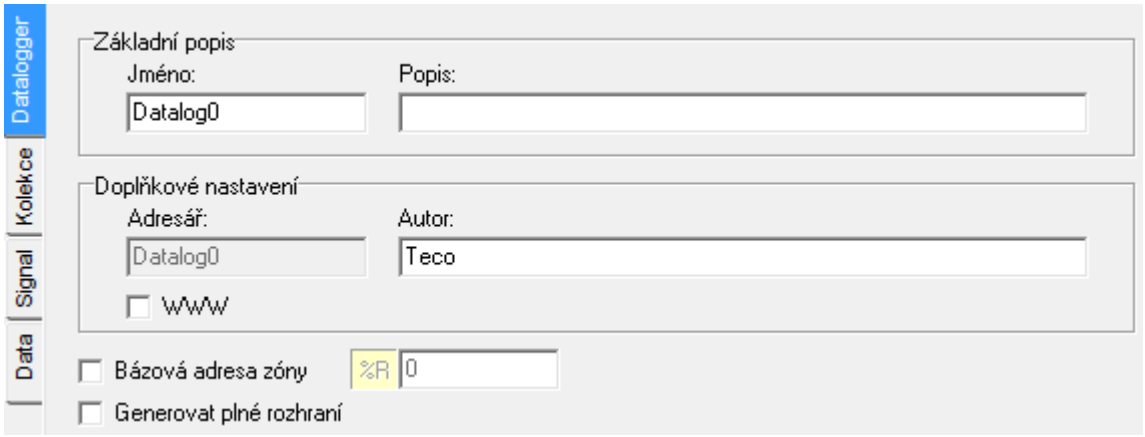
počtem zápisů během předpokládané doby životnosti aplikace nebo prescribe a mandatory periodic replacement of the card.

3.2 Datalogger control from the user program

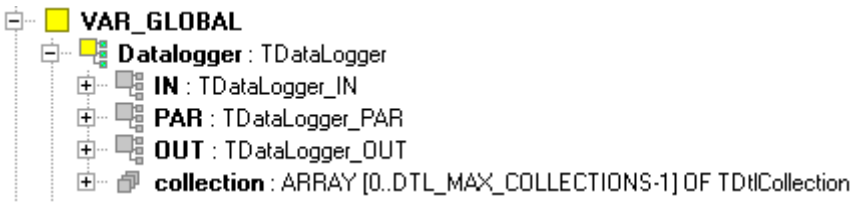
The configuration tool for the Datalogger in Mosaic generates automatically the file *Datalogger.ST* with declaration of variables for possible control and operation of the Datalogger from the user program. This file can be generated in the basic or full version. Variables that are contained in the basic version are generated always and consume cca 2kB of the memory. The full version generates based on the option *„Generate full interface*“ in the Datalogger configuration and generated variables consume cca 13kB of the memory.

3.2.1 Basic interface

The basic interface generates the configuration tool for Datalogger in case that the option *„Generate full interface*“ is not ticked in the Datalogger configuration.



As an interface the global variable named *Datalogger* is generated. It is the structure of *TdataLogger* type.



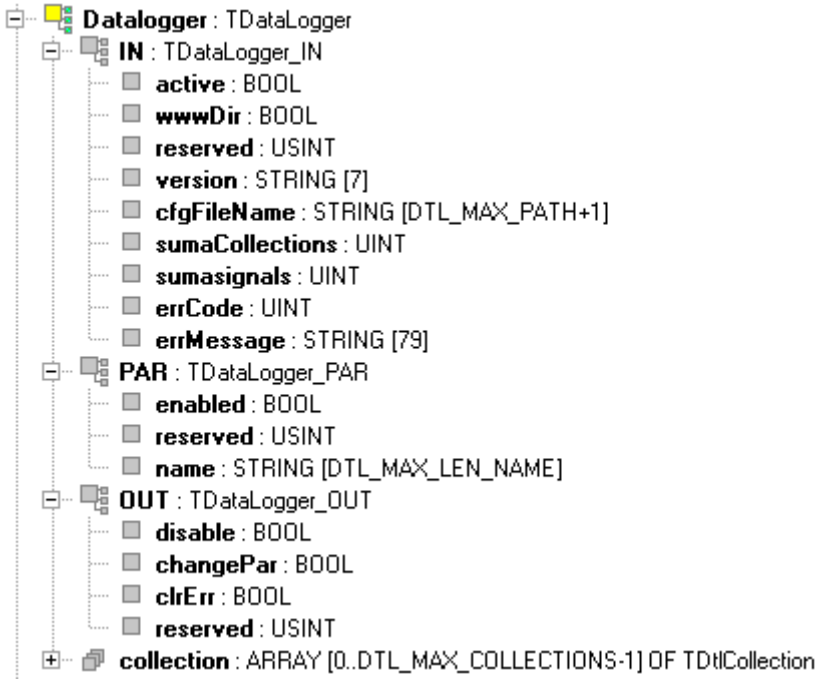
The meaning of particular items within the *Datalogger* variable is as follows:

Datalogger.IN Dataloggeru status.
Datalogger.PAR Selected parameters of the Dataloggeru that can be changed from the user program.

Datalogger.OUT Dataloggeru control.
Datalogger.collection Information about individual collections in the Dataloggeru.

Dataloggeru variables

Detailed description of the *Datalogger* structure is as follows:



Datalogger.IN

Dataloggeru status. Variables in this structure are read-only, the entry is prohibited. The significance of items of the *Datalogger.IN* variable is as follows:

Datalogger.IN.active

Datalogger is active which means that it is enabled in the configuration tool for the Datalogger and the variable *Datalogger.OUT.disable* is not set. The evaluation of conditions for saving data to CSV files in all collections that are set in the configuration is under run.

Datalogger.IN.wwwDir

Data saving of CSV files into the directory that is accessible for the web server of the pLC is enabled (see option Additional settings – WWW chapter).

Datalogger.IN.reserved

Not used yet.

Datalogger.IN.version

Firmware version of the Datalogger.

Datalogger.IN.cfgFileName

The name of the configuration XML file including the path (typically „DATALOG.XML“).

Datalogger.IN.sumaCollections
Actual number of defined collections.

Datalogger.IN.sumaSignals
Actual number of all defined signals (in all collections).

Datalogger.IN.errCode
Error code.
1 error during file entry
2 new directory can not be created
3 file can not be created
4 file can not be closed

Datalogger.IN.errMessage
Description of an error announced by the Datalogger.

Datalogger.PAR
Selected parameters of the Datalogger that can be changed from the user program. The change can be undertaken only when the Datalogger is stopped, the change will be undertaken onto the entering edge of the variable *Datalogger.OUT.changePar* (see further). The significance of these items is as follows:

Datalogger.PAR.enabled
Datalogger is enabled. Ticking box by the node Datalogger is ticked (see chapter Chyba: zdroj odkazu nenalezen).

Datalogger.PAR.reserved
Not used yet.

Datalogger.PAR.name
Datalogger name (see Basic description – name, chapter Chyba: zdroj odkazu nenalezen).

Datalogger.OUT
Datalogger control. These variables enable to stop data saving into the CSV files (for all Collections at one time), to change parameters of the Datalogger settings, etc. The significance of items of the variable *Datalogger.OUT* is as follows:

Datalogger.OUT.disable
Stop the whole Datalogger, i.e. To cease the data saving into the CSV files and close all open files. After this variable is set to TRUE, items of the structures *.PAR* can be changed. For example *Datalogger.PAR.name := 'MyLogger';* changes the name of the Datalogger. The change will be undertaken onto the entering edge of the variable *Datalogger.OUT.changePar.*

Datalogger.OUT.changePar
Change Datalogger parameters (onto entering edge of this variable).

Datalogger.OUT.clrErr

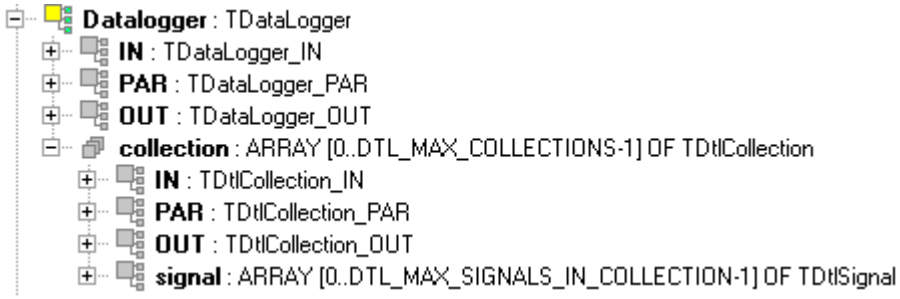
Clear the error of the Datalogger. Datalogger undertakes complete initialization (set parameters for collections saving again) and will initiate the data saving into the CSV files.

Datalogger.OUT.reserved

Not used yet.

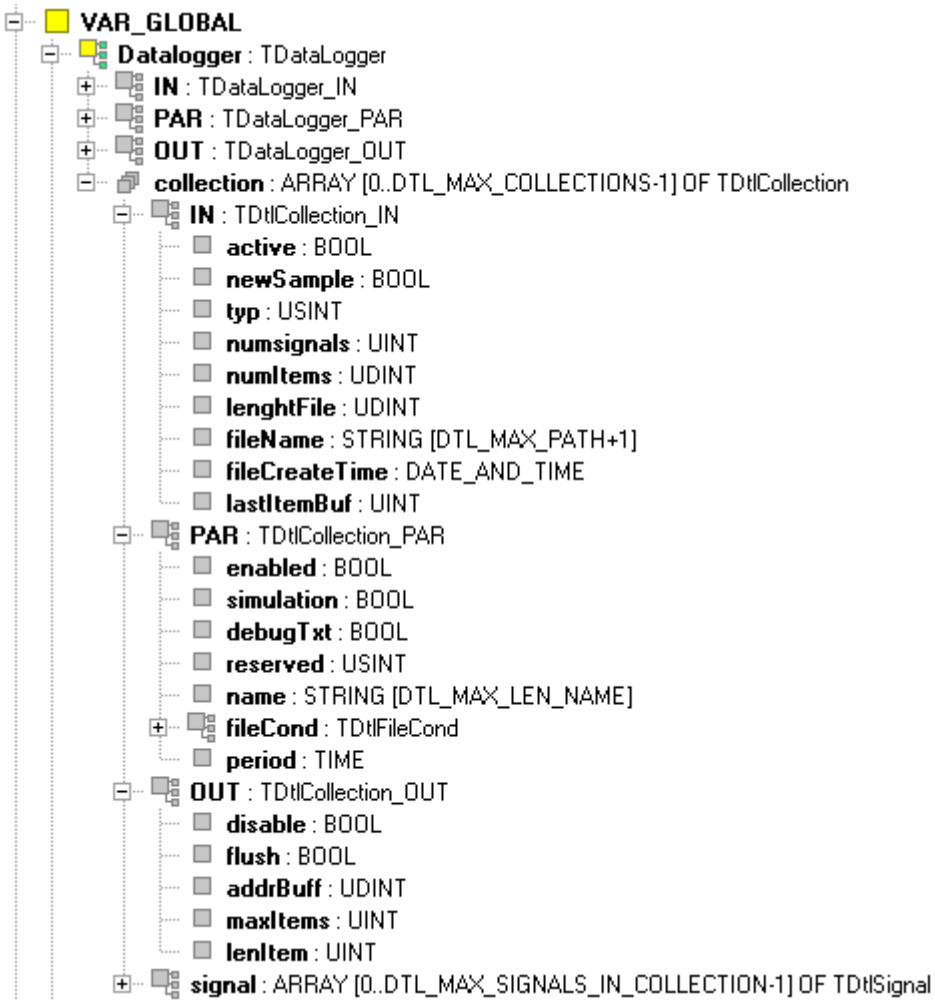
Collections variables in the Datalogger

Item *Datalogger.collection* contains information about individual Collections within the Datalogger. Regarding that the Datalogger can contain several Collections, this item is of a field type. Variable *Datalogger.collection[0]* corresponds to the first collection, variable *Datalogger.collection[1]* corresponds to the second collection, etc.



The significance of individual items of the variable *Datalogger.collection* is as follows*:*

Datalogger.collection[i].IN Status of nth Collection.
Datalogger.collection[i].PAR Selected parameters of nth Collection that can be changed from the user program.
Datalogger.collection[i].OUT Control of the nth Collection.
Datalogger.collection[i].signal Information about signals contained in the nth Collection.



Datalogger.collection[i].IN

Collection status. Variables in this structure are read only, entry into these variables is restricted. The significance of the individual items is as follows:

Datalogger.collection[i].IN.active

Collection is active, i.e. it is enabled in the configuration tool for the Datalogger (see ticking boxes located by the node Collection, chapter Chyba: zdroj odkazu nenalezen) and the variable *Datalogger.collection[i].OUT.disable* is not set. The condition evaluation is running for the data saving into the CSV file of the Collection. If the conditions are met, data are saved into the CSV file of the Collection.

Datalogger.collection[i].IN.newSample

Indication of the new data sample saving into the CSV file of the Collection (variable is set to the value TRUE for the period of one PLC cycle).

Datalogger.collection[i].IN.typ

Collection type.
0 unknown collection (*DTL_UNKNOWN_COLLECTION*)
1 periodic collection (*DTL_PERIOD_COLLECTION*)

2 event collection (*DTL_EVENT_COLLECTION*)
3 signal collection (*DTL_SIGNAL_COLLECTION*)

Datalogger.collection[i].IN.numSignals
Number of signals defined in the collection.

Datalogger.collection[i].IN.numItems
Number of sentences (lines) entered into the CSV file of the collection.

Datalogger.collection[i].IN.lengthFile
Actual lenght of the open CSV file of the collection (number of bytes).

Datalogger.collection[i].IN.fileName
Name of actually opened CSV file (incl. the path).

Datalogger.collection[i].IN.fileCreateTime
Date and time of the actually opened CSV file creation.

Datalogger.collection[i].IN.lastItemBuf
If the set address of the buffer is in the variable *Datalogger.collection[i].OUT.addrBuf* Datalogger records data into this buffer together with the record of data into the CSV file. This variable then contains intex of the last item entered in the buffer.

Datalogger.collection[i].PAR
Selected parameters of the Collection that can be changed from the user program. The change can be undertaken only when the whole Datalogger is stopped (it is not enough to stop the Collection only). The change will be undertaken onto the entering edge of the variable *Datalogger.OUT.changePar* (see above). The significance of individual items is as follows:

Datalogger.collection[i].PAR.enabled
Collection is enabled. Ticking box by the node of the Collection is ticked (see chapter Chyba: zdroj odkazu nenalezen).

Datalogger.collection[i].PAR.simulation
Simulation mode switched on (see ticking box Information about file – simulation, chapter). Datalogger works without data saving into the CSV file.

Datalogger.collection[i].PAR.debugTxt
Debug mode switched on (see ticking box Debug mode chapter). Datalogger saves into the CSV file also debug information from which the cause of the entry can be recognised (e.g.: data entry into the CSV file was caused by elapsing of the set period, by the change of the variable value, etc.).

Datalogger.collection[i].PAR.reserved
Not used yet.

Datalogger.collection[i].PAR.name
Collection name (see field Basic description – name, chapter.).

Datalogger.collection[i].PAR.fileCond.maxPeriodEnable
The closing of the CSV file is enabled after the set time, by the box Regularity, elapsed (see Conditions for new file creation – ticking box Regularity, chapter).

Datalogger.collection[i].PAR.fileCond.maxSizeEnable
The closing of the CSV file is enabled after the set size of the file, by the box Max.size, is gained (see Conditions for the new file creation – ticking box Max. Size, chapter).

Datalogger.collection[i].PAR.fileCond.maxItemsEnable
The closing of the CSV file is enabled after entry of the particular number of the lines set by the box Max. items (see Conditions for the new file creation – ticking box Max. Items, chapter).

Datalogger.collection[i].PAR.fileCond.maxTimeEnable
The closing of the CSV file is enabled after elapsing the time set by the box Max. Time (see Conditions for the new file creation – ticking box Max. Time, chapter).

Datalogger.collection[i].PAR.fileCond.maxPeriodUnits
The time unit for the field Regularity (see Conditions for the new file creation – selecting dialogue Regularity, chapter).
72 ('H') hours, 68 ('D') days, 87 ('W') weeks, 77 ('M') months

Datalogger.collection[i].PAR.fileCond.maxPeriodValue
The value of the field Regularity, the number of time units (see Conditions for the new file creation – box Regularity, chapter).

Datalogger.collection[i].PAR.fileCond.maxSizeValue
The value of the field Max. size, the number of kBytes (see Conditions for the new file creation – box Max. Size, chapter).

Datalogger.collection[i].PAR.fileCond.maxItemsValue
The value of the field Max. items, the number of lines (see Conditions for the new file creation – box Max. Items, chapter).

Datalogger.collection[i].PAR.fileCond.maxTimeValue
The value of the field Max. time, time unit (s ee Conditions for the new file creation – box Max. Time, chapter).

Datalogger.collection[i].PAR.period
Data saving period into the CSV file, valid only for periodic Collection with the permanent period (see field Period, chapter).

Datalogger.collection[i].OUT
Collection control. These variables enable to stop data saving into the CSV file, to change

setup parameters of the Collection (variable *Datalogger.collection[i].PAR...*), etc. The significance of items is as follows*:*

Datalogger.collection[i].OUT.disable

Stop Collection, i. e. To cease the data saving into the CSV file of the Collection and to close the opened file of the Collection. As long as this variable is TRUE, the operation of this Collection is stopped. This status is indicated by the variable *Datalogger.collection[i].IN.active* = FALSE. By the change of the value *Datalogger.collection[i].OUT.disable* to FALSE, the Collection will start operation again, new CSV file will open and data saving will start. This variable enables to activate and stop the data saving of the particular Collection from the PLC user program. Collection parameters are not influenced by Collection cessation or initiatization.

Datalogger.collection[i].OUT.flush

One-time saving of the elaborated CSV file. This action will reset internal timer in the Datalogger that undetakes automatic file saving once in 10 minutes.

Datalogger.collection[i].OUT.addrBuf

If we want the Datalogger to copy the data saved into the CSV file also into the program variable, it is necessary to fulfill also the address of this variable (buffer) into the *Datalogger.collection[i].OUT.addrBuf*. Regarding that the CSV file contains texts, this variable can be a field of strings. The first string in the field will contain the CSV file header, every other string of this field will then contain one line of the CSV file. Lines will be, into this field, saved gradually in the way they are saved into the file. The variable *Datalogger.collection[i].IN.lastItemBuf* then contain the index of the field where the last line was saved. It is presumed that the field index starts from 0. If the whole field is filled up and a requirement for another line saving will appear, then, all lines saved in the field are moved so, that the oldest line will be removed and then the new line is entered into the end of the field. So, for example, if the field consists of 11 strings, then the string with the index 0 always contain the CSV file header and another 10 strings contains the last 10 lines saved in the file so, that the line saved as a last one has the highest index (i. e. 10). The essential for the right functioning is the filling of the number of the strings in the field of the variable *Datalogger.collection[i].OUT.maxItems* and the lenght of one string into the variable *Datalogger.collection[i].OUT.lenItem.*

Datalogger.collection[i].OUT.maxItems

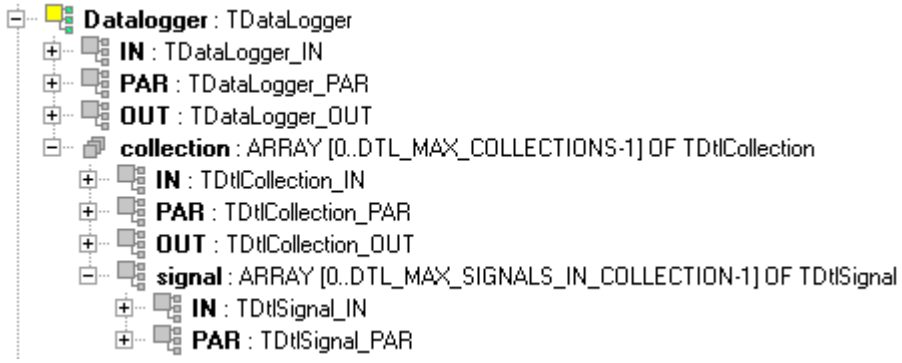
Number of items (strings) in the buffer where the Datalogger will paralelly save similar data as it saves into the CSV file. See the description of the variable *Datalogger.collection[i].OUT.addrBuf.*

Datalogger.collection[i].OUT.lenItem

The lenght of one item (string) in the buffer where the Datalogger will paralelly save similar data as it saves into the CSV file. See the description of the variable *Datalogger.collection[i].OUT.addrBuf.*

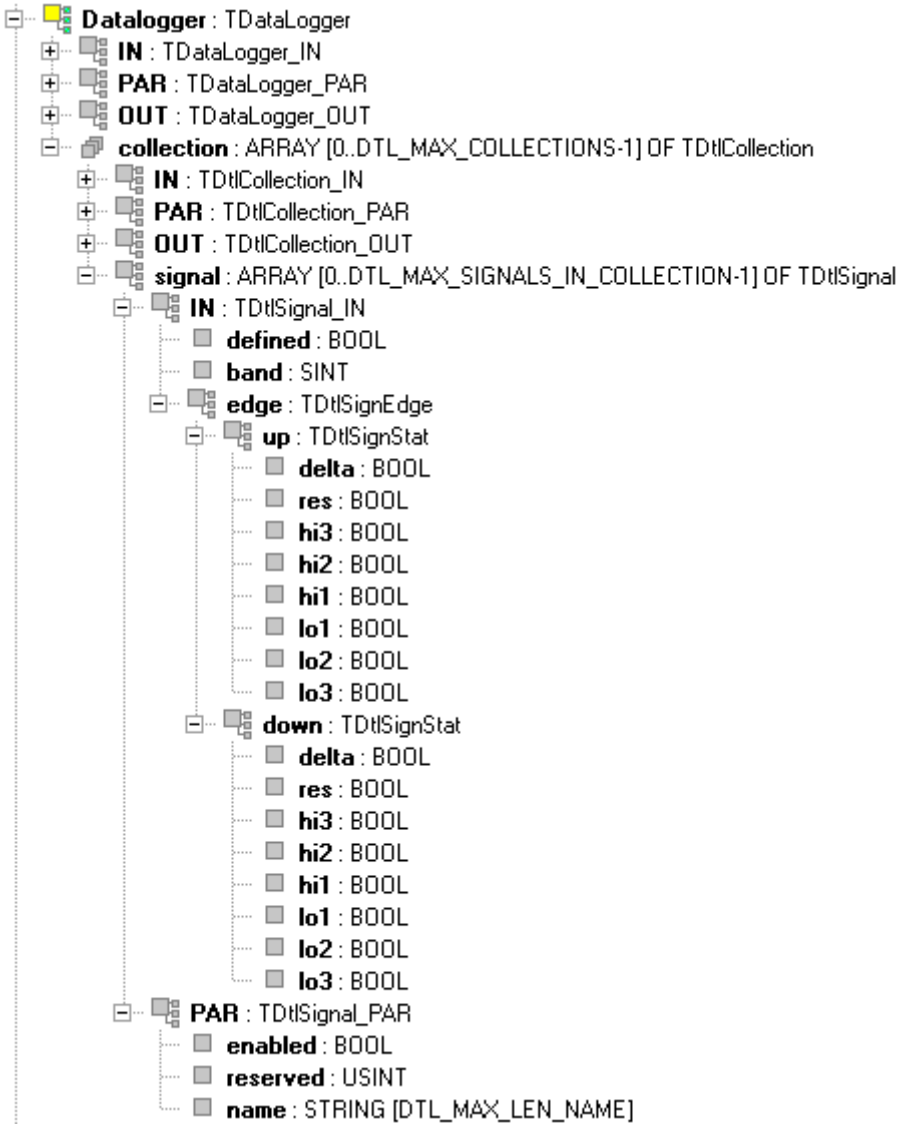
Signals variables in Collections

Information about signals in Collection is contained in the variable *Datalogger.collection[i].signal*.



The significance of individual items of the variable *Datalogger.collection[i].signal* is as follows:

Datalogger.collection[i].signal[j] .IN	|Signal statuses in the nth Collection.
Datalogger.collection[i].signal[j] .PAR	Selected parameters of the signals in the nth Collection



Datalogger.collection[i].signal[j] .IN

Status of the xth signal in the nth Collection. Variables in this structure are read only, the entry is restricted. The significance of the individual items is as follows:

Datalogger.collection[i].signal[j] .IN.defined

Signal is defined. Ticking box by the relevant node Signal is ticked (see chapter Chyba: zdroj odkazu nenalezen).

Datalogger.collection[i].signal [j].IN.band

The range where the signal value is located.

Datalogger.collection[i].signal [j].IN.edge

Events edges on the signal.

Datalogger.collection[i].signal [j].IN.edge.up

Events edges on the signal if the signal value is increasing.

Datalogger.collection[i].signal [j].IN.edge.down

Events edges on the signal if the signal value is decreasing.

Datalogger.collection[i].signal [j].PAR

Selected Signal parameters that can be changed from the user program. The change can be undertaken only when the whole Datalogger is stopped (it is not enough to stop the Collection only). The change will be undertaken onto the entering edge of the variable *Datalogger.OUT.changePar* (see above). The significance of individual items is as follows:

Datalogger.collection[i].signal [j].PAR.enabled

Signal is enabled within the configuration. The ticking box by the relevant node of the Signal si ticked (see chapter Chyba: zdroj odkazu nenalezen).

Datalogger.collection[i].signal [j].PAR.reserved

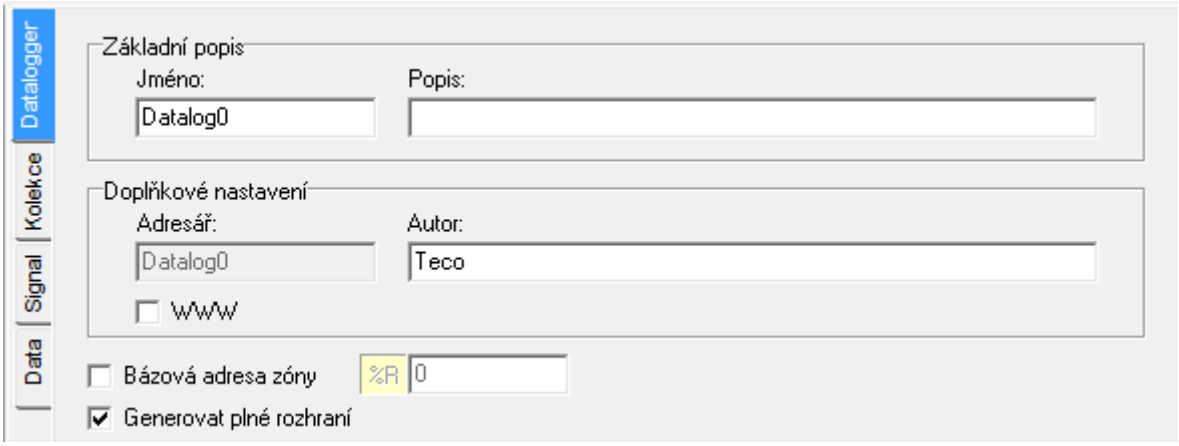
Not used yet.

Datalogger.collection[i].signal [j].PAR.name

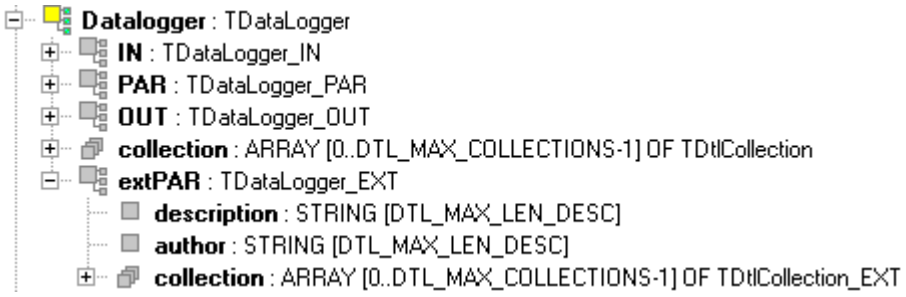
Signal name. Content of the field Basic description – name, chapter).

3.2.2 Full interface

The full interface generates the configuration tool for the Datalogger in case that the option *„Generate full interface“* is ticked within the Datalogger configuration (see chapter Chyba: zdroj odkazu nenalezen). This makes sense in case that it will be neccessary to change the detailed features of Collections and Signals in the Datalogger, e. g.: from the operator´s panel (i. e. From the PLC program). In another words, it is not enough to configure the Datalogger in the Mosaic environment but it is also neccessary to change some parameters for the Datalogger directly in the application.



As an interface the global variable is generated with the name *Datalogger* which is the structure of the type *TDataLogger*. This structure contains, in case of the full interface, all items that are in the basic interface. Moreover, there is the structure *extPAR* which contains other parameters of the Collection and Signals, in case that these parameters must be changed from the PLC user program (e. g.: names of individual signals, etc.)



The description of items *Datalogger.IN, Datalogger.PAR, Datalogger.OUT* and *Datalogger.collection* see chapter Chyba: zdroj odkazu nenalezen. The significance of individual items in the variable *Datalogger.extPAR* is as follows:

Datalogger.extPAR.description field Description in the Datalogger configuration (see option Basic description – Description, chapter).
Datalogger.extPAR.author field Author in the Datalogger configuration (see option Additional settings – Author, chapter).
Datalogger.extPAR.collection extended parameters of Collections

Collections variables in the Datalogger

Datalogger.extPAR.collection

Item *Datalogger.extPAR.collection* contains additional information about individual Collections in the Datalogger. Regarding that the Datalogger can contain several Collection is this item of the field type. The variable *Datalogger.extPAR.collection[0]* corresponds to the first collection in the Datalogger, the variable *Datalogger.extPAR.collection[1]* corresponds to the second collection, etc. All parameters contained in the variable *Datalogger.extPAR.collection* can be changed from the user program. The change can be undertaken only when the whole Datalogger is stopped (the variable *Datalogger.OUT.disable = TRUE*). The change will be undertaken onto the entering edge of the variable *Datalogger.OUT.changePar* (see chapter Chyba: zdroj odkazu nenalezen). The significance of the individual items is as follows*:*

Datalogger.extPAR.collection[i].description

Collection description (see field Basic description – Description, chapter).

Datalogger.extPAR.collection[i].separator

Column devider in the CSV file (see field Information about file – Column devider, chapter). This variable can contain one of the following strings: ';' or ',' or 'TAB'.

Datalogger.extPAR.collection[i].decimalPoint

The devider of the decimal part of the real numbers in the CSV file (see field Information

about file – Decimal devider, chapter). In case that a dot is used as a devider of the decimal part of the number (e. g.: 12.5), this variable has the value 0, in case that a comma is used (e.g. 12,5), this variable has the value 1. Other values are restricted. Table calculators designed for the czech language use usually a comma, the same programs for english version use a dot.

Datalogger.extPAR.collection[i].dirForDays

Set the directory for a day (see field Information about file – Set directories for days, chapter). Value 0 in this variable means that directories for days will not be created, in another words, all CSV files from the same month will be in the same directory. Value 1 means that for each day will be created separate directory, i. e. CSV files from the same month will be in different directories according to the day in which the file was created. The strategy of creation and giving names to directories, see chapter Chyba: zdroj odkazu nenalezen.

Datalogger.extPAR.collection[i].dateEnable

Enable the entry of the column Date into the CSV file (see ticking box General signals – Date, chapter).

Datalogger.extPAR.collection[i].dateHeader

Name of the column Date in the CSV file (see ticking box General signals – Date Header, chapter).

Datalogger.extPAR.collection[i].dateFormat

Format of the entry of the value in the column Date in the CSV file (see field General signals – Date Format, chapter). This variable contains format string for conversions DATE_TO_STRING, for details see Library ToStringLib chapter 1.4.

Datalogger.extPAR.collection[i].timeEnable

Enable the entry of the column Time into the CSV file (see ticking box General signals – Time, chapter).

Datalogger.extPAR.collection[i].timeHeader

Time column name in the CSV file (see field General signals – Time Header, chapter).

Datalogger.extPAR.collection[i].timeFormat

Format of the entry of the value in the column Time in the CSV file (see field General signals – Time Format, chapter). This variable contains format strings for conversions TIME_TO_STRING, for details see Library ToStringLib chapter 1.5.

Datalogger.extPAR.collection[i].periodControl

Enabled periodic control of the Collection with the controlled period (see selective field Collection type – Controlled period, chapter).

Datalogger.extPAR.collection[i].periodReset

Enabled asynchronous reset of the period (is meaningfull within periodic Collection only) (see ticking box Collection type – Asynchronous reset, chapter).

Datalogger.extPAR.collection[i].periodOff

If the periodic Collection with the controlled period is enabled, then this variable contains the size of the period for the status when the control variable is = 0 (see selective field Collection type – Period for variable == 0, chapter).

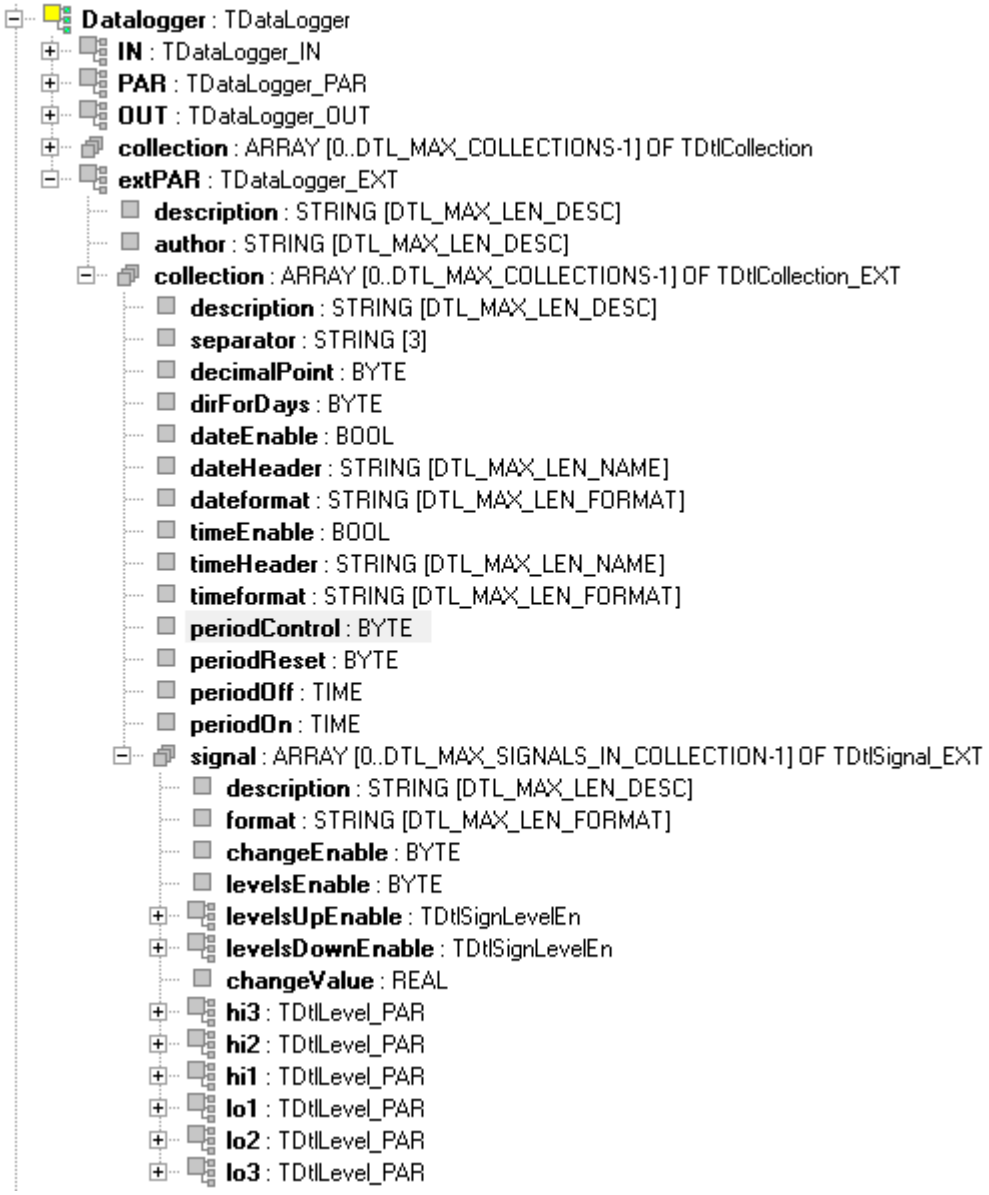
Datalogger.extPAR.collection[i].periodOn

If the periodic Collection with the controlled period is enabled, then this variable contains the size of the period for the status when the control variable is <> 0 (see selective field Collection type – Period for variable <> 0, chapter).

Datalogger.extPAR.collection[i].signal

Signal parameters in the Collection, see the following picture and further description.

Signal variables in Collections



Datalogger.extPAR.collection[i].signal

Item *Datalogger.extPAR.collection[i].signal* contains additional information about individual signals within the Collection. Regarding that the Collection can contain several Signals, this item is of a field type. The variable *Datalogger.extPAR.collection[0].signal[0]* corresponds to the first signal in the Collection, *Datalogger.extPAR.collection[0].signal[1]* corresponds to the second Signal, etc. All parameters contained in the variable *Datalogger.extPAR.collection[i].signal* can be changed from the user program. The change can be undertaken only when the whole Datalogger is stopped (variable *Datalogger.OUT.disable = TRUE*). The change will be undertaken onto the entering edge of the variable *Datalogger.OUT.changePar* (see chapter Chyba: zdroj odkazu nenalezen). The significance of the individual items is as follows:

Datalogger.extPAR.collection[i].signal[j].description

Signal description. The content of the field Basic description – Description (see chaptre).

Datalogger.extPAR.collection[i].signal[j].format

Format of the entry of the Signal value into the CSV file. The content of the fields Format (see chapter). This variable contains a format string that is dependant on the data type of the variable that is saved into the CSV file. Forms of format strings, see Library ToStringLib chapter 1.

Datalogger.extPAR.collection[i].signal[j].changeEnable

The data saving into the CSV file on the basis of the change of the signal value is enabled. The content of the ticking box Follow changes (see chapter).

Datalogger.extPAR.collection[i].signal[j].levelsEnable

The control of signal levels enabled. The content of the ticking box Follow levels (see chapter).

Datalogger.extPAR.collection[i].signal[j].levelsUpEnable

Enabling levels that will be compared with the signal value. If the level is reached, the data will be saved into the CSV file. The content of the ticking box Follow levels – Event UP (see chapter).

Datalogger.extPAR.collection[i].signal[j].levelsDownEnable

Enabling levels that will be compared with the signal value. If the level fall down, the data will be saved into the CSV file. The content of the ticking box Follow levels – Event DOWN (see chapter).

Datalogger.extPAR.collection[i].signal[j].changeValue

The value of the followed change. If the signal is changed within the set value, the signal values will be saved into the CSV file. The content of the field Follow changes – Value (see chapter).

Datalogger.extPAR.collection[i].signal[j].hi3
Datalogger.extPAR.collection[i].signal[j].hi2

Datalogger.extPAR.collection[i].signal[j].hi1
Datalogger.extPAR.collection[i].signal[j].lo1
Datalogger.extPAR.collection[i].signal[j].lo2
Datalogger.extPAR.collection[i].signal[j].lo3

The value of the followed level. If the signal reaches this level, the signal values will be saved into the CSV file. The content of the field Follow levels – Value (see chapter). Variables *Datalogger.extPAR.collection[i].signal[j].levelsUpEnable* and *Datalogger.extPAR.collection[i].signal[j].levelsDownEnable* determine the evaluation trend in which the level is reached.

3.3 CSV file saved by the Datalogger

Datalogger saves measured Signal values into the CSV file. The first line of the CSV file contains information for possible automatic file processing. On the second line is the header that contains names of the saved signals. From the third line up to the end of the file measured Signal values are saved.

Example :

```
Tecomat 100 CP1004K V6.6 DataLogger v1.0 (2011-07-18-11:19:46) : Datalog0 /
Kolekce0 ! Type : PERIODIC | DATE : %TYYYY-MM-DD | TIME : %Thh:mm:ss | REAL :
%8.1f | REAL : %6.1f | ,;
DATE;TIME;presure;temperature
2011-07-18;11:19:46;  1007,9;  22,0;
2011-07-18;11:20:46;  1007,9;  22,1;
2011-07-18;11:21:46;  1007,9;  22,2;
```

The first line in the CSV file contains information from which it is possible to determine: which system generated the CSV file (incl. Systém versions), when the file was created, what is the name of the Datalogger and Collection to which the file is associated and lastly, which Signals it contains (signal names and values saving format). The significance of information on the first line is as follows:

Tecomat 100 CP1004K V6.6	type of control system Foxtrot and its version
DataLogger v1.0 (2011-07-18-11:19:46)	Datalogger version (date and time of CSV creation)
:	divider of further information
Datalog0 / Kolekce0	Datalogger name / Collection name
!	divider of further information
Type : PERIODIC |	Collection type
DATE : %TYYYY-MM-DD |	name : format for the first item saved
TIME : %Thh:mm:ss |	name : format for the second item saved
REAL : %8.1f |	name : format for the third item saved
REAL : %6.1f |	name : format for the fourth item saved
,;	divider used for the decimal part (comma) and used divider for columns (semicolon)

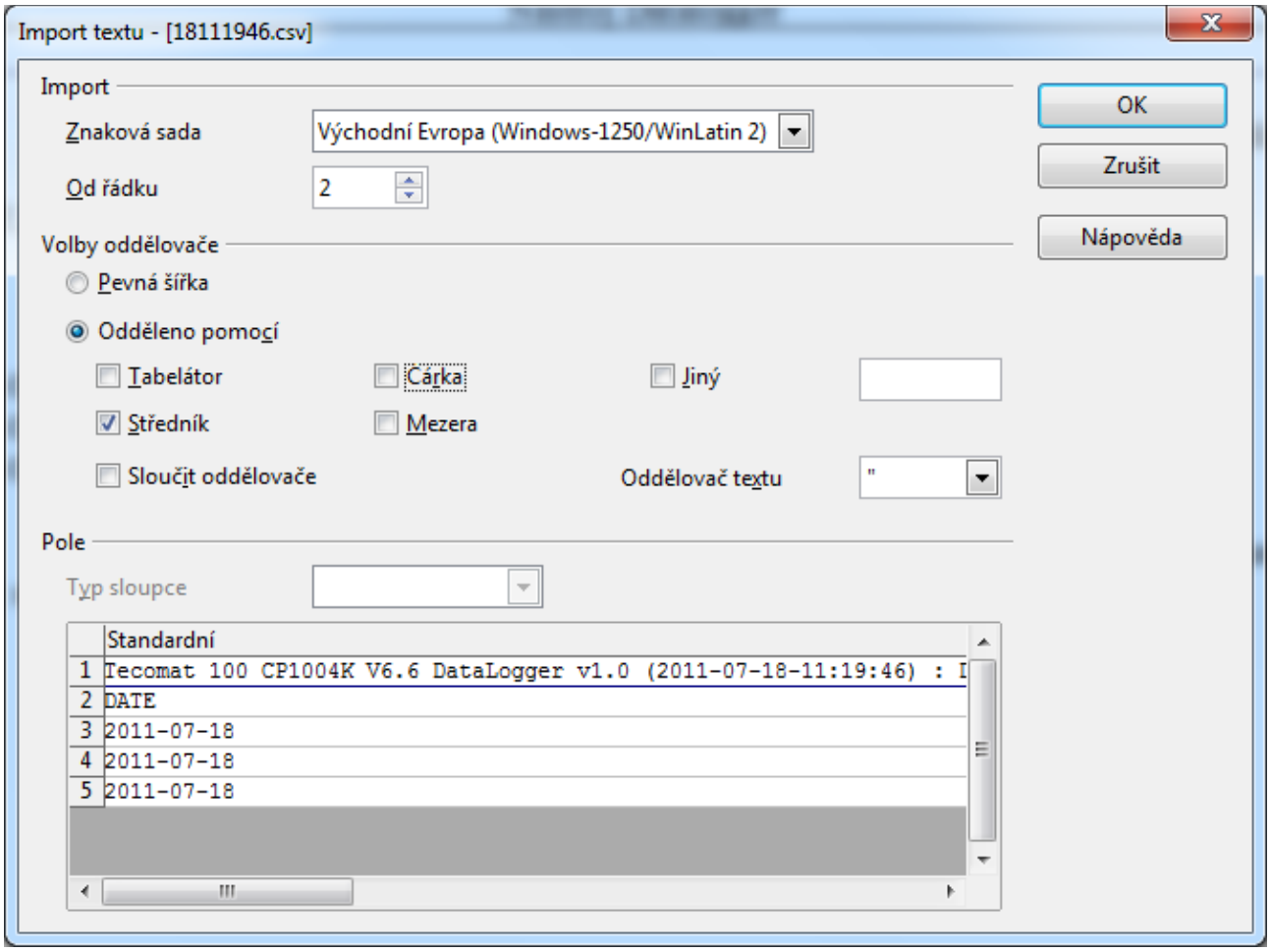
The second line of the CSV file contains the header where names of all saved signals are:

DATE;TIME;presure;temperature

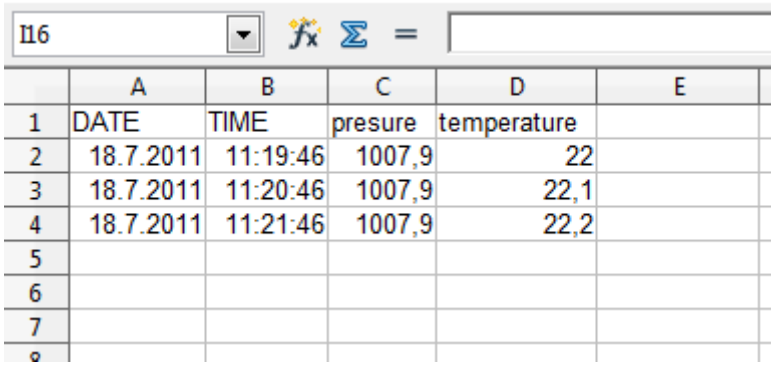
Third and other lines in the file contains signal values:

2011-07-18;11:19:46; 1007,9; 22,0;
2011-07-18;11:20:46; 1007,9; 22,1;
2011-07-18;11:21:46; 1007,9; 22,2;

If the CSV file will be uploaded into the tab-calculator, it is reccomended to opt out the first line with information for automatic processing. The example of settings of the dialogue for import of the CSV file into the program Calc (part of the office package Open Office) is on the following picture.



The result of the import of the CSV file into the program Calc is the following table of values:



	A	B	C	D	E
1	DATE	TIME	presure	temperature	
2	18.7.2011	11:19:46	1007,9	22	
3	18.7.2011	11:20:46	1007,9	22,1	
4	18.7.2011	11:21:46	1007,9	22,2	
5					
6					
7					

4 Examples

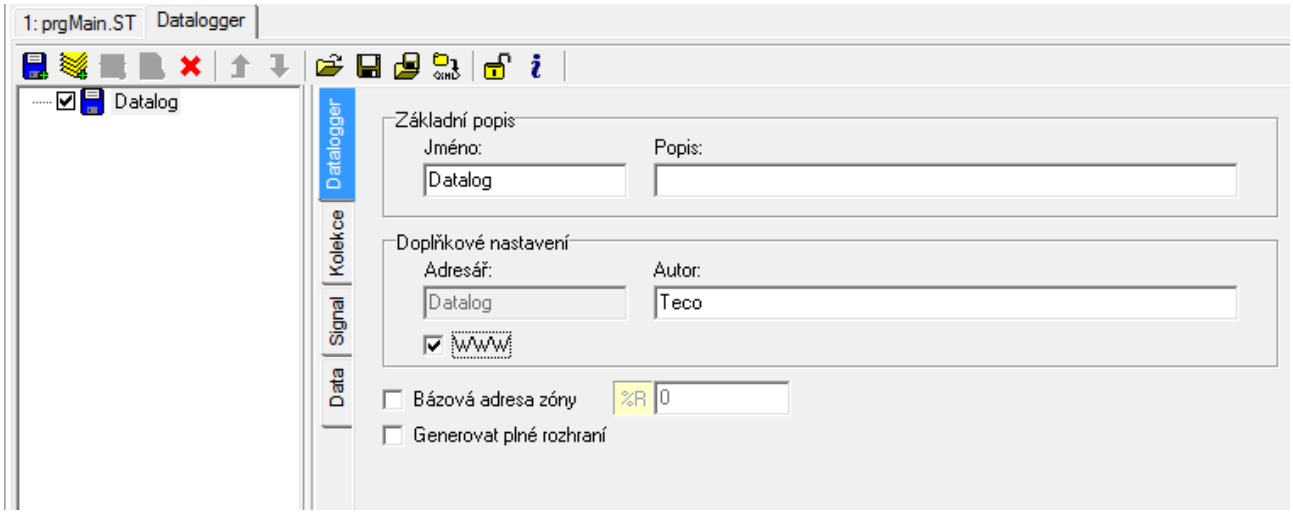
4.1 Periodically saved collection

Lets suppose that the user program for PLC contains two global variables *presure* and *temperature* which values we want to save each minute into the CSV file. For each day there will be one CSV file set up. These files can be downloaded via the web browser.

Variables will be in the program declared, for example, as follows:

```
VAR_GLOBAL
  presure     : REAL;
  temperature : REAL;
END_VAR
```

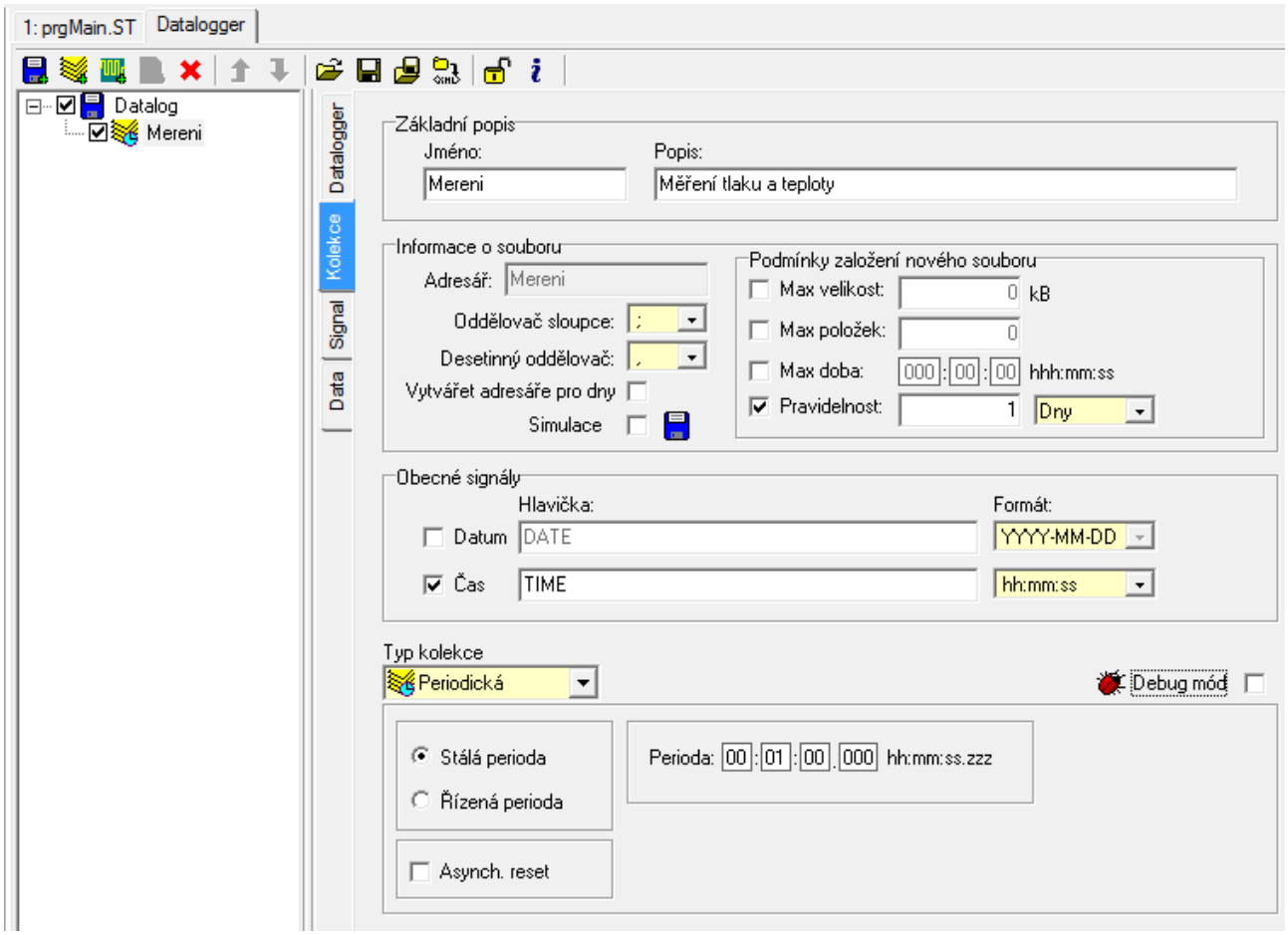
We launch the tool for the Datalogger configuration in the Mosaic environment and we will set up the Datalogger section (pressing the icon). We fill in the *Basic description*, i.e. items *Name* (Datalog), *Description* and *Author*. All CSV files created by Datalogger will be saved into the directory structure the top of which will be the directory Datalog. We will tick the field *WWW*, which cause that the directory Datalog will be set up in such a path that is accesible from the web server of the PLC. The ticking box *Generate the full interface* stay unticked (configuration tool for the Datalogger will generate only basic interface).



Further, we add Collection into the Datalogger (pressing the icon). We fill in the *Name* of the collection (Measurement) and its description (pressure and temperature measurement). The ticking box *Create directories for days* will stay unticked (i. e. That CSV files from one calendar month will be saved in the similar directory). We tick the field *Regularity* and set 1 day. It means that data will be saved during the whole day into one file and for each day there will be a separate directory created. Then we set *General signals Date and Time.* Regarding that data form one day will be saved in one file, it is not neccessary to save into the CSV file date for each sample of data

saved. The field *Date* will then stay unticked. Conversely, the field *Time* will be ticked so, that each data sample has a time entry. In the field *Collection type* the *Periodic* will be selected and the period will be set to 1 minute. The field *Debug mode* will stay unticked (debugging information will not be saved into the CSV file).

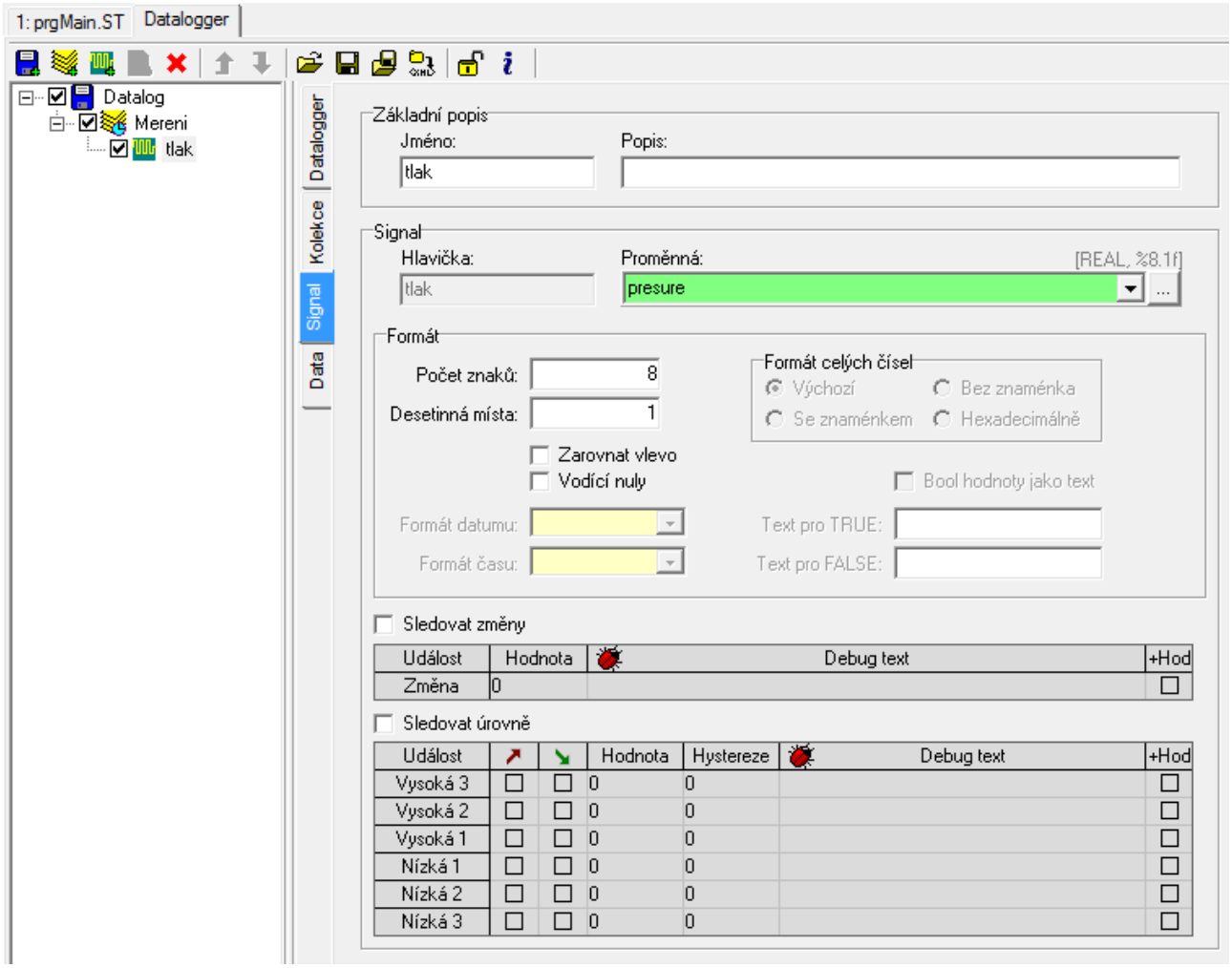
The Collection settings will then appears as follows:

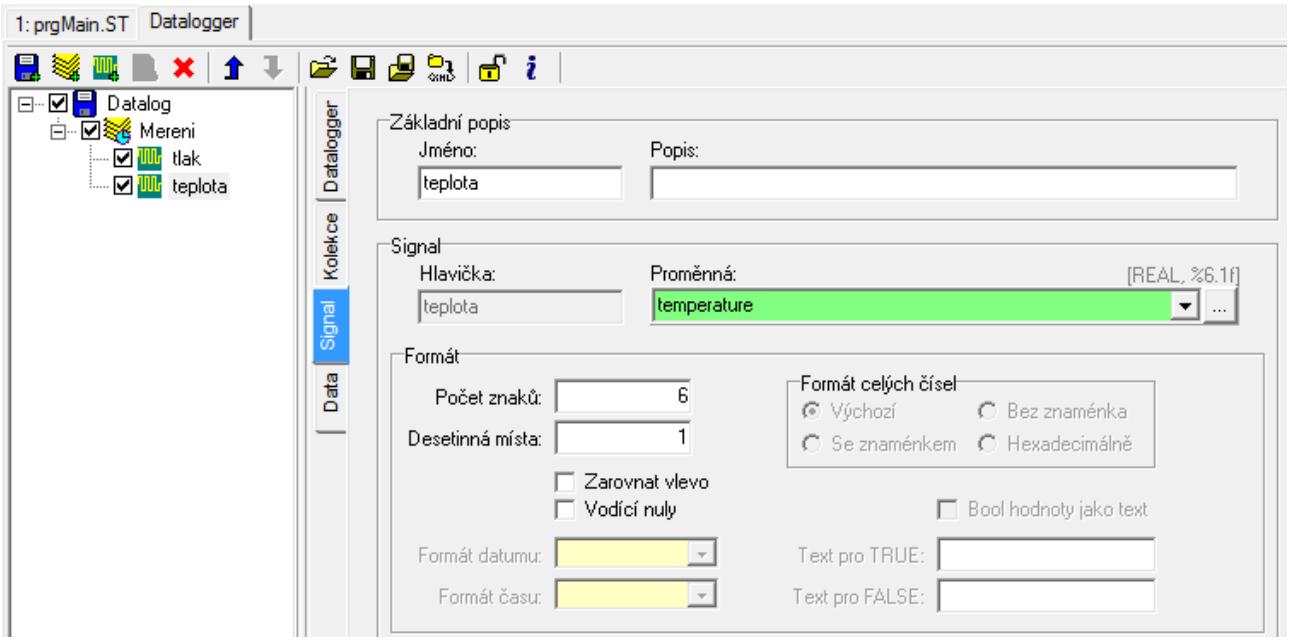


Now we can add Signals into the Measurement Collection (pressing the icon). Lets say that the first saved variable will be the variable *presure*. In the field *Pressure* we will select the system variable *presure* and will fill in the name in the *Basic description* (e.g. pressure). Further, we set the *Format* of saving, e.g. *Number of characters 8 and Decimal places 1* (variable *presure* lenght will be in the CSV file 8 characters in total from which there will be 1 decimal place). Other fields will remain unticked, or rather, blank.

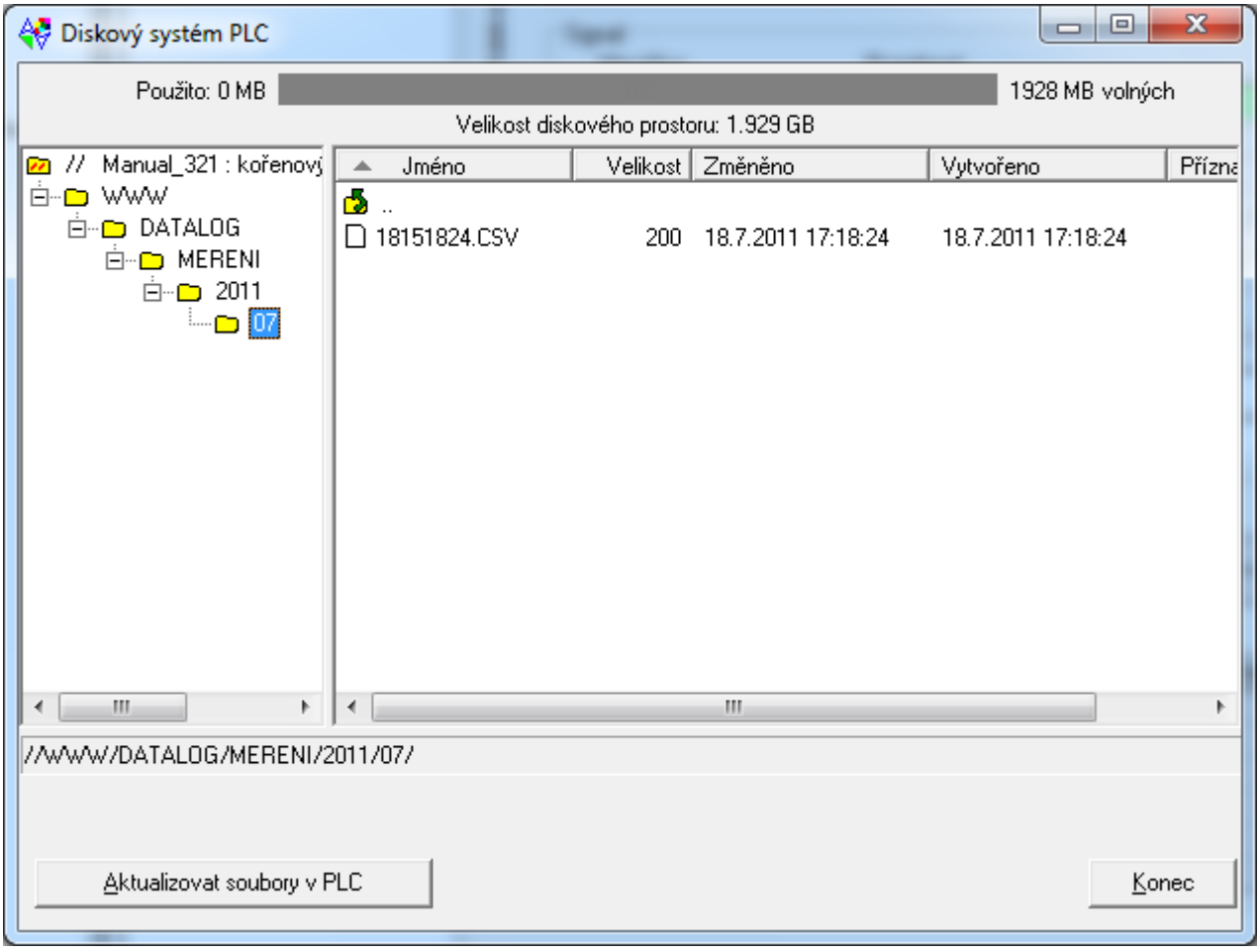
In the same way the next Signal can be added, this time for variable *temperature* that will be named temperature. Here we select format of 6 characters with one decimal place. Other fields will remain blank.

Signals settings dialogues will thus appear as follows:

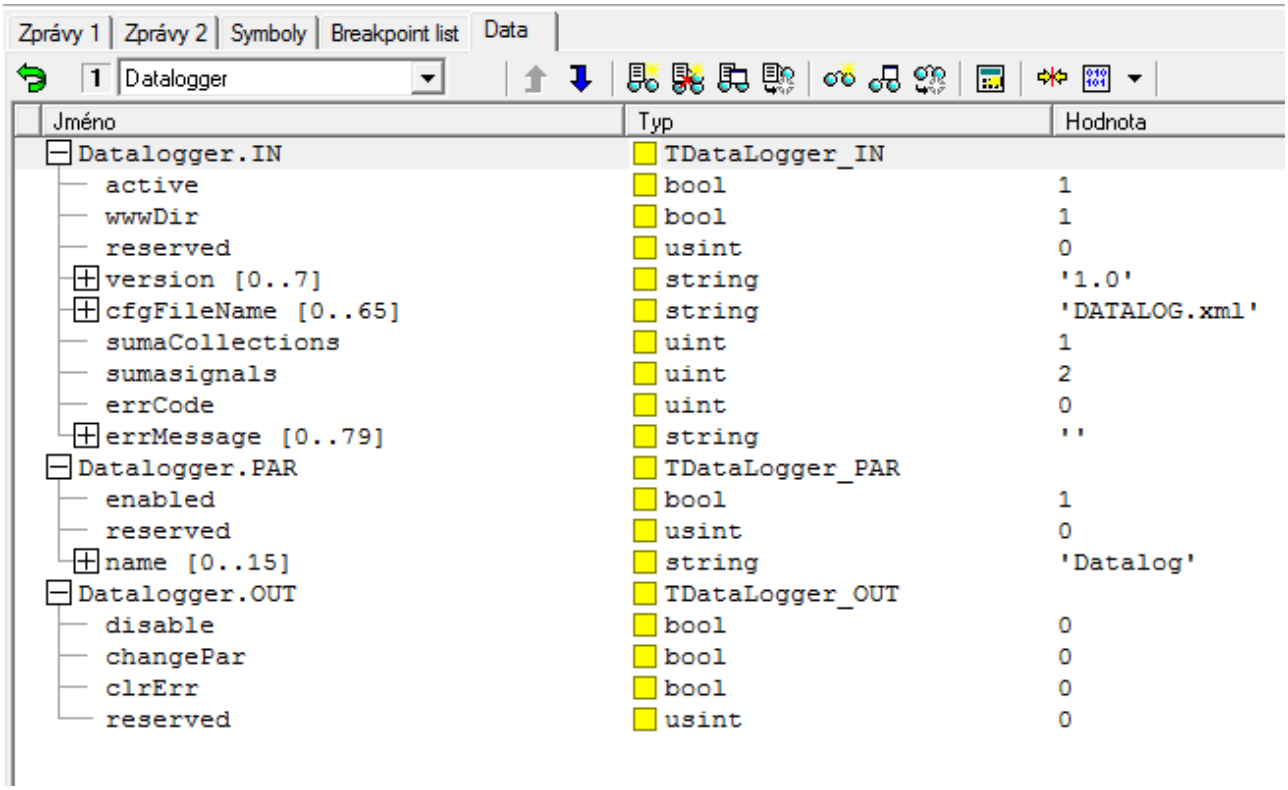

1: prgMain.ST
Datalogger
Datalog
Mereni
tlak
Datalogger
Kolekce
Signal
Data
Základní popis
Jméno:
tlak
Popis:
Signal
Hlavička:
tlak
Proměnná:
[REAL, %8.1f]
presure
Formát
Počet znaků:
8
Desetinná místa:
1
Formát celých čísel
Výchozí
Bez znaménka
Se znaménkem
Hexadecimálně
Zarovnat vlevo
Vodící nuly
Bool hodnoty jako text
Formát datumu:
Formát času:
Text pro TRUE:
Text pro FALSE:
Sledovat změny
Událost
Hodnota
Debug text
+Hod
Změna
0
Sledovat úrovně
Událost
Hodnota
Hystereze
Debug text
+Hod
Vysoká 3
Vysoká 2
Vysoká 1
Nízká 1
Nízká 2
Nízká 3


1: prgMain.ST
Datalogger
Datalog
Mereni
tlak
teplota
Datalogger
Kolekce
Signal
Data
Základní popis
Jméno:
teplota
Popis:
Signal
Hlavička:
teplota
Proměnná:
[REAL, %6.1f]
temperature
Formát
Počet znaků:
6
Desetinná místa:
1
Formát celých čísel
Výchozí
Bez znaménka
Se znaménkem
Hexadecimálně
Zarovnat vlevo
Vodící nuly
Bool hodnoty jako text
Formát datumu:
Formát času:
Text pro TRUE:
Text pro FALSE:

Now, the Datalogger configuration is complete. Remaining is to compile the project (F9) and upload the compiled code into the PLC and switch the PLC into the RUN mode (*CTRL+F9*). Consequently, the Datalogger in the PLC is also switched on and each minute it saves the status of variables *presure* and *temperature* into the CSV file. The following picture shows how the actual directory structure, into which the CSV file is saved, looks like. All files produced by the Datalogger can be found under the directory Datalog (see name of the Datalogger). This is located under the directory WWW (to be accessible from the web server of the PLC). Each Collection has its own directory set up, here Measurement (see name of the Collection). Under the directory of the collection is the directory for the year (2011) and under it are directories for individual months (07). In these directories CSV files for particular days can be found. The name of each file has 8 characters (numbers), first two numbers show the day (for the file created on 18.7.2011 it will be 18). Another 6 numers states hours, minutes and seconds when the file was created. I. e. that in the file with the name 18151824.CSV saved in the directory DATALOG/MERENI/2011/07/ are data measured from 18.7.2011 15:18:24.



Actual status of variables of the Datalogger can be seen on the following picture:



Produced CSV file appears as follows:

```
Tecomat 100 CP1004K V6.6 DataLogger v1.0 (2011-07-18-15:18:24) : Datalog /
Mereni - Měření tlaku a teploty ! Type : PERIODIC | TIME : %Thh:mm:ss | REAL :
%8.1f | REAL : %6.1f | ,;
TIME;tlak;teplota
15:18:24;  1007,9;  22,0;
15:19:24;  1007,9;  22,1;
15:20:24;  1007,9;  22,2;
15:21:24;  1007,9;  22,3;
15:22:24;  1007,9;  22,4;
15:23:24;  1007,9;  22,5;
15:24:24;  1007,9;  22,6;
15:25:24;  1007,9;  22,7;
15:26:24;  1007,9;  22,8;
15:27:24;  1007,9;  22,9;
15:28:24;  1007,9;  23,0;
15:29:24;  1007,9;  23,1;
15:30:24;  1007,9;  23,2;
15:31:24;  1007,9;  23,3;
15:32:24;  1007,9;  23,4;
15:33:24;  1007,9;  23,5;
15:34:24;  1007,9;  23,6;
```

If we want the data saving into the CSV file to be undertaken e.g. Only within time from 8:00 to 15:00, then we can use the Collection control via the variable *Datalogger.collection[0].OUT.di-sable*. The relevant program could appear as follows:

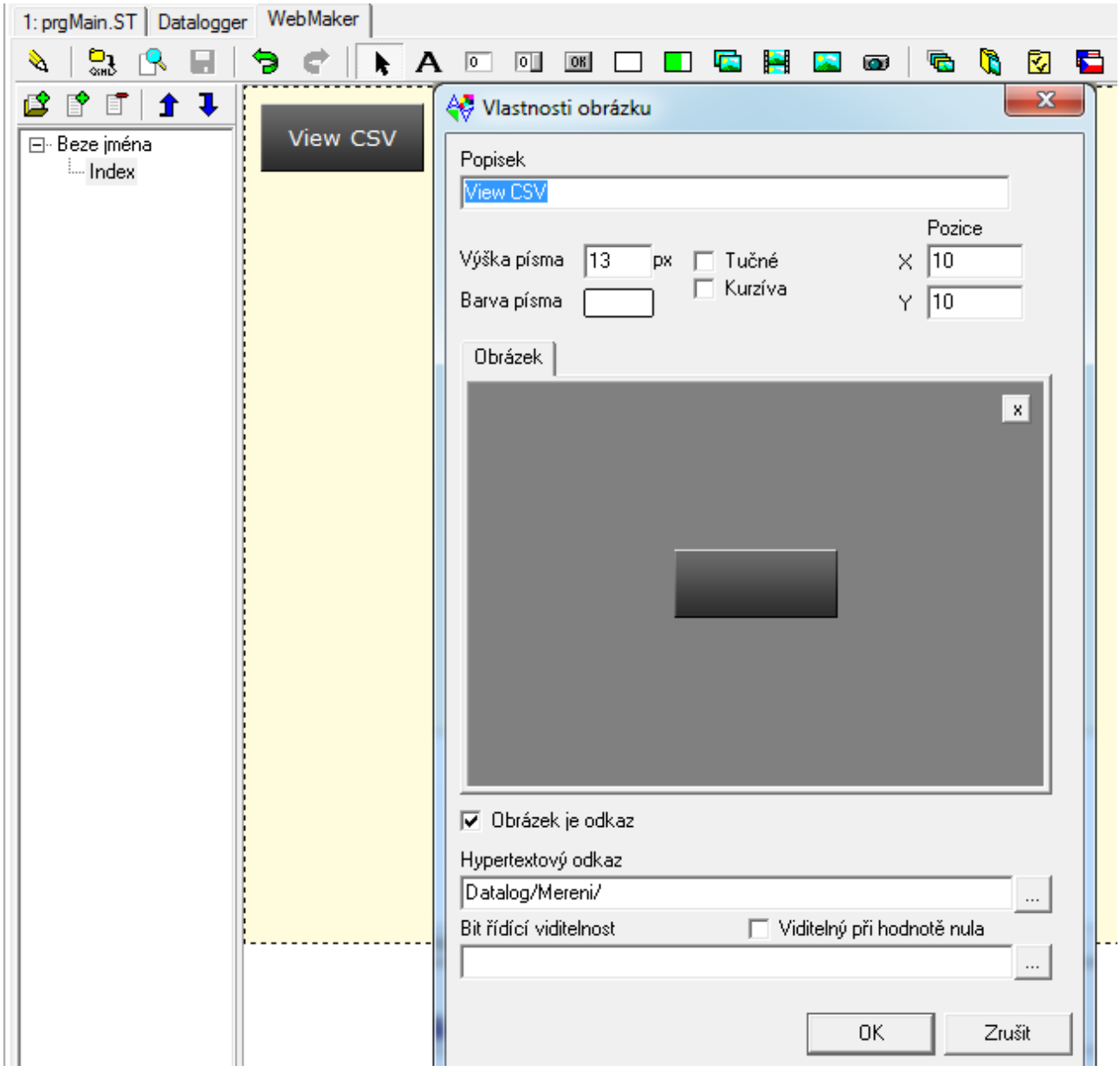
```
VAR_GLOBAL
  presure     : REAL;
  temperature : REAL;
END_VAR

PROGRAM prgMain
  VAR
    actTime    : TIME;
  END_VAR

  // ukladat data do csv souboru pouze od 8:00 do 15:00
  actTime := GetTime();         // aktualni cas
  Datalogger.collection[0].OUT.disable := actTime < T#8h0m OR actTime > T#15h0m;
END_PROGRAM
```

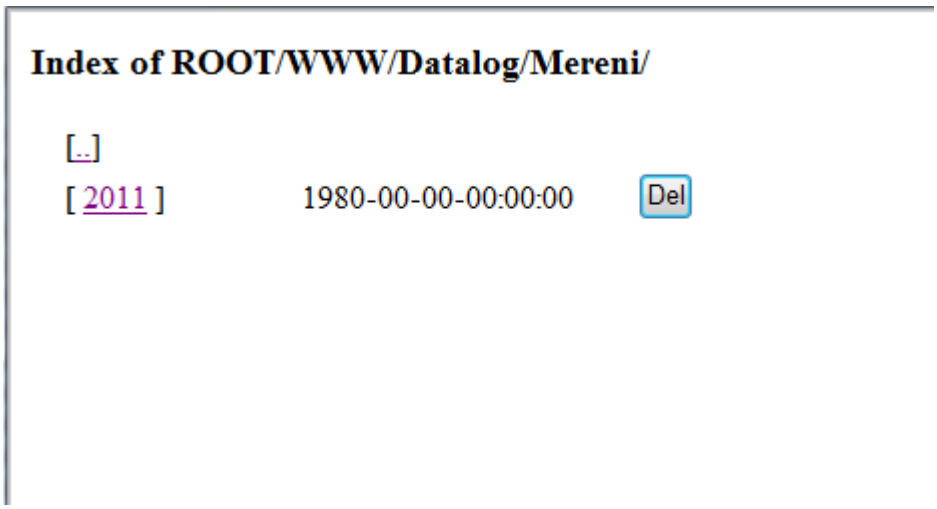
The access to CSV files via the web browser

For the access to the CSV files from the web browser, it is only neccessary to insert in the WebMaker into the web page, for example, statictical picture and fill in the features of the picture the field *Hypertext reference* where we enter the directory the content of which will be displayed after the click on the picture in the browser. In our case it will be *Datalog/Measurement/*.

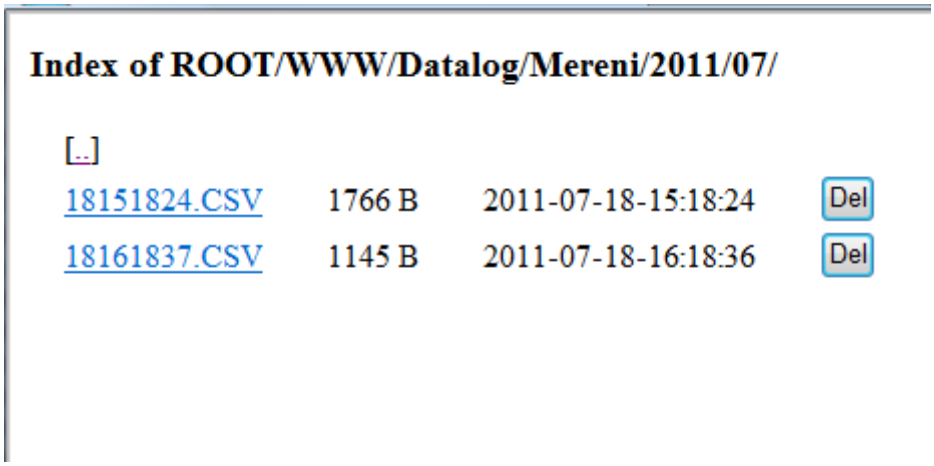


The default situation in the browser can look like this:

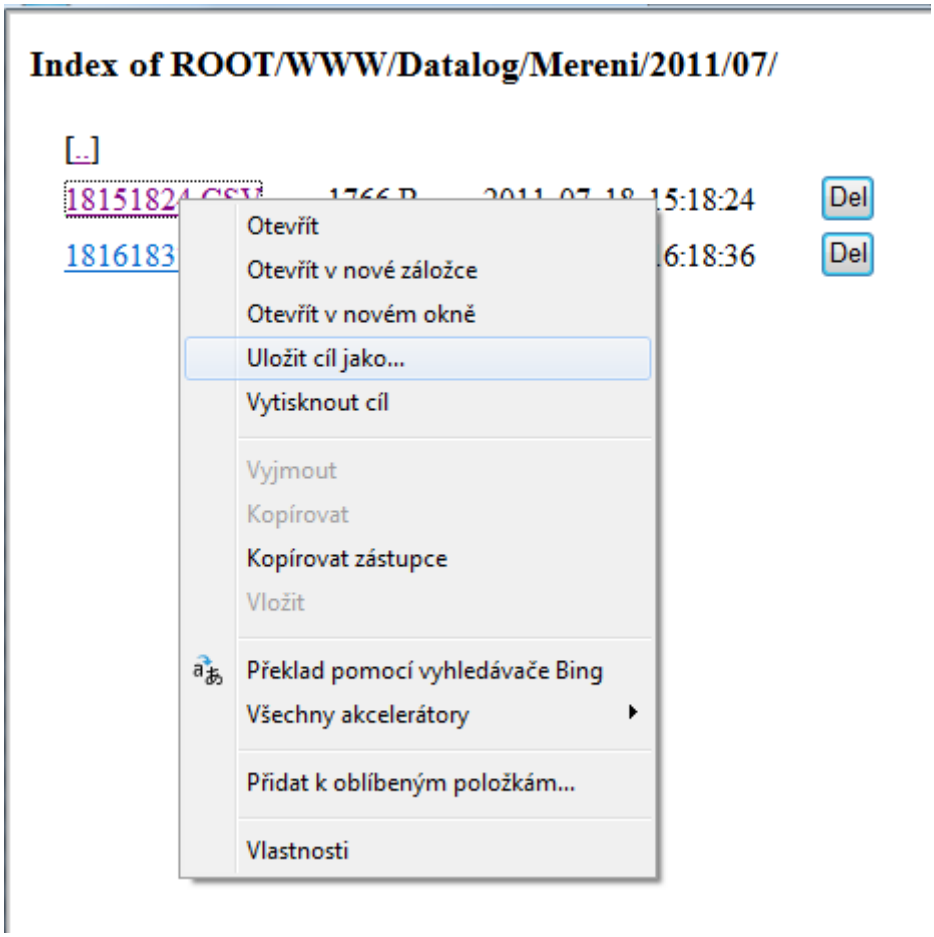
By clicking the button *View CSV,* the browser will be directed to the view of the directory *WWW/DATALOG/MERENI/*. This directory contains the directory *2011* where all CSV files are saved during this year.



By clicking the [*2011]* , we enter the directory *2011*. Further structuring is, as we already know, by months. Therefore, we click on the month which data we are interested in and the list of CSV files will display. The situation can then appear as follows:



By clicking the right mouse button on the file name, the menu will appear where we can select the file opening or file saving in the computer. By pressing the button *Del* on the page, the file can be deleted. The described procedure can differ according to the browser version.



Objednávky a informace:
Teco a. s. Havlíčkova 260, 280 58 Kolín 4, tel. 321 737 611, fax 321 737 633

TXV 003 30.01