Name: SFN_StringUtils

Revision: 0.9

Ladder: false
Complex: false

Graphic Schema:

SFN_StringUtils	
	ToObject
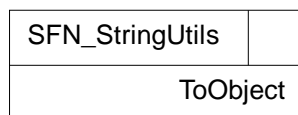
Comment: String utilities.
Note: Lasal OS supports now also direct access to string functions.
(see _memset() and

Server: Name: ToObject
            GUID: {F1A26105-F750-432C-9382-46CDE636176F}
            Class: SFN_StringUtils
            Visualized: false
            DataType: DINT
            Type: Object Channel
            Initialize: false
            WriteProtected: true
            Retentive: false

Methods: Name: GetNextDelimiter
            Virtual: false
            Global access: true
            AWL implementation: false
            CDecl: false
            Input::   Name: p_Buf
                                   Type: ^CHAR
                                   Pointer: true
                                   Register: <undefined>
            Input::   Name: p_Delimiters
                                   Type: ^CHAR
                                   Pointer: true
                                   Register: <undefined>
            Output::   Name: p_DelimiterChar
                                   Type: ^CHAR
                                   Pointer: true
                                   Register: <undefined>

            Name: GetSubString
            Virtual: false
            Global access: true
            AWL implementation: false
            CDecl: false
            Comment: This method returns a zero-terminated substring.
                                   Parameters:
                                   -Delimiters   : a string containing the delimiters.
                                   -nrDelimiters   : the number of delimiters in the delimiter string.
                                   -InputBufferSize : The size of the imput buffer
                                   -p_InputBuffer   : pointer to the input buffer
                                   -p_Substring   : pointer to the string the result wwill be stored in
                                   Returns:
                                   -If the substring is parsed successfully: the size of the returned substring
                                   -No delimiter is found: -1
            Input::   Name: Delimiters
                                   Type: ^CHAR
                                   Pointer: true
                                   Register: <undefined>
            Input::   Name: NrDelimiters
                                   Type: BYTE
                                   Pointer: false
                                   Register: <undefined>
            Input: Name: InputBufferSize
                                   Pointer: false

	Register:	<undefined>	
Input::	Name:	p_InputBuffer	
	Type:	^CHAR	
	Pointer:	true	
	Register:	<undefined>	
Input::	Name:	p_Substring	
	Type:	^CHAR	
	Pointer:	true	
	Register:	<undefined>	
Output::	Name:	NrCharsParsed	
	Type:	INT	
	Pointer:	false	
	Register:	<undefined>	

Name:	IsNumChar
Virtual:	false
Global access:	true
AWL implementation:	false
CDecl:	false
Comment:	This method determines if a character is of type numeric.
	Parameters:
	-c : character to be analyzed
	Returns:
	-true : character is numeric.
	-false: character is not numeric.

Input::	Name:	c
	Type:	CHAR
	Pointer:	false
	Register:	<undefined>
Output::	Name:	RetVal
	Type:	BOOL
	Pointer:	false
	Register:	<undefined>

Name:	IsAlphaChar
Virtual:	false
Global access:	true
AWL implementation:	false
CDecl:	false
Comment:	This method determines if a character is of type alpha.
	Parameters:
	-c : character to be analyzed
	Returns:
	-true : character is alpha.
	-false: character is not alpha.

Input::	Name:	c
	Type:	CHAR
	Pointer:	false
	Register:	<undefined>
Output::	Name:	RetVal
	Type:	BOOL
	Pointer:	false
	Register:	<undefined>

Name:	IsAlphaNumChar
Virtual:	false
Global access:	true
AWL implementation:	false
CDecl:	false
Comment:	This method determines if a character is alphanumeric.
	Parameters:
	-c : character to be analyzed
	Returns:
	-true : character is alphanumeric.
	-false: character is not alphanumeric.

Input::	Name:	c
	Type:	CHAR
	Pointer:	false
	Register:	<undefined>
Output::	Name:	RetVal
	Pointer:	false

Name:	AddCharInString		
Virtual:	false		
Global access:	false		
AWL implementation:	false		
CDecl:	false		
Input::	Name:	p_string	
	Type:	pChar	
	Pointer:	false	
	Register:	\<undefined\>	
Input::	Name:	charValue	
	Type:	CHAR	
	Pointer:	false	
	Register:	\<undefined\>	

Name:	ConvertUdintToAsciiHex		
Virtual:	false		
Global access:	true		
AWL implementation:	false		
CDecl:	false		
Input::	Name:	p_string	
	Type:	^CHAR	
	Pointer:	true	
	Register:	\<undefined\>	
Input::	Name:	Value	
	Type:	UDINT	
	Pointer:	false	
	Register:	\<undefined\>	
Input::	Name:	valueSize	
	Type:	UDINT	
	Pointer:	false	
	Register:	\<undefined\>	

Name:	ConvertUdintToAsciiDec		
Virtual:	false		
Global access:	true		
AWL implementation:	false		
CDecl:	false		
Input::	Name:	p_string	
	Type:	^CHAR	
	Pointer:	true	
	Register:	\<undefined\>	
Input::	Name:	Value	
	Type:	UDINT	
	Pointer:	false	
	Register:	\<undefined\>	

Name:	ConvertDintToAsciiHex		
Virtual:	false		
Global access:	true		
AWL implementation:	false		
CDecl:	false		
Input::	Name:	p_string	
	Type:	^CHAR	
	Pointer:	true	
	Register:	\<undefined\>	
Input::	Name:	Value	
	Type:	DINT	
	Pointer:	false	
	Register:	\<undefined\>	
Input::	Name:	valueSize	
	Type:	UDINT	
	Pointer:	false	
	Register:	\<undefined\>	

Name:	ConvertDintToAsciiDec		
Virtual:	false		
Global access:	true		
AWL implementation:	false		
CDecl:	false		
	Type:	^CHAR	

	Register:	<undefined>	
Input::	Name:	Value	
	Type:	DINT	
	Pointer:	false	
	Register:	<undefined>	

Name:	ConvertAsciiToDint
Virtual:	false
Global access:	true
AWL implementation:	false
CDecl:	false

Input::	Name:	p_string
	Type:	^CHAR
	Pointer:	true
	Register:	<undefined>
Output::	Name:	Output
	Type:	DINT
	Pointer:	false
	Register:	<undefined>

Name:	ConvertAsciiHexToDint
Virtual:	false
Global access:	false
AWL implementation:	false
CDecl:	false

Input::	Name:	p_string
	Type:	^CHAR
	Pointer:	true
	Register:	<undefined>
Output::	Name:	Output
	Type:	DINT
	Pointer:	false
	Register:	<undefined>

Name:	ConvertAsciiDecToDint
Virtual:	false
Global access:	false
AWL implementation:	false
CDecl:	false

Input::	Name:	p_string
	Type:	^CHAR
	Pointer:	true
	Register:	<undefined>
Output::	Name:	Output
	Type:	DINT
	Pointer:	false
	Register:	<undefined>

Name:	ConvertAsciiToUdint
Virtual:	false
Global access:	true
AWL implementation:	false
CDecl:	false

Input::	Name:	p_string
	Type:	^CHAR
	Pointer:	true
	Register:	<undefined>
Output::	Name:	Output
	Type:	UDINT
	Pointer:	false
	Register:	<undefined>

Name:	ConvertAsciiHexToUdint
Virtual:	false
Global access:	false
AWL implementation:	false
CDecl:	false

Input::	Name:	p_string
	Type:	^CHAR
	Pointer:	true
Output::	Name:	Output

		Pointer:	false
		Register:	<undefined>
Name:	ConvertAsciiDecToUdint		
Virtual:	false		
Global access:	false		
AWL implementation:	false		
CDecl:	false		
Input::		Name:	p_string
		Type:	^CHAR
		Pointer:	true
		Register:	<undefined>
Output::		Name:	Output
		Type:	UDINT
		Pointer:	false
		Register:	<undefined>
Name:	StringCat		
Virtual:	false		
Global access:	true		
AWL implementation:	false		
CDecl:	false		
Input::		Name:	p_Dest
		Type:	^CHAR
		Pointer:	true
		Register:	<undefined>
Input::		Name:	p_Add
		Type:	^CHAR
		Pointer:	true
		Register:	<undefined>
Name:	StringCatLimited		
Virtual:	false		
Global access:	true		
AWL implementation:	false		
CDecl:	false		
Input::		Name:	p_Dest
		Type:	^CHAR
		Pointer:	true
		Register:	<undefined>
Input::		Name:	p_Add
		Type:	^CHAR
		Pointer:	true
		Register:	<undefined>
Input::		Name:	DestSize
		Type:	UDINT
		Pointer:	false
		Register:	<undefined>
Name:	StringCompare		
Virtual:	false		
Global access:	true		
AWL implementation:	false		
CDecl:	false		

Comment: This method compares 2 zero-terminated strings. The caller has to make sure both strings are zero-terminated.
Parameters:
-string1: A pointer to the first string
-string2: A pointer to the second string
Returns:
-true   : if string2 and string2 are equal
-false : if string1 and string2 are not equal.

Input::		Name:	p_String1
		Type:	^CHAR
		Pointer:	true
		Register:	<undefined>
Input::		Name:	p_String2
		Type:	^CHAR
		Pointer:	true
		Register:	<undefined>
		Type:	BOOL

		Register:	<undefined>

Name:	StringCopy
Virtual:	false
Global access:	true
AWL implementation:	false
CDecl:	false
Comment:	Description:

Copies a zero-terminated string from source to destination.
Parameters:
-p_Src   : pointer to the source string
-p_Dest   : pointer to the destination string
Returns   : nothing

Input::	Name:	p_Src
	Type:	^CHAR
	Pointer:	true
	Register:	<undefined>
Input::	Name:	p_Dest
	Type:	^CHAR
	Pointer:	true
	Register:	<undefined>

Name:	StringLength
Virtual:	false
Global access:	true
AWL implementation:	false
CDecl:	false
Comment:	This method determined the length of a given zero-terminated string. The user has to make sure the supplied

string is zero-terminated.
Parameters:
-p_String : Pointer to the string.
Returns   : The length of the given string including the zero-terminator

Input::	Name:	p_String
	Type:	^CHAR
	Pointer:	true
	Register:	<undefined>
Output::	Name:	Length
	Type:	UINT
	Pointer:	false
	Register:	<undefined>

Name:	StringNCopy
Virtual:	false
Global access:	true
AWL implementation:	false
CDecl:	false
Input::	Name:	p_Src
	Type:	^CHAR
	Pointer:	true
	Register:	<undefined>
Input::	Name:	p_Dest
	Type:	^CHAR
	Pointer:	true
	Register:	<undefined>
Input::	Name:	size
	Type:	UDINT
	Pointer:	false
	Register:	<undefined>

Name:	StrStr
Virtual:	false
Global access:	true
AWL implementation:	false
CDecl:	false
Comment:	Searches a (sub)string in a string
Input::	Name:	p_searchStr
	Type:	^CHAR
	Pointer:	true
	Register:	<undefined>
Input::	Name:	p_lookforStr

	Pointer:	true
	Register:	\<undefined\>
	Comment:	String to look for in search string
Output::	Name:	p_ReturnStr
	Type:	^CHAR
	Pointer:	true
	Register:	\<undefined\>
	Comment:	Return the position in search string
		where the lookfor string is found.
		If lookfor string is not found
		or wrong parameters where used
		(NIL pointers) a NIL is returned

Defines:
Dependencies:
      Types: pChar