FB Visu Creator

The product “FB Visu Creator” is a plug-in which automatically creates visualizations from function blocks.

Product description

The ‘FB Visu Creator’ plug-in automatically creates visualizations from function blocks. All desired function blocks and the input and output fields can be marked with attributes. The tool generates visualizations for the marked function blocks. Afterwards the visualization can be linked with an instance of the function block by a frame reference.

The generation of the visualization is controlled by attributes. There are attributes for functions blocks and attributes for input and output fields.

1.1 Attributes for function blocks

Attribute	Function
`{attribute 'FBVisuCreator'}`	Tags a function block to create a visualization. Example (The FB “Test” should be visualized): `{attribute FBVisuCreator'} FUNCTION_BLOCK Test`

`{attribute 'FBVisuCreatorTextColor':= 'COLOR' }`	Color of the text.
	Argument: Name of the color (english) or `#rrggbb` HTML color code, e.g. `#000000` for black or `#ffffff` for white.
	Example:
	`{attribute 'FBVisuCreatorTextColor' := 'Green'}`
	`{attribute 'FBVisuCreatorTextColor' := "#4223FF"}`

`{attribute 'FBVisuCreatorTextStyleColor' := 'COLOR' }`	Text color from a style. Argument: Name of the style color as defined in the file `styledef.xml`, e.g. `'Element-Alarm-Color'`. Example: `{attribute 'FBVisuCreatorTextStyleColor' := 'Element-Alarm-Color'}` The attribute `TextStyleColor` overwrites the attribute `TextColor`.
`{attribute 'FBVisuCreatorCustomName' := 'NAME' }`	Custom name for the visualization header. Default is the name of the FB. Argument: String with the Name ‘NEW-NAME’ Example: `{attribute 'FBVisuCreatorCustomName' := 'NEW-NAME'}`
`{attribute 'FBVisuCreatorPrefix' := 'PREFIX' }`	Prefix of the name of the visualization. All created visualizations are named by a prefix and the name of the function block. The default prefix is VISU_NEW_”.

`{attribute 'FBVisuCreatorTemplate':= 'NAME_TEMPLATE' }`	Links a visualization to a template function block. All marked inputs and outputs of the template function block will be shown in the created visualization. A template can be used for function blocks with often used inputs and outputs or for basic function blocks with derivatives. Within the template the attribute `FBVisuCreator` has to be set. All inputs and outputs of the templates are shown in addition to the inputs and outputs of the corresponding function block. The template itself is a function block with the attribute `FBVisuCreator`. Example:: A use case of templates is the visualization of inputs and outputs from parent classes. If you want to create a visualization of the input field `xExecute` of a derived function block with the base of `ETRIG` you can create a template FB with the input `xExecute`. The template function block only contains the desired inputs and outputs and the attributes (see Example 2 and 3).
`{attribute 'FBVisuCreatorInputWidth' := '200'}`	Option: Overwrites the default width of input fields (in pixel).
`{attribute 'FBVisuCreatorOutputWidth' := '400'}`	Option: Overwrites the default width of output fields (in pixel).

1.2 Attributes for input and output fields

Attribute	Function

`{attribute 'FBVisuCreatorFormatString' := 'FORMATSTRING' }`	Sets the format string for the variable
	The format string should be set for each visualized variable.
	Argument: Format string in c style, e.g. %1.2f
	Format strings:
	`%d %i: Decimal signed integer.`
	`%o: Octal integer.`
	`%x: Hex integer.`
	`%u: Unsigned integer.`
	`%c: Character.`
	`%s: String.`
	`%f: double`
	Example:
	`{attribute 'FBVisuCreatorFormatString' := '%1.2f'}`

`{attribute 'FBVisuCreatorExcludeEntry'}`	Excludes an input or output.

`{attribute 'FBVisuCreatorTextColor' := 'COLOR' }`	Color of the text.
	Argument: Name of the color (english) or `#rrggbb` HTML color code, e.g.. `#000000` for black or `#ffffff` for white.
	Example:
	`{attribute 'FBVisuCreatorTextColor' := 'Green'}`
	`{attribute 'FBVisuCreatorTextColor' := '#4223FF'}`

`{attribute 'FBVisuCreatorTextStyleColor':= 'COLOR' }`	Text color from a style Argument: Name of the style color as defined in `styledef.xml`, e.g. `'Element-Alarm-Color'` Example: `{attribute 'FBVisuCreatorTextStyleColor' := 'Element-Alarm-Color'}` The attribute `TextStyleColor` overwrites the attribute `TextColor`.

`{attribute 'FBVisuCreatorCustomName' := 'NAME' }`	Name of thr variable in the visualization. Argument: String with the name “New variable name” Example: `{attribute 'FBVisuCreatorCustomName' := "New variable name"}`
`{attribute 'FBVisuCreatorBoolColor' := 'COLOR' }`	Color of the lamp or the switch. This attribute can only be used for BOOL values. Argument: Blue, Gray, Yellow, Green, Red. Default: Gray Example: `{attribute 'FBVisuCreatorBoolColor' := 'Blue'}`

`{attribute 'FBVisuCreatorBitField'}`	Creates a 4x8 field of bits.
	This attribute can only be used for DWORD values.
	For DWORD as input variable the single bits can be selected per mouse click.
	Example:
	`{attribute 'FBVisuCreatorBitField'}`
	`dwErrorCode : DWORD;`

`{attribute 'FBVisuCreatorMembers' := '.member1, format1, .member2, format2..' }`
Pointer:
`{attribute 'FBVisuCreatorMembers' :=  '^.member1, format1, ^.member2, format2..' }`
	The attribute `FBVisuCreatorMembers` can be used to generate input and output fields for STRUCTs and UNIONs. Each member variable and the format string are listed in the attribute value.
	The format string “Bool” will generate switches and lamps for the given member variable. The standard format strings are described in chapter 1.2., Attribute ‘FBVisuCreatorFormatString’
	See example 4 for further information.

1.3 Start the generator

If all attributes are set then the visualizations can be created by the button “Generate visualizations” (Menu/ Project). Afterwards the visualization can be linked with an instance of the function block by a frame reference.

2. Examples

2.1 Simple configuration without a template

```
{attribute 'FBVisuCreator'}
FUNCTION_BLOCK AddINT
VAR_INPUT
        {attribute 'FBVisuCreatorFormatString' := '%d'}
        iInput : INT;
END_VAR
VAR_OUTPUT
        {attribute 'FBVisuCreatorFormatString' := '%d'}
        eError : ERROR;
END_VAR
```

2.2 Configuration with a template

```
{attribute 'FBVisuCreatorTemplate' :=  'ETrigATemplate'}
FUNCTION_BLOCK Init EXTENDS CBM.ETrigA
VAR_INPUT
        sDirectoryPath         : STRING; (* The directory, where the file is saved, e.g. "C:\direct
        sFileName              : STRING; (* The file name, e.g. "file.csv" *)
        sRowSeparator          : STRING := '$R$N'; (* The row separator. Default: '$n' *)
        sColumnSeparator       : STRING := ';'; (* The column separator. Default: ';' *)
END_VAR
VAR_OUTPUT
        eError : ERROR;
END_VAR
VAR_IN_OUT
        rCSVWriter : CSVWriter; (* Reference to the CSVWrtiter-FB *)
END_VAR
VAR
END_VAR
```

2.3 Template function block

```
{attribute 'FBVisuCreator'}
(* Template for the FBVisuGenerator. Don't use this FB in your code. *)
FUNCTION_BLOCK ETrigATemplate
VAR_INPUT
        xExecute       : BOOL;
        xAbort         : BOOL;
END_VAR
VAR_OUTPUT
        {attribute 'FBVisuCreatorFormatString' := '%d'}
        eError                  : ERROR;
        {attribute 'FBVisuCreatorBoolColor' := 'Green'}
        xDone          : BOOL;
        {attribute 'FBVisuCreatorBoolColor' := 'Green'}
        xBusy          : BOOL;
        {attribute 'FBVisuCreatorBoolColor' := 'Red'}
        xError         : BOOL;
        {attribute 'FBVisuCreatorBoolColor' := 'Red'}
        xAborted       : BOOL;
END_VAR
```

2.4 Visualization of member variables

Structure (Example):

```
TYPE Struct1 :
STRUCT
        x : INT;
        y : INT;
        z : INT;
        b : BOOL;
        re : REAL;
        str : STRING;
        dw : DWORD;
END_STRUCT
END_TYPE
```

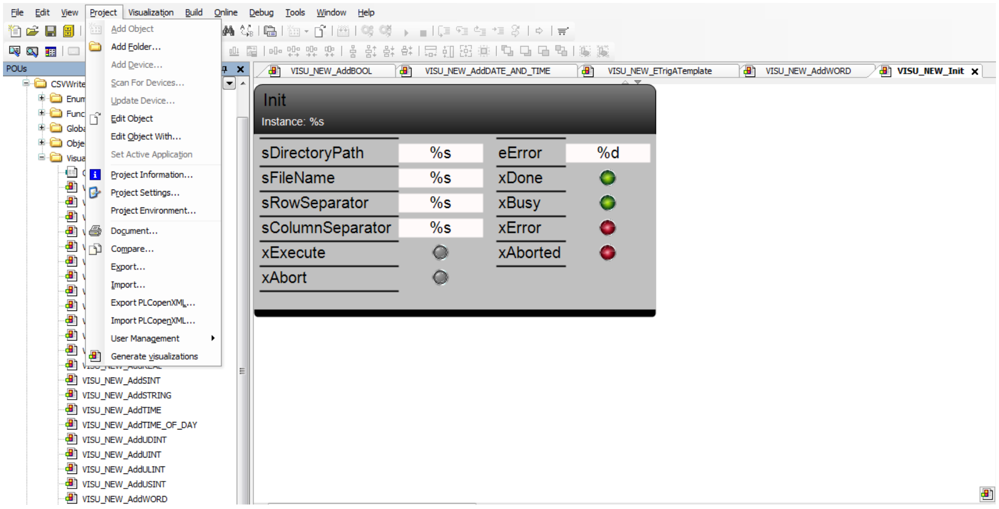
Declaration:

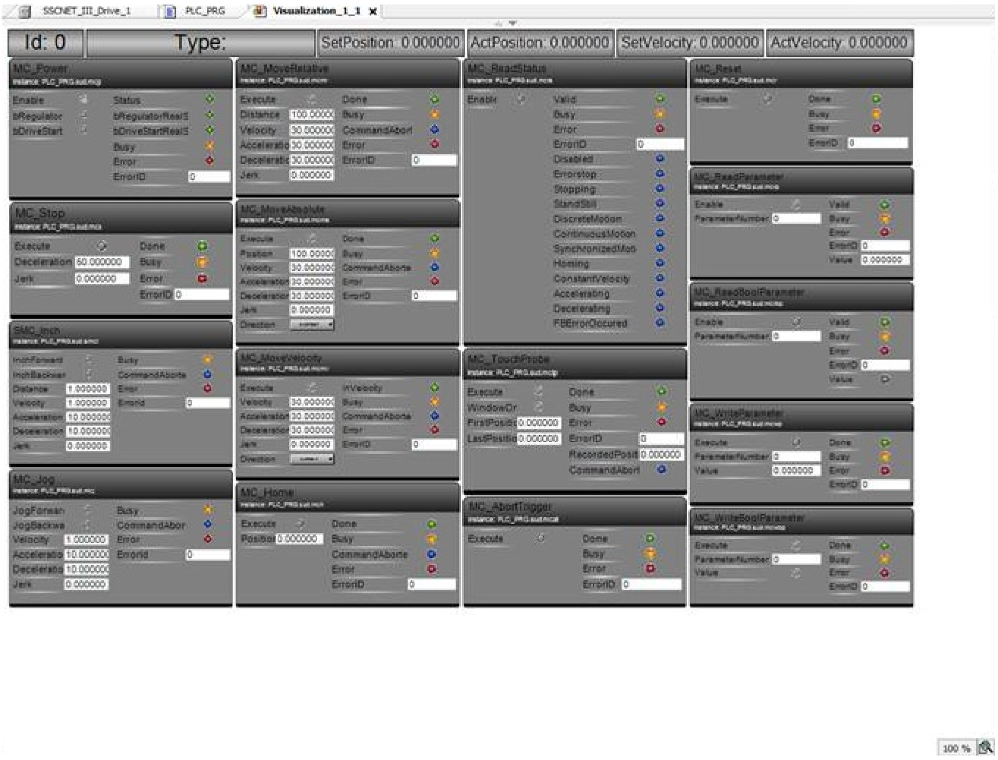
```
{attribute 'FBVisuCreator'}
FUNCTION_BLOCK FB1
VAR_INPUT
        {attribute 'FBVisuCreatorFormatString' := '%s'}
        a : STRING;
        {attribute 'FBVisuCreatorFormatString' := '%s'}
        b : STRING;
        {attribute 'FBVisuCreatorMembers' := '.x, %d, .y, %d, .z, %d, .b, Bool, .re, %f, .str, %s,
        s1 : Struct1;

END_VAR

        {attribute 'FBVisuCreatorMembers' := '         .x, %d,
                                                       .y, %d,
                                                       .z, %d,
                                                       .b, Bool,
                                                       .re, %f,
                                                       .str, %s,
                                                       .dw, %d'}
        so : Struct1;
        ao : STRING;
        bo : STRING;
END_VAR
VAR
END_VAR
```

Screenshots





General information

Vendor:

CODESYS GmbH
Memminger Strasse 151
87439 Kempten
Germany

Support:

https://support.codesys.com

Item:
FB Visu Creator
Item number:
000055
Sales:

CODESYS Store
https://store.codesys.com

Included in delivery:
CODESYS Package

System requirements and restrictions

Programming System	CODESYS Development System Version 3.5.8.0 or higher
Runtime System	CODESYS Control Version 3.5.8.0
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	Only function blocks of libraries and POU pools can be used to create visualizations.

Note: Not all CODESYS features are available in all territories. For more information on geographic restrictions, please contact sales@codesys.com.
Note: Technical specifications are subject to change. Errors and omissions excepted. The content of the current online version of this document applies.