SIEMENS

Class:ELOLib 1

COMOS

Platform
Class documentation ELOLib_dll

Programming Manual

09/2014
A5E32035430-AA

Legal information

Warning notice system

This manual contains notices you have to observe in order to ensure your personal safety, as well as to prevent damage to property. The notices referring to your personal safety are highlighted in the manual by a safety alert symbol, notices referring only to property damage have no safety alert symbol. These notices shown below are graded according to the degree of danger.

⚠ DANGER

indicates that death or severe personal injury **will** result if proper precautions are not taken.

⚠ WARNING

indicates that death or severe personal injury **may** result if proper precautions are not taken.

⚠ CAUTION

indicates that minor personal injury can result if proper precautions are not taken.

NOTICE

indicates that property damage can result if proper precautions are not taken.

If more than one degree of danger is present, the warning notice representing the highest degree of danger will be used. A notice warning of injury to persons with a safety alert symbol may also include a warning relating to property damage.

Qualified Personnel

The product/system described in this documentation may be operated only by **personnel qualified** for the specific task in accordance with the relevant documentation, in particular its warning notices and safety instructions. Qualified personnel are those who, based on their training and experience, are capable of identifying risks and avoiding potential hazards when working with these products/systems.

Proper use of Siemens products

Note the following:

⚠ WARNING

Siemens products may only be used for the applications described in the catalog and in the relevant technical documentation. If products and components from other manufacturers are used, these must be recommended or approved by Siemens. Proper transport, storage, installation, assembly, commissioning, operation and maintenance are required to ensure that the products operate safely and without any problems. The permissible ambient conditions must be complied with. The information in the relevant documentation must be observed.

Trademarks

All names identified by ® are registered trademarks of Siemens AG. The remaining trademarks in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owner.

Disclaimer of Liability

We have reviewed the contents of this publication to ensure consistency with the hardware and software described. Since variance cannot be precluded entirely, we cannot guarantee full consistency. However, the information in this publication is reviewed regularly and any necessary corrections are included in subsequent editions.

Siemens AG
Industry Sector
Postfach 48 48
90026 NÜRNBERG
GERMANY

A5E32035430-AA
Ⓟ 09/2014 Subject to change

Table of contents

Class:ELOLib 1

1.1 Public Functions

VDECreateType

```
VDECreateType(ByVal MainObject As Object, VDEType As String) As
Boolean
```

Applies to VDE-compliant cables. After changing a specification, the type and cross-section are re-calculated.

GetColour

```
GetColour(ByVal ComosConnector As IComosDConnector) As String
```

Gets color information from a connector which was set in a plan by means of connection information.

GetConnectorLabel

```
GetConnectorLabel(ByVal Conn As Object) As String
```

Calculates the label of the connection point of an auxiliary contact.

GetAddressSpace

```
GetAddressSpace(ByVal Karte As Object) As String
```

Calculates the address space of a tab.

GetAddress

```
GetAddress(ByVal Channel As Object) As String
```

Calculates the address of a channel.

GetAddrStop

```
GetAddrStop(ByVal Karte As Object, ByVal IO As String) As String
```

Calculates the stop address of a tab.

BelongsToSameBridge

```
BelongsToSameBridge(ByVal StartTerminal As IComosDDevice, _ByVal
PartnerTerminal As IComosDDevice, _ByVal IO As String, ByRef
CableIndex As String) As Boolean
```

Finds out if two terminals belong to the same bridge.

ConnectBySLine

```
ConnectBySLine(ByVal MSL As IComosDConnector) As String
```

Sets electric connections when a single line connection has been established.

DisConnectBySLine

```
DisConnectBySLine(ByVal MSL As IComosDConnector) As String
```

Disconnects electric connections when a single line connection has been disconnected.

CreateConnection

```
CreateConnection(ByRef CP1 As IComosDConnector, ByRef CP2 As
IComosDConnector, _ByVal Cable As IComosDDevice, ByVal Index As
Integer, _ByVal Modus As Integer) As Boolean
```

Creates a connection with a mode:

- Mode = -1: Disconnect connection.
- Mode = 0: If both connectors are free.
- Mode = 1: If connectors are not free, then re-connectors are created.
- Mode = 2: If the attempted connection already exists, the result is returned.
- Mode = 3: Connection is forced, even if connection already exists.

SetWiresBySLine

```
SetWiresBySLine(ByVal MSL As IComosDConnector, ByVal Cable As
IComosDDevice) As String
```

Sets wires in IT connectors, if the cable was set upon a single line connection.

ReleaseWiresBySLine

```
ReleaseWiresBySLine(ByVal Cable As IComosDDevice) As String
```

Disconnects wires in IT-connectors, if the wire was set upon a single line connection.

CloseMarshallingConnections

```
CloseMarshallingConnections(ByVal Document As IComosDDocument, ByVal
Mode As Integer) As String
```

Processes all signals in the objects that are in drag&drop boxes and tries to close or disconnect the marshalling.

AssemblyAssign

```
AssemblyAssign(ByVal SourceObject As IComosBaseObject, _ByVal
TargetObject As IComosBaseObject, _ByVal Action As EActions,
_Optional ByVal Param As Variant, _Optional ByVal SearchDetailClass
As String = vbNullString, _Optional ByVal ActionDev As IComosDDevice
= Nothing) As String
```

- Definition `SourceObject = Object` which is merged, e.g. strip terminal X1, or if SearchDetailClass is set, then SourceObject = Owner (unit), where you search for objects in the target with the same name as all objects with `DetailClass` .
- `TargetObject` = `Owner` (cabinet), whereunder you search for an object with the same name (X1).

You can find additional information on this topic in the manual "EI&C Administration", keyword "Reference of the tab "System data"" and keyword "Reference of the predefined task functions".

ObjectAssign

```
ObjectAssign(ByVal SourceObject As IComosBaseObject, _ByVal
TargetObject As IComosBaseObject, _ByVal Action As EActions,
_Optional ByVal Param As Variant, _Optional ByVal ActionDev As
IComosDDevice = Nothing) As StringOn Error GoTo ComosStdErrorHandler
```

Triggers an action e.g. "place template" or "search object".

You can find additional information on this topic in the manual "EI&C Administration", keyword "Reference of the tab "System data"" and keyword "Reference of the predefined task functions".

UndoAction

```
UndoAction(ByVal ActionDev As IComosDDevice) As String
```

EvalSearchString

```
EvalSearchString(ByVal StartObj As IComosBaseObject, ByVal
SearchString As String, _Optional ByVal SkipEvalLast As Boolean =
False, Optional ByRef NewNameAs String = vbNullString) As
IComosBaseObject
```

Syntax:

- `+L1\L2\B1 => absolute location search`
- `=U1\U2\B1 => absolute unit search`
- `X1\X2\B1 => relative search from StartObj, with possibility`
- `..\..\B1 => change to owner`
- `..(U)\B1 => change to owner by class (e.g. Unit)`

Beginning at the start object, searches for a COMOS object by means of the entered search string and returns it.

1.2 Public Subs

MovePotConnections

```
MovePotConnections(ByVal Node1 As IComosDDevice, ByVal Node2 As
IComosDDevice)
```

Move all potentials and connections from one node to another.

ConnectedObjectAsChild

```
ConnectedObjectAsChild(ByVal Con As IComosDConnector)
```

If a valve is connected with a function, the valve should be moved below the function.

EnumerateReferences

```
EnumerateReferences(ByVal Node As IComosDDevice, ByVal Start As
Integer, ByVal Sort As String)
```

Numbers all references hierarchically below a node. Is applied to the Hierarchy document group.

EnumerateReferences

MoveObjectsToBoard

```
MoveObjectsToBoard(ByVal Platte As IComosDDevice, ByVal Planart As
String)
```

Is used in Fluid. Objects that are surrounded by a ground plate are moved below it.

SetCDevBySpecs

```
SetCDevBySpecs(ByVal Dev As IComosDDevice, _ByVal BaseString As
String, _ByVal SpecName1 As String, _ByVal SpecName2 As String)
```

Switches the base object on functions in dependency of the number of process connections.

SetWireNumbersByCoord

```
SetWireNumbersByCoord(ByVal ReportDocument As REPORTLib.Document)
```

Numbers all wires of a document.

Triathlon_Export

```
Triathlon_Export(ByVal Document As IComosDDocument, ByVal Dateiname
As String)
```

Exports the control cabinet assembly to the Triathlon software.

VDECreateWires

```
VDECreateWires(ByVal MainObject As Object, ByVal VDEType As String)
```

Create wires for VDE cables.

VDECreateWiresNew

```
VDECreateWiresNew(ByVal MainObject As Object, ByVal VDEType As
String)
```

Creates the right wire order for a VDE cable.

DoImplementPotential

```
DoImplementPotential(ByVal SourcePot As IComosDDevice,ByVal
TargetPot As IComosDDevice)
```

Implements two potentials or potential bundles in succession.

UndoImplementPotential

```
Public Sub UndoImplementPotential(ByVal SourcePot As IComosDDevice)
```

Disconnects two potentials or potential bundles from one another.