CSV Utility SL

The library “CSV Utility SL” enables the reading and writing of CSV files with a CODESYS control.

Product description

The library `CSV Utility SL` provides function blocks to read and write CSV files. The library contains the components `CSVReader` (read a CSV file) and `CSVWriter` (write a CSV file). All function blocks can be used in a classical or object oriented way. An example project demonstrates the usage of all components.

Read a CSV file

The folder `CSVReader` contains the components to read CSV files. The initialization is done by the function block `CSVReaderInit`. The values of a CSV file can be read by element, by line or in one action. The functions blocks `NextLine`, `NextElement` and `ReadAll` are used to read the values of a CSV file. The usage of these components is demonstrated in the example project `CSV Utility SL Examples.project` in the application `CSVReaderExample`. The maximum length of an element must be greater or equal than the maximum length of a column.

Write a CSV file

The folder `CSVWriter` contains the components to write CSV files. Data is saved in three steps. Within the first step the library will be initialized and the path and the file name will be defined. Within the next step the data will be stored in a buffer. During the last step the content of the buffer will be written into a CSV file.

Classical programming

The function blocks for the classical programming can be found in the directory `FunctionBlocks`. For each function block an additional visualization object will be created, which can be used in other projects. An example can be found in the project `CSV Utility SL Examples.project` in the application `CSVWriterCFCExample`.

Object-oriented programming

The function blocks and interfaces for the object oriented programming are stored in the directory `Objects`. An example can be found in the project `CSV Utility SL Examples.project` in the application `CSVWriterSTExample`.

Function blocks in the directory “FunctionBlocks”

The function blocks in this directory are designed for an asynchronous handling and are derived from the function block `CBM.ETrigA`.

Initialization:

The initialization will be done with the function block `Init`. Here the file path and the file name will be assigned.

To add values:

Values can be added to the buffer via the function block `AddXXX`. A new line will be added with the function block `NewLine`. A new file will be created with `NewFile`.

Save:

The writing of the buffer will be done via the function block `WriteFile`. The values will be written in the standard format of the corresponding data type.

Visualization:

For every function block a visualization of the in- and outputs exists.

Important note for the usage of the function blocks:

All function blocks in the directory `FunctionBlocks` react to a rising edge of the input `xExecute`. Please remember that the operation will be executed asynchronous and the oputputs `xDone`, `xError`, `eError` will be sampled respectively. The execution will not take place within a cycle! (Description can be found in the CAA Guidelines in the CODESYS Online Help.)

Function Block CSVWriter (Directory “Objects”)

The function block `CSVWriter` provides general methods to save CSV files.

To add values:

By calling an Add-method a value will be written into the buffer. A new line will be attached as soon as the corresponding method is called. If the buffer is full an error message will be returned. Again, after saving the data, space is available for new values.

- `AddXXX`: These methods insert data at the end of the buffer
- `NewLine`: A new line will be attached

Save:

- `Save`: Saves the value of the buffer in a CSV file.

At the beginning the `InitSave` method must be called to create a new file or to open an existing file.

Additional methods:

- `GetFileSize`: This method returns the size of the current file.

- `Newfile`: This method creates a new file (with a known or automatically generated name).

Error: All methods have an error field, where a possible error of the type `CSV_ERROR` will be stored.

Example project “CSV Utility SL Examples”

The project `CSV Utility SL Example.project` contains three example applications. One example shows the usage of the `CSVReader` and two examples show the usage of the `CSVWriter`.

Application “CSVReaderExample”

The application shows the three ways to read values from a CSV file. The program `Prog` demonstrates the usage of the function blocks `CSVReaderInit`, `NextElement`, `ReadAll` and `NextLine`. A visualizations displays the values of the file `"CSVReader.csv"`.

Note

To test the examples on a CODESYS Control Win the file “CSVReader.csv” can be copied from the target directory of the installation into the directory “c:\temp”.

Application “CSVWriterCFCExample”

This example is based on a CFC example and shows how the functions blocks of the library `CSVWriter` can be used. The integrated visualization will be demonstrated as well.

Functionality:

The program `WriteValues` creates a CSV file in the directory `"c:/temp/CSVWriterCFCExampleData.csv"`. Within the visualization the file will be written via a click on the button “Click here to add “. The initialization will be done via the function block `CSV.Init`. The file path and the file name will be transferred here. The function block `AddWord`, `AddString` and `NewLine` demonstrate the attachment of values to the integrated buffer. The counter ctu increases the line number during each event. After the attachment of a new line the values will be written via the function block `WriteFile` into the CSV file.

Application “CSVWriterSTExample”

This example explains the possibility to write variable values into a CSV file via an object oriented and a sequential way. Each process is executed in two tasks. One task writes the value into a buffer and the other writes the data from the buffer into a file.

Functionality:

- `ObjectOriented_Save:`

  This program calls the `Save` method from `DataObject`, in which the save procedure is implemented.

- `ObjectOriented_Write:`

  This program calls the `Write` method from `DataObject`, in which the write procedure is implemented.

- `DataObject:`

  This function block implements the interface `ICSVObject`. The `Write` method includes the local variables, which will be stored later. For every data type the corresponding `AddXXX` method will be called. After a couple of lines a new file with other values will be created. As soon as an error occurs, the error status will be reached and nothing will be written anymore. The method `Save` implements the save procedure. At the beginning the method `InitSave` of the `CSVWriter` instance will be called. After that the `Save` method will be called.

  To activate the following programs the POU”s in the tasks have to be changed.

- `Sequential_Save:`

  The save procedure must be started with the method `InitSave`. As soon as the CSV file has reached a defined size, a new file will be created. If there is no file name defined the previous name with automatic numeration will be used.

- `Sequential_Write:`

  Different variables will be written in the buffer. If the buffer is full the error `EndOfBuffer` will be returned. In contrast to the `Sequential_Save` described above the error will be ignored and the buffer will be written again. Some data will get lost, but the program is still running. As soon as the next save process has been started the buffer is free again.

General information

Supplier:

CODESYS GmbH
Memminger Strasse 151
87439 Kempten
Germany

Support:

Technical support is not included with this product. To receive technical support, please purchase a CODESYS Support Ticket.

https://support.codesys.com

Item:

CSV Utility SL

Item number:

Sales/Source of supply:

CODESYS Store
https://store.codesys.com

Included in delivery:

CODESYS Package

System requirements and restrictions

Programming System	CODESYS Development System V3.5.15.0
Runtime System	CODESYS Control V3.5.15.0
Supported Platforms/ Devices	All Note: Use the project “Device Reader” to find out the supported features of your device. “Device Reader” is available for free in the CODESYS Store.
Additional Requirements	-
Restrictions	-
Licensing	License activation optional on CODESYS Key or Soft Key (Soft Key: free of charge component of CODESYS Controls) Licensing via Soft Key is strictly linked to hardware. Note: Without a license the software runs for 30 minutes in demo mode.
Required accessories	-

Note: Technical specifications are subject to change. Errors and omissions excepted. The content of the current online version of this document applies.

Creation date: 2023-08-21