Schematic API Reference

Summary

This reference provides a concise reference of the Schematic API as part of the Altium Designer Run Time Library.

Technical Reference
TR0139 (v1.7) September 25, 2008

The Schematic Application Programming Interface (API) reference details the object interfaces for schematic objects such as schematic documents and schematic design objects. The Schematic API is defined in the `RT_Schematic` unit which is embedded in the scripting engine or added explicity in the `Uses` clause on a unit in a server project.

Schematic API, Schematic Object Model and Functions

The Schematic API consists of the Schematic Object model and Schematic API functions. The Schematic API is supported by the Schematic Editor in Altium Designer. The Schematic design object interfaces and methods are available to use in your scripts in all script languages that Altium Designer supports.

Object Interfaces

Basically an interface is simply a list of methods that a class declares that it implements. That is, each method in the interface is implemented in the corresponding class. Interfaces are declared like classes but cannot be directly instantiated and do not have their own method definitions. The Schematic design objects are wrapped by their corresponding Schematic interfaces that make it possible to manipulate them.

Main Schematic Object Interfaces

The `ISch_ServerInterface` interface is the main interface in the Schematic API and it represents the main Schematic Editor object. To use Schematic Object interfaces, you need to obtain the `ISch_ServerInterface` interface by invoking the `SchServer` function. `The ISch_ServerInterface` interface is the gateway to fetching other Schematic objects.

The `ISch_GraphicalObject` interface is a generic interface used for all Schematic design object interfaces.

The `ISch_Document`, `ISch_Sheet` and `ISch_Lib` interfaces represent an existing Schematic or library documents.

SchServer function

To obtain the Schematic interface that represents the Schematic editor object, invoke the `SchServer` function in your script which returns you the `ISch_ServerInterface` interface. This object interface obtains the Schematic editor server object and then you can extract data from existing Schematic objects and invoke these Schematic object's methods.

For example, the `SchServer` function is illustrated in light blue color in the example below.

```
Var
    Sheet : ISch_Sheet;
Begin
    Sheet := SchServer.GetCurrentSchDocument
    If Sheet = Nil then Exit;
    // do something here
End;
```

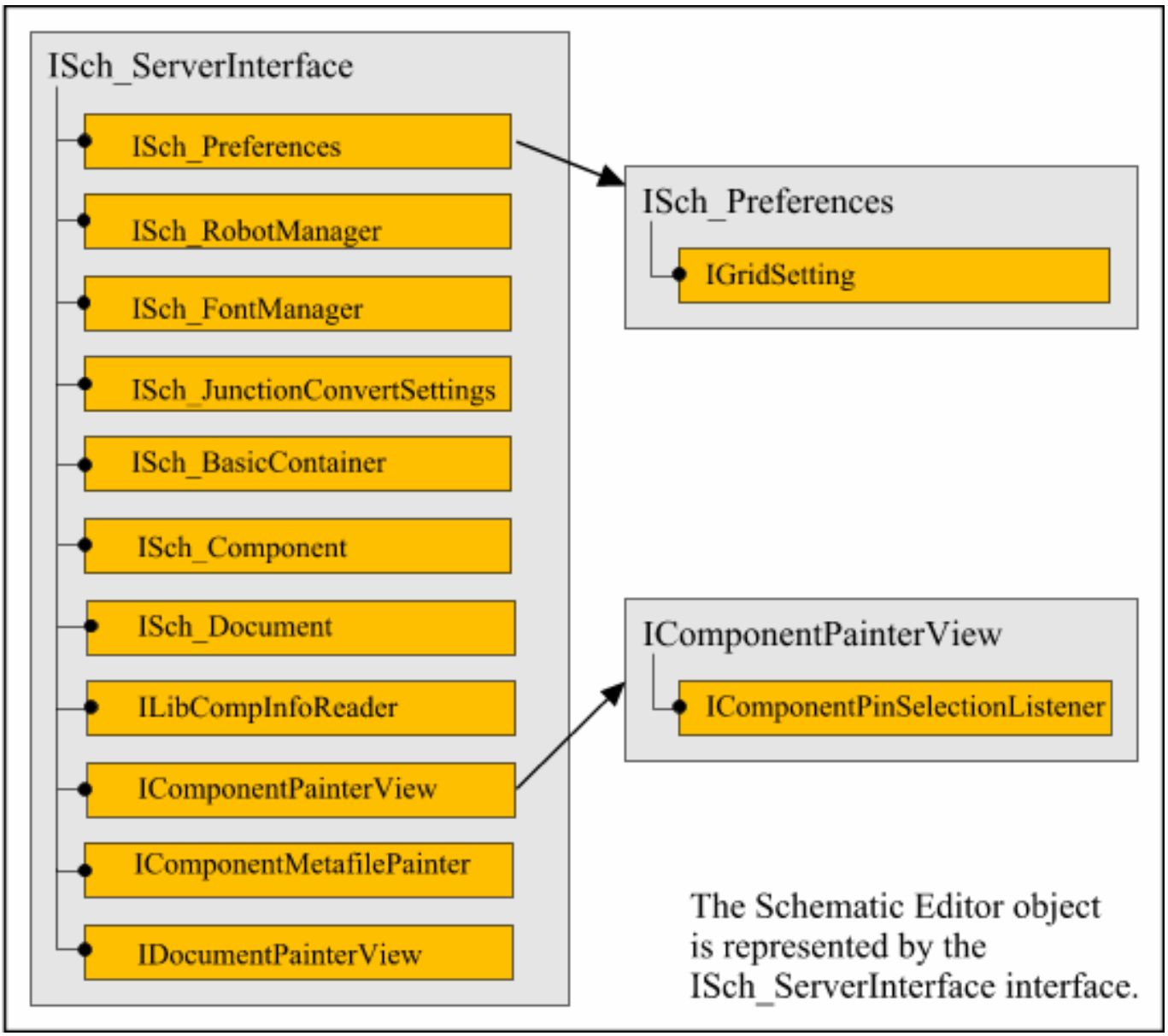
Script Examples

There are Schematic script examples in the Altium Designer's standard installation folder, `\Examples\Scripts\DelphiScript\SCH` folder which demonstrate the use of Schematic interfaces.

Schematic Object Model Hierarchy

The Schematic Object Model compromises of Schematic Object Interfaces and standalone utility functions that allow you to deal with Schematic objects from a Schematic document open in Altium Designer. An object interface is just a means of access to an object in memory.

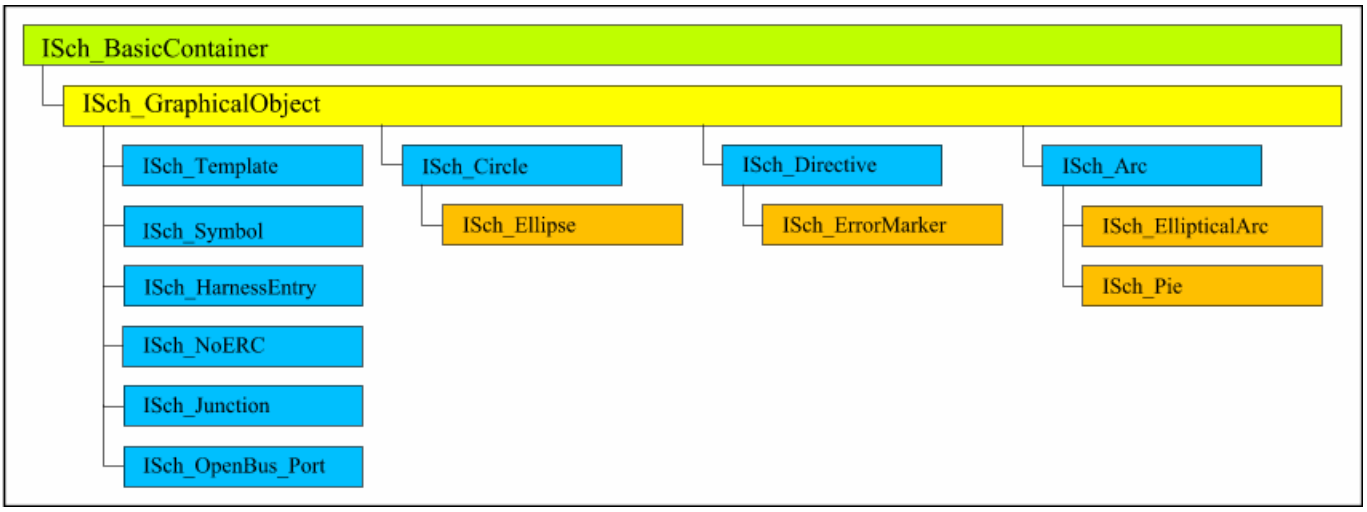
To have access to the Schematic Editor server and manipulate certain schematic design objects, you need to invoke the `SchServer` function which extracts the `ISch_ServerInterface` interface which represents the loaded schematic server in Altium Designer. The `ISch_ServerInterface` interface is the main object interface and contains sub object interfaces within as shown in the diagram below.

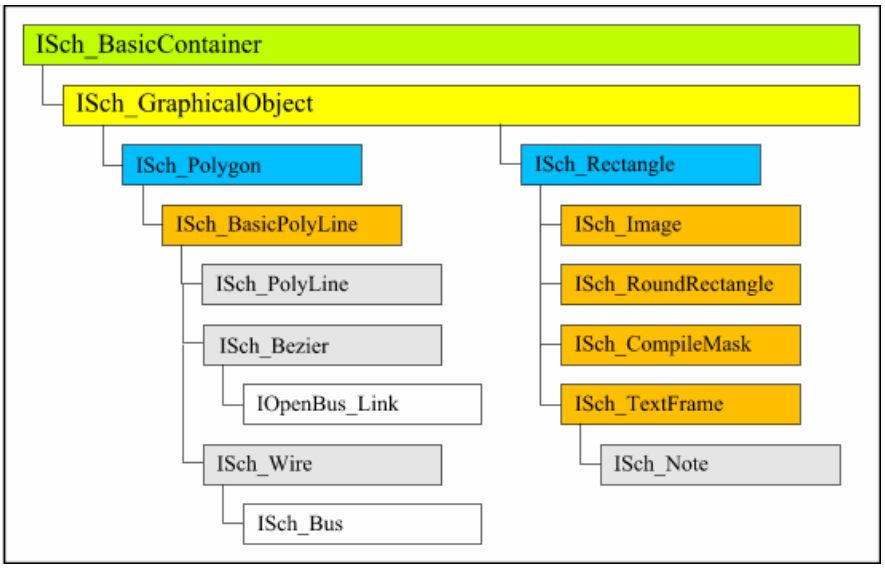


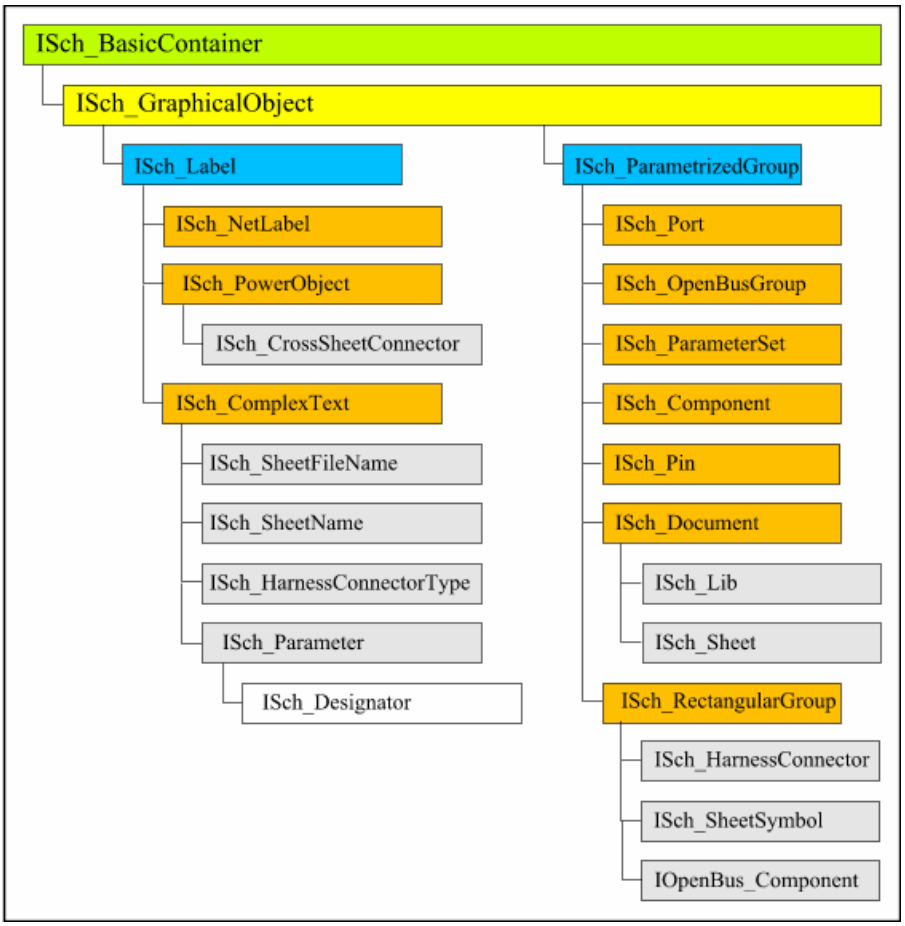
The `ISch_ServerInterface` and `ISch_Document` object interfaces to name the few are the main object interfaces that you will be dealing with, when you are working with a opened schematic document in Altium Designer.

Schematic Object Interfaces Hierarchy Map

The following diagrams represents the hierarchy map of design objects. The ISch_BasicContainer interface is the ancestor object interface. All the descendant interfaces inherit methods and properties from their immediate parent interfaces. For example the ISch_Pie interface has its own methods and properties as well as inherited methods and properties from the `ISch_Arc`, `ISch_GraphicalObject` and finally the ancestor `ISch_BasicContainer` interfaces.






ISch_BasicContainer
ISch_GraphicalObject
ISch_Label
ISch_NetLabel
ISch_PowerObject
ISch_CrossSheetConnector
ISch_ComplexText
ISch_SheetFileName
ISch_SheetName
ISch_HarnessConnectorType
ISch_Parameter
ISch_Designator
ISch_ParametrizedGroup
ISch_Port
ISch_OpenBusGroup
ISch_ParameterSet
ISch_Component
ISch_Pin
ISch_Document
ISch_Lib
ISch_Sheet
ISch_RectangularGroup
ISch_HarnessConnector
ISch_SheetSymbol
IOpenBus_Component

System Interfaces

IConnection Interface

Overview

The `IConnection` interface represents whether the wire or bus connection has a manual junction on it or not, with location, wire or bus objects count and the thickness of wire or bus objects.

The object count denotes the number of connections from this connection location for example one end of a capacitor can have two or more wire connections because it is tied to the Ground as well as to other points on the schematic. A connection that has 3 or more wire / bus objects denotes that a junction (system generated or manually placed) is required to tied the connections together. Thus you can use the `IConnection` interface to determine the number of wire or bus connections at the specified location.

The project that has schematics need to be compiled first before `IConnection` interfaces can be extracted with valid data.

Notes

The `ISch_Sheet` interface has the IConnectionsArray interface which in turn has the IConnection interface.

The `ISch_Document` can be either `ISch_Sheet` or `ISch_Lib` interfaces depending on which document (Schematic Sheet or Schematic Library) you are working with.

A manual junction (placed by an user) may signify a forced connection of at least 3 or more connections on a schematic document.

IConnection Methods and Properties Table

IConnection methods	IConnection properties
GetState_Location	Location
GetState_ObjectsCount	ObjectsCount
GetState_IsManualJunction	IsManualJunction
SetState_Location	
SetState_ObjectsCount	
SetState_IsManualJunction	

See also

IConnectionsArray interface

ISch_Junction interface

ISch_Sheet interface

IConnection GetState and SetState Methods

GetState_Location method

(ISch_Connection interface)

Syntax

```
Function GetState_Location : TLocation;
```

Description

The `GetState_Location` method retrieves the X,Y location of the wire or bus connection on the schematic document. This method is used by the `Location` property.

See also

ISch_Connection interface

Location Property and Example

TLocation type

GetState_ObjectsCount method

(ISch_Connection interface)

Syntax

Function GetState_ObjectsCount

Description

The GetState_ObjectsCount method reports the number of wire or bus connections at a location on the schematic sheet.

See also

ISch_Connection interface

ObjectsCount Property and Example

GetState_Location method

(ISch_Connection interface)

Syntax

```
Function  GetState_IsManualJunction  : Boolean;
```

Description

The `GetState_IsManualJunction` function determines whether the connection has a manual junction or not.

See also

ISch_Connection interface

Location property and example

SetState_Location method

(ISch_Connection interface)

Syntax

```
Procedure SetState_Location (AValue : TLocation);
```

Description

The procedure adds a location to the `IConnection` object.

See also

ISch_Connection interface

SetState_ObjectsCount method

(ISch_Connection interface)

Syntax

```
Procedure SetState_ObjectsCount (AValue : Integer);
```

Description

This procedure sets the objects count for the `IConnection` object.

See also

ISch_Connection interface

SetState_IsManualJunction method

(ISch_Connection interface)

Syntax

```
Procedure SetState_IsManualJunction(AValue : Boolean);
```

Description

This procedure sets the IsManualJunction Boolean setting for the `IConnection` object.

See also

ISch_Connection interface

IConnection Properties

ObjectsCount property

(IConnection interface)

Syntax

```
Property ObjectsCount : Integer Read GetState_ObjectsCount Write SetState_ObjectsCount;
```

Description

This property retrieves or sets the Objects Count for Bus or Wire connection represented by the `IConnection` object.

Example

```
Var
    I,J         : Integer;
    WS          : IWorkspace;
    Prj         : IProject;
    Doc         : IDocument;
    CurrentSch  : ISch_Sheet;
    TheWireConnections : IConnectionsArray;
    WireConnection     : IConnection;
    Connectionslist    : TStringList;
    FileName           : String;
    FilePath           : String;
    ReportDocument     : IServerDocument;
Begin
    WS  := GetWorkspace;
    If WS = Nil Then Exit;
    Prj := WS.DM_FocusedProject;
    If Prj = Nil Then Exit;
    Prj.DM_Compile;
    Doc := WS.DM_FocusedDocument;
    ConnectionsList := TStringList.Create;
    If Doc.DM_DocumentKind = 'SCH' Then
    Begin
         CurrentSch := SchServer.GetSchDocumentByPath(Doc.DM_FullPath);
         If CurrentSch <> Nil Then
         Begin
              TheWireConnections := CurrentSch.WireConnections;
              // Collect data for wire connections (IConnectionArray)
              ConnectionsList.Add('Wire Connections');
              For J := 0 To TheWireConnections.ConnectionsCount - 1 Do
              Begin
                 WireConnection := TheWireConnections.Connection(J);
                 If WireConnection <> Nil Then
                 Begin
                     ConnectionsList.Add('Wire Connection Count: '                + IntToStr
(WireConnection.ObjectsCount));
                     ConnectionsList.Add('Wire Connection Location: '             +
LocationtoStr(WireConnection.Location)); // currently 0,0
                     ConnectionsList.Add('Wire Connection has a manual junction: ' +
BooleantoStr (WireConnection.IsManualJunction));
```

```
                    ConnectionsList.Add('Wire Connection size: '                    + SizeToStr
(WireConnection.Size));
                    ConnectionsList.Add('');
                End;
             End;
          End;
    End;

    FilePath := ExtractFilePath(Doc.DM_FullPath);
    FileName := FilePath + '\ConnectionsReport.Txt';;
    ConnectionsList.SaveToFile(FileName);
    ConnectionsList.Free;

    ReportDocument := Client.OpenDocument('Text', FileName);
    If ReportDocument <> Nil Then
        Client.ShowDocument(ReportDocument);
End;
```

See also

IConnection interface

Location property

(IConnection interface)

Syntax

```
Property Location : TLocation Read GetState_Location Write SetState_Location;
```

Description

This property retrieves or sets the Location of Bus or Wire connection represented by the `IConnection` object.

Example

```
    WS  := GetWorkspace;
    If WS = Nil Then Exit;
    Prj := WS.DM_FocusedProject;
    If Prj = Nil Then Exit;
    Prj.DM_Compile;
    Doc := WS.DM_FocusedDocument;
    If Doc.DM_DocumentKind = 'SCH' Then
    Begin
         CurrentSch := SchServer.GetSchDocumentByPath(Doc.DM_FullPath);
         If CurrentSch <> Nil Then
         Begin
              TheWireConnections := CurrentSch.WireConnections;
              For J := 0 To TheWireConnections.ConnectionsCount - 1 Do
              Begin
                 WireConnection := TheWireConnections.Connection(J);
                 If WireConnection <> Nil Then
                 Begin
                     X := WireConnection.Location.X;
                     Y := WireConnection.Location.Y;
              End;
```

```
        End;
    End;
```

See also

IConnection interface

IsManualJunction property

(IConnection interface)

Syntax

```
Property IsManualJunction : Boolean Read GetState_IsManualJunction Write
SetState_IsManualJunction;
```

Description

This property retrieves or sets the IsManualJunction setting of Bus or Wire connection represented by the `IConnection` object.

Example

```
    WS  := GetWorkspace;
    If WS = Nil Then Exit;
    Prj := WS.DM_FocusedProject;
    If Prj = Nil Then Exit;
    Prj.DM_Compile;
    Doc := WS.DM_FocusedDocument;
    If Doc.DM_DocumentKind = 'SCH' Then
    Begin
         CurrentSch := SchServer.GetSchDocumentByPath(Doc.DM_FullPath);
         If CurrentSch <> Nil Then
         Begin
              TheWireConnections := CurrentSch.WireConnections;
              For J := 0 To TheWireConnections.ConnectionsCount - 1 Do
              Begin
                 WireConnection := TheWireConnections.Connection(J);
                 If WireConnection <> Nil Then
                 Begin
                     ManualJunctionAtConnection := WireConnection.Location.IsManualJunction;
                     //rest of code
              End;
         End;
    End;
```

See also

IConnection interface

IConnectionsArray Interface

Overview

The `IConnectionsArray` represents the bus and wire connections in a schematic document. Bus and wire connections that have more than 3 connections could be connected by an automatic junction or a manual junction (placed by an user).

A schematic with valid buses and wires will have connections. An `IConnectionsArray` interface has all the connections for this schematic sheet and each element in the `IConnectionsArray` interface is a `IConnection` interface type.

IConnectionsArray Methods and Properties Table

IConnectionsArray methods	IConnectionsArray properties

AddConnection
AddConnectionXY
GetConnectionAt
GetState_Connection
GetState_ConnectionsCount
GraphicallyInvalidate
RemoveAllConnectionsAt
RemoveAllConnectionsForLine
ResetAllConnections

ConnectionsCount
Connection

See also

IConnection interface
ISch_Sheet interface

IConnectionsArray Methods

AddConnectionXY method

(IConnectionsArray interface)

Syntax

```
Procedure AddConnectionXY(X, Y : TCoord);
```

Description

This procedure adds a connection with X,Y parameters into the `IConnectionsArray` object.

See also

IConnectionsArray interface
AddConnection method

AddConnection method

(IConnectionsArray interface)

Syntax

```
Procedure AddConnection (ALocation : TLocation);
```

Description

This procedure adds a connection with a location parameter into the `IConnectionsArray` object.

See also

IConnectionsArray interface
AddConnectionXY method

GetConnectionAt method

(IConnectionsArray interface)

Syntax

```
Function GetConnectionAt(ALocation : TLocation) : IConnection;
```

Description

This function retrieves the connection of IConnection type based on the Location parameter.

Example

```
Connection :=  Connections.GetConnectionAt(ALocation);
If Connection <> Nil Then ShowMessage(IntToStr(Connection.ObjectsCount));
```

See also

IConnectionsArray interface

GetState_Connection method

(IConnectionsArray interface)

Syntax

```
Function GetState_Connection(Index : Integer) : IConnection;
```

Description

This function retrieves the indexed connection of IConnection type from the IConnectionsArray interface.

Example

```
For J := 0 To TheBusConnections.GetState_ConnectionsCount - 1 Do
Begin
    BusConnection := TheBusConnections.GetState_Connection(J); //IConnection
    If BusConnection <> Nil Then
    Begin
        // statements here
    End;
End;
```

See also

IConnectionsArray interface

Connection property

GetState_ConnectionsCount method

(IConnectionsArray interface)

Syntax

```
Function GetState_ConnectionsCount : Integer;
```

Description

This function returns the number of connections for wires or buses on the schematic sheet. For each

Example

```
For J := 0 To TheBusConnections.GetState_ConnectionsCount - 1 Do
Begin
    BusConnection := TheBusConnections.GetState_Connection(J); //IConnection
    If BusConnection <> Nil Then
    Begin
        // statements here
    End;
End;
```

See also

IConnectionsArray interface

ConnectionsCount property

GraphicallyInvalidate method

(IConnectionsArray interface)

Syntax

```
Procedure GraphicallyInvalidate;
```

Description

This procedure puts the group of design objects (bus or wire objects in an connection array) in an invalid state. A redraw is required to update the schematic sheet.

Example

```
TheWireConnections.GraphicallyInvalidate;
```

```
// puts the wires part of the connection group in an invalid state that requires a graphical
redraw
```

See also

IConnectionsArray interface

RemoveAllConnectionsAt method

(IConnectionsArray interface)

Syntax

```
Function RemoveAllConnectionsAt(ALocation : TLocation) : Boolean;
```

Description

This function removes all connections at this specified location on the schematic document.

Example

```
If BusConnection.ObjectsCount > 1 Then
    TheBusConnections.RemoveAllConnectionsAt(BusConnection.Location);
// BusConnection = IConnection type, TheBusConnections = IConnectionsArray type
```

See also

IConnectionsArray interface

RemoveAllConnectionsForLine method

(IConnectionsArray interface)

Syntax

```
Function RemoveAllConnectionsForLine(L1, L2 : TLocation) : Boolean;
```

Description

This function removes all connections for the specified line with L1 and L2 parameters. If the call was successful, a true value is returned. The Connections can either represent bus or wire connections.

See also

IConnectionsArray interface

ResetAllConnections method

(IConnectionsArray interface)

Syntax

```
Procedure ResetAllConnections;
```

Description

This procedure resets all connections (frees all items) in the `IConnectionsArray` interface for either wire or bus connections.

Example

```
TheBusConnections.ResetAllConnections;
//TheBusConnections = IConnectionsArray type
```

See also

IConnectionsArray interface

IConnectionsArray Properties

Connection property

(IConnectionsArray interface)

Syntax

```
Property Connection[i : Integer] : IConnection Read GetState_Connection;
```

Description

Example

```
For J := 0 To TheBusConnections.GetState_ConnectionsCount - 1 Do
```

```
Begin
    BusConnection := TheBusConnections.GetState_Connection(J); //IConnection
    If BusConnection <> Nil Then
    Begin
        // statements here
    End;
End;
```

See also

IConnectionsArray interface

ConnectionsCount property

(IConnectionsArray interface)

Syntax

```
Property ConnectionsCount : Integer Read GetState_ConnectionsCount;
```

Description

Example

```
For J := 0 To TheBusConnections.GetState_ConnectionsCount - 1 Do
Begin
    BusConnection := TheBusConnections.GetState_Connection(J); //IConnection
    If BusConnection <> Nil Then
    Begin
        // statements here
    End;
End;
```

See also

IConnectionsArray interface

ISch_Document Interface

Overview

This interface is the immediate ancestor interface for ISch_Sheet and ISch_Lib interfaces.

Notes

You can modify or set the document's preference settings.

You can iterate design objects in a Schematic or library document, see ISch_Iterator interface for details.

You can invoke the `ChooseLocationInteractively` or `ChooseRectangleInteractively` methods to obtain coordinates from the Schematic sheet or library sheet.

You can create a library from a project that has components

You can check whether objects exist on a particular point on a schematic or library document.

Notes

The `ISch_Document` interface hierarchy is as follows;

ISch_BasicContainer

ISch_GraphicalObject

ISch_ParameterizedGroup

ISch_Document

ISch_Document Methods and Properties Table

ISch_Document methods	ISch_Document properties
BoundingRectangle_Selected	BorderOn
ChooseLocationInteractively	CustomMarginWidth
ChooseRectangleInteractively	CustomSheetStyle
CountContextMenuObjects	CustomX
CreateHitTest	CustomXZones
CreateLibraryFromProject	CustomY
Graphical_VirtualRectangle	CustomYZones
LockViewUpdate	DisplayUnit
ObjectReferenceZone	DocumentBorderStyle
PlaceSchComponent	DocumentName
PopupMenuHitTest	HotSpotGridOn
RedrawToDC	HotSpotGridSize
RegisterSchObjectInContainer	InternalTolerance
UnLockViewUpdate	LoadFormat
UnregisterAndFreeAllConnectionLines	ReferenceZonesOn
UnRegisterSchObjectFromContainer	SheetMarginWidth
UpdateDocumentProperties	SheetSizeX
	SheetSizeY
GetState_BorderOn	SheetStyle
GetState_CustomMarginWidth	SheetZonesX
GetState_CustomSheetStyle	SheetZonesY
GetState_CustomX	ShowTemplateGraphics
GetState_CustomXZones	SnapGridOn
GetState_CustomY	SnapGridSize
GetState_CustomYZones	SystemFont
GetState_DocumentBorderStyle	TemplateFileName
GetState_DocumentName	TitleBlockOn
GetState_HotSpotGridOn	UnitSystem
GetState_HotSpotGridSize	UseCustomSheet
GetState_InternalTolerance	VisibleGridOn
GetState_LoadFormat	VisibleGridSize
GetState_ReferenceZonesOn	WorkspaceOrientation
GetState_SheetMarginWidth	
GetState_SheetSizeX	
GetState_SheetSizeY	
GetState_SheetStyle	
GetState_SheetZonesX	
GetState_SheetZonesY	
GetState_ShowTemplateGraphics	
GetState_SnapGridOn	
GetState_SnapGridSize	
GetState_SystemFont	
GetState_TemplateFileName	

GetState_TitleBlockOn
GetState_Unit
GetState_UnitSystem
GetState_UseCustomSheet
GetState_VisibleGridOn
GetState_VisibleGridSize
GetState_WorkspaceOrientation

SetState_BorderOn
SetState_CustomMarginWidth
SetState_CustomSheetStyle
SetState_CustomX
SetState_CustomXZones
SetState_CustomY
SetState_CustomYZones
SetState_DocumentBorderStyle
SetState_HotSpotGridOn
SetState_HotSpotGridSize
SetState_LoadFormat
SetState_ReferenceZonesOn
SetState_SheetMarginWidth
SetState_SheetSizeX
SetState_SheetSizeY
SetState_SheetStyle
SetState_SheetZonesX
SetState_SheetZonesY
SetState_ShowTemplateGraphics
SetState_SnapGridOn
SetState_SnapGridSize
SetState_SystemFont
SetState_TemplateFileName
SetState_TitleBlockOn
SetState_Unit
SetState_UseCustomSheet
SetState_VisibleGridOn
SetState_VisibleGridSize
SetState_WorkspaceOrientation

See also

ISch_Sheet interface
ISch_Lib interface

ISch_Document Methods

BoundingRectangle_Selected method

(ISch_Document interface)

Syntax

```
Function BoundingRectangle_Selected : TCoordRect;
```

Description

The function returns the coordinates of the selected bounding rectangle on the current schematic document.

Example

```
Rect := Sheet.BoundingRectangle_Selected;
MinX := Floor(CoordToMils(Rect.x1));
MinY := Floor(CoordToMils(Rect.y1));
MaxX := Ceil (CoordToMils(Rect.x2));
MaxY := Ceil (CoordToMils(Rect.y2));
```

See also

ISch_Document interface

TCoordRect type

ChooseLocationInteractively method

(ISch_Document interface)

Syntax

```
Function ChooseLocationInteractively(Var ALocation : TLocation; Prompt : TDynamicString) :
Boolean;
```

Description

To monitor the mouse movement and clicks from your script, the `ISch_Document` document interface and its descendant interfaces, `ISch_Lib` and `ISch_Sheet` interfaces has several interactive feedback methods. The `ChooseLocationInteractively` when invoked prompts the user to set the location (point) on the schematic sheet.

The `ChooseLocationinteractively` method can be used to fetch the coordinates of the clicked point on the schematic sheet and can be used for the `ISch_HitTest` interface.

Example

```
    If SchServer = Nil Then Exit;
    CurrentSheet := SchServer.GetCurrentSchDocument;
    If CurrentSheet = Nil Then Exit;

    ALocation := TLocation; //
    //Using the ChooseLocationInteractively method to capture the
    // location's coordinates clicked on the sheet by the user.
    If Not CurrentSheet.ChooseLocationInteractively(ALocation,
                                                    'Please select the location') Then Exit;
```

See also

ISch_Document interface

ISch_HitTest interface

ChooseRectangleInteractively method

(ISch_Document interface)

Syntax

```
Function ChooseRectangleInteractively(Var ARect : TCoordRect;Prompt1 : TDynamicString;Prompt2
: TDynamicString) : Boolean;
```

Description

To monitor the mouse movement and clicks from your script, the `ISch_Document` document interface and its descendant interfaces, `ISch_Lib` and `ISch_Sheet` interfaces has several interactive feedback methods. The `ChooseRectangleinteractively` when invoked prompts the user to set the two corners of the bounding rectangle on the schematic sheet.

The ChooseRectangleinteractively method can be used to fetch the coordinates of the bounding rectangle (of TCoordRect type) for the Spatial iterator where it needs the bounds of a rectangle on the schematic document to search within.

DelphiScript Example

```
Var
    CurrentSheet    : ISch_Document;
    SpatialIterator : ISch_Iterator;
    GraphicalObj    : ISch_GraphicalObject;
    Rect            : TCoordRect;
Begin
    If SchServer = Nil Then Exit;
    CurrentSheet := SchServer.GetCurrentSchDocument;
    If CurrentSheet = Nil Then Exit;
    Rect := TCoordRect;

    If Not CurrentSheet.ChooseRectangleInteractively(Rect,
           'Please select the first corner',
           'Please select the final corner') Then Exit;

    SpatialIterator := CurrentSheet.SchIterator_Create;
    If SpatialIterator = Nil Then Exit;
    Try
        SpatialIterator.AddFilter_ObjectSet(MkSet(eJunction,eSchComponent));
        SpatialIterator.AddFilter_Area(Rect.left, Rect.bottom, Rect.right, Rect.top);
        GraphicalObj := SpatialIterator.FirstSchObject;
        While GraphicalObj <> Nil Do
        Begin
           // do what you want with the design object
           GraphicalObj := SpatialIterator.NextSchObject;
        End;

    Finally
        CurrentSheet.SchIterator_Destroy(SpatialIterator);
    End;
End;
```

See also

ISch_Document interface

TCoordRect type

CountContextMenuObjects method

(ISch_Document interface)

Syntax

```
Function CountContextMenuObjects (AObjectSet : TObjectSet) : Integer;
```

Description

The function counts the contextual objects based on the AObjectSet parameter of TObjectSet type.

Example

```
SchDoc  := SchServer.GetCurrentSchDocument;
```

```
Visible := (SchDoc <> Nil) And (SchDoc.CountContextMenuObjects([eSchComponent]) > 0);
```

DelphiScript Example

```
SchDoc  := SchServer.GetCurrentSchDocument;
ShowMessage(IntToStr(SchDoc.CountContextMenuObjects(MkSet(eSchComponent)) > 0);
// DelphiScript cannot handle sets like Borland Delphi does so we need to use MkSet function.
```

See also

ISch_Document interface

TObjectSet

CreateHitTest method

(ISch_Document interface)

Syntax

```
Function CreateHitTest (ATestMode : THitTestMode;ALocation : TLocation) : ISch_HitTest;
```

Description

The `CreateHitTest` function creates an hit test object which is represented by the `ISch_HitTest` interface with the `ATestMode` and `ALocation` parameters.

With this `ISch_HitTest` interface, the number of objects and the object type at a particular point on the schematic document can be returned.

Example

```
Doc := SchServer.GetCurrentSchDocument;
If Doc = Nil Then Exit;

Doc.ChooseLocationInteractively(ALocation,'Choose a location to click');
AHitTestMode := eHitTest_AllObjects;
AHitTest := Doc.CreateHitTest(AHitTestMode,ALocation);
For I := 0 to AHitTest.HitTestCount - 1 Do
Begin
    APrim := AHitTest.HitObject[I];
    ShowMessage(ObjectIdToString(APrim.ObjectId) + #13 +
                'Location coordinates - '         + #13 +
                ' X= ' + IntToStr(ALocation.X)    + #13 +
                ' Y= ' + IntToSTr(ALocation.Y));
End;
```

See also

ISch_Document interface

ISch_HitTest interface

THitTestMode type

ChooseLocationInteractively method

CreateLibraryFromProject method

(ISch_Document interface)

Syntax

```
Procedure CreateLibraryFromProject (AddLibToProject : Boolean;FileName : WideString; RunQuiet
: Boolean);
```

Description

This procedure creates a schematic library based on the components on a schematic project. If `AddLibToProject` parameter is set to true, then the created library is put in the same project where the components are in. The `RunQuiet` parameter set to true avoids the Information dialog from coming up.

Example

```
CurrentSheet := SchServer.GetCurrentSchDocument;
If (CurrentSheet = Nil) or (CurrentSheet.ObjectID = eSchLib) Then
Begin
    ShowError('Please run the script on a schematic document.');
    Exit;
End;
CurrentSheet.CreateLibraryFromProject(True,'NewLibrary.SchLib',False);
```

See also

ISch_Document interface

Graphical_VirtualRectangle method

(ISch_Document interface)

Syntax

```
Function Graphical_VirtualRectangle : TCoordRect;
```

Description

The function returns the coordinates of TCoordRect type of the virtual rectangle of the graphical window in Altium Designer.

Example

```
Rect := Sheet.Graphical_VirtualRectangle;
MinX := Floor(CoordToMils(PrintRect.x1));
MinY := Floor(CoordToMils(PrintRect.y1));
MaxX := Ceil (CoordToMils(PrintRect.x2));
MaxY := Ceil (CoordToMils(PrintRect.y2));
```

See also

ISch_Document interface

TCoordRect type

LockViewUpdate method

(ISch_Document interface)

Syntax

```
Procedure LockViewUpdate;
```

Description

This procedure prevents the views of Schematic documents and panels from being refreshed or updated. This is especially used in the situations when a component is being created in the Schematic Library Editor. See the `UnLockViewUpdate` procedure.

Example in Delphi Code

```
If SchServer = Nil Then Exit;
If Not Supports (SchServer.GetCurrentSchDocument, ISch_Lib, CurrentLib) Then Exit;

CurrentLib.LockViewUpdate;
CurrentComponent := CurrentLib.CurrentSchComponent;
SimPortMap := '';
SimModel   := CreateSimObject(SimPortMap, ModelName, ModelDescription, FileLocation,
CurrentLib);
CurrentLib.CurrentSchComponent.AddSchObject(SimModel);
CurrentLib.UnLockViewUpdate;
```

See also

ISch_Document interface

UnLockViewUpdate method

ObjectReferenceZone method

(ISch_Document interface)

Syntax

```
Function ObjectReferenceZone(AObject : ISch_BasicContainer): WideString;
```

Description

The function returns the reference zone string for the design object on the schematic sheet. For example, if a sheet entry object is in the vicinity of Reference Zone C (vertically) and 2 (horizontally) for a Standard Style A document then the function will return a 2C for this sheet entry.

Example

```
SchPort.CrossReference := ChangeFileExt(ExtractFileName(ServerDocument.FileName),'') +
                          '[' + SchDocument.ObjectReferenceZone(SchSheetEntry) + ']' ;
```

See also

ISch_Document interface

PlaceSchComponent method

(ISch_Document interface)

Syntax

```
Procedure PlaceSchComponent (ALibraryPath : WideString;ALibRef : WideString;Var SchObject :
TSchObjectHandle);
```

Description

This procedure places a component on a schematic sheet from the schematic library with `ALibraryPath` and `ALibRef` parameters. The object handle of this component is returned.

Example

```
Var
   CurrentSheet : ISch_Document;
   SchObject    : TSchObjectHandle;
   ALibraryPath : WideString;
   ALibRef      : WideString;
Begin
    CurrentSheet := SchServer.GetCurrentSchDocument;
    If (CurrentSheet = Nil) or (CurrentSheet.ObjectID = eSchLib) Then
    Begin
        ShowError('Please run the script on a schematic document.');
        Exit;
    End;

    SchObject    := 0;
    ALibraryPath := 'C:\Program Files\Altium Designer\Examples\Reference Designs\4 Port Serial
Interface\Libraries\4 Port Serial Interface.SchLib';
    ALibRef      := 'Crystal';

    CurrentSheet.PlaceSchComponent (ALibraryPath, ALibRef, SchObject);
    ShowMessage(IntToStr(SchObject));
End;
```

See also

ISch_Document interface

RedrawToDC method

(ISch_Document interface)

Syntax

```
Procedure RedrawToDC(DC : HDC; PrintKind : Integer; PrintWhat : Integer);
```

Description

The DC parameter is a Handle of the canvas (a encapsulation of a device context).

PrintKind is an ordinal value of the TPrintKind type, `TPrintKind = (ePrintKind_FullColor,ePrintKind_GrayScale,ePrintKind_Monochrome);`

PrintWhat is an ordinal value of the TPrintWhat type, `TPrintWhat = (ePrintAllDocuments,ePrintActiveDocument,ePrintSelection,ePrintScreenRegion);`

Example

```
SchLibrary.RedrawToDC(DC, Ord(KindToPrint), Ord(PrinterOptions.PrintWhat));
```

See also

ISch_Document interface

RegisterSchObjectInContainer method

(ISch_Document interface)

Syntax

```
Procedure RegisterSchObjectInContainer (AObject : ISch_BasicContainer);
```

Description

The `RegisterSchObjectInContainer` procedure registers the object of `ISch_BasicContainer` type (including its descendants) in the parent object itself. In this case, the document registers a new design object. For example when you create a new port object, you are required to register the port object in the schematic document.

DelphiScript Example

```
SchPort := SchServer.SchObjectFactory(ePort,eCreate_GlobalCopy);
If SchPort = Nil Then Exit;
SchPort.Location  := Point(MilsToCoord(1000),MilsToCoord(1000));
SchPort.Style     := ePortRight;
SchPort.IOType    := ePortBidirectional;
SchPort.Alignment := eHorizontalCentreAlign;
SchPort.Width     := MilsToCoord(1000);
SchPort.AreaColor := 0;
SchPort.TextColor := $FFFFFF;
SchPort.Name      := 'Test Port';
SchDoc.RegisterSchObjectInContainer(SchPort);
```

See also

ISch_Document interface

UnLockViewUpdate method

(ISch_Document interface)

Syntax

```
Procedure UnLockViewUpdate;
```

Description

This procedure allows the views of Schematic documents and panels from being refreshed or updated after being locked by the `LockViewUpdate` method. This is especially used in the situations when a component is being created in the Schematic Library Editor. See the `LockViewUpdate` procedure.

Example

```
If SchServer = Nil Then Exit;
If Not Supports (SchServer.GetCurrentSchDocument, ISch_Lib, CurrentLib) Then Exit;

CurrentLib.LockViewUpdate;
```

```
CurrentComponent := CurrentLib.CurrentSchComponent;
SimPortMap := '';
SimModel   := CreateSimObject(SimPortMap, ModelName, ModelDescription, FileLocation,
CurrentLib);
CurrentLib.CurrentSchComponent.AddSchObject(SimModel);
CurrentLib.UnLockViewUpdate;
```

See also

ISch_Document interface

LockViewUpdate method

UnRegisterSchObjectFromContainer method

(ISch_Document interface)

Syntax

```
Procedure UnRegisterSchObjectFromContainer (AObject : ISch_BasicContainer);
```

Description

When a schematic object is unregistered from the container, it is explicitly freed and cannot be used again.

Example

See also

ISch_Document interface

UnregisterAndFreeAllConnectionLines method

(ISch_Document interface)

Syntax

```
Procedure UnregisterAndFreeAllConnectionLines;
```

Description

When this procedure is invoked, the connection lines are unregistered and freed from the database associated with the schematic document.

Example

```
SchDoc.UnregisterAndFreeAllConnectionLines;
```

See also

ISch_Document interface

ISch_ConnectionLine interface

UpdateDocumentProperties method

(ISch_Document interface)

Syntax

```
Procedure UpdateDocumentProperties;
```

Description

This method forces an update of the document properties after the properties have been modified programmatically.

Example

```
Document.UpdateDocumentProperties;
```

See also

ISch_Document interface

ISch_Document GetState and SetState Methods

GetState_BorderOn method

(ISch_Document interface)

Syntax

```
Function GetState_BorderOn : Boolean;
```

Description

This BorderOn property determines whether the border on around the outside of the current schematic document will be displayed or not.

The method returns a boolean value whether the Border is displayed or not and is used in the BorderOn property.

Example

See also

ISch_Document interface

GetState_CustomMarginWidth method

(ISch_Document interface)

Syntax

```
Function GetState_CustomMarginWidth : TCoord;
```

Description

The CustomMarginWidth property sets the margin from the bounds of the schematic sheet inwards. This method sets the CustomMarginWidth property.

Notes

The UseCustomSheet property must be set to true before you can massage the attributes for the custom style of the schematic sheet.

Example

See also

ISch_Document interface

TCoord type

GetState_CustomSheetStyle method

(ISch_Document interface)

Syntax

```
Function GetState_CustomSheetStyle : WideString;
```

Description

This property represents custom sheet style property which values can be inherited from one of the standard sheet styles and customized further. This function sets the custom sheet style.

Example

See also

ISch_Document interface

GetState_CustomX method

(ISch_Document interface)

Syntax

```
Function GetState_CustomX : TCoord;
```

Description

The CustomX property determines the width of the custom sheet for the document. This method gets the CustomX value and is used in the CustomX property.

Example

See also

ISch_Document interface

TCoord type

GetState_CustomXZones method

(ISch_Document interface)

Syntax

```
Function GetState_CustomXZones : TCoord;
```

Description

This property determines the number of regions or reference zones that are displayed along the horizontal and vertical borders. The reference zones form a reference grid along the border of your schematic. This reference grid is only for display purposes and does not affect the Snap, Visible or Electrical Grids that are used when placing schematic objects.

This method gets the CustomXZones property.

Example

See also

ISch_Document interface

TCoord type

GetState_CustomY method

(ISch_Document interface)

Syntax

```
Function GetState_CustomY : TCoord;
```

Description

The CustomY property determines the height of the custom sheet for the document. This method gets the CustomY value and is used in the CustomY property.

Example

See also

ISch_Document interface

TCoord type

GetState_CustomYZones method

(ISch_Document interface)

Syntax

```
Function GetState_CustomYZones : TCoord;
```

Description

This property determines the number of regions or reference zones that are displayed along the horizontal and vertical borders. The reference zones form a reference grid along the border of your schematic. This reference grid is only for display purposes and does not affect the Snap, Visible or Electrical Grids that are used when placing schematic objects.

This method sets the CustomYZones property.

Example

See also

ISch_Document interface

TCoord type

GetState_DocumentBorderStyle method

(ISch_Document interface)

Syntax

```
Function GetState_DocumentBorderStyle : TSheetDocumentBorderStyle;
```

Description

The DocumentBorderStyle property determines the current document/border style for the schematic sheet - ANSI or Standard block.

The function gets the current document border style and is used in the DocumentBorderStyle property.

Example

See also

ISch_Document interface

TSheetDocumentBorder style

GetState_DocumentName method

(ISch_Document interface)

Syntax

```
Function GetState_DocumentName : WideString ;
```

Description

The read only DocumentName property determines the schematic document name. This method is used in the DocumentName property.

Example

See also

ISch_Document interface

GetState_HotSpotGridOn method

(ISch_Document interface)

Syntax

```
Function GetState_HotSpotGridOn : Boolean;
```

Description

The electrical grid supports the Schematic Editor's guided wiring feature. When you are moving an electrical object in the workspace, and when it falls within the electrical grid range of another electrical object that you could connect to, the object you are moving will snap to the fixed object and a hot spot or highlight dot will appear. This dot guides you as to where a valid connection can be made. The electrical grid (hot spot) should be set slightly lower than the current snap grid or else it becomes difficult to position electrical objects one snap grid apart.

The procedure gets the boolean value whether the hot spot grid is on or not and is used in the HotSpotGridOn property.

Example

See also

ISch_Document interface

GetState_HotSpotGridSize method

(ISch_Document interface)

Syntax

```
Function GetState_HotSpotGridSize : TCoord;
```

Description

The electrical grid supports the Schematic Editor's guided wiring feature. When you are moving an electrical object in the workspace, and when it falls within the electrical grid range of another electrical object that you could connect to, the object you are moving will snap to the fixed object and a hot spot or highlight dot will appear. This dot guides you as to where a valid connection can be made. The electrical grid (hot spot) should be set slightly lower than the current snap grid or else it becomes difficult to position electrical objects one snap grid apart.

The procedure gets the hot spot grid size and is used in the HotSpotGridSize property.

Example

See also

ISch_Document interface

GetState_InternalTolerance method

(ISch_Document interface)

Syntax

```
Function GetState_InternalTolerance : TCoord;
```

Description

Example

See also

ISch_Document interface

GetState_LoadFormat method

(ISch_Document interface)

Syntax

```
Function GetState_LoadFormat : WideString;
```

Description

Example

See also

ISch_Document interface

GetState_ReferenceZonesOn method

(ISch_Document interface)

Syntax

```
Function GetState_ReferenceZonesOn : Boolean;
```

Description

This property determines the number of regions or reference zones that are displayed along the horizontal and vertical borders. The reference zones form a reference grid along the border of your schematic. This reference grid is only for display purposes and does not affect the Snap, Visible or Electrical Grids that are used when placing schematic objects.

The procedure gets the value whether the reference zones can be displayed or not and is used in the ReferenceZonesOn property.

Example

```
Procedure TurnOffReferenceZones;
Var
    I          : Integer;
    Project    : IProject;
    Doc        : IDocument;
    CurrentSch : ISch_Document;
Begin
    Project := GetWorkspace.DM_FocusedProject;
    If Project = Nil Then Exit;

    For I := 0 to Project.DM_LogicalDocumentCount - 1 Do
    Begin
```

```
        Doc := Project.DM_LogicalDocuments(I);
        If Doc.DM_DocumentKind = 'SCH' Then
        Begin
            CurrentSch := SchServer.GetSchDocumentByPath(Doc.DM_FullPath);
            If (CurrentSch <> Nil) And CurrentSch.GetState_ReferenceZonesOn Then
            Begin
             SchServer.RobotManager.SendMessage(CurrentSch.I_ObjectAddress, c_BroadCast,
SCHM_BeginModify, c_NoEventData);
             CurrentSch.SetState_ReferenceZonesOn(False);
             SchServer.RobotManager.SendMessage(CurrentSch.I_ObjectAddress, c_BroadCast,
SCHM_EndModify , c_NoEventData);
             End;
        End;
    End;
End;
```

See also

ISch_Document interface

GetState_SheetMarginWidth method

(ISch_Document interface)

Syntax

```
Function GetState_SheetMarginWidth : TCoord;
```

Description

The SheetMarginWidth property determines the margin from the bounds of the schematic sheet inwards.

The SheetMarginWidth function gets the width of the sheet margin and is used in the SheetMarginWidth property.

Notes

The UseCustomSheet property must be set to False before you can massage the attributes for the schematic sheet.

Example

See also

ISch_Document interface

GetState_SheetSizeX method

(ISch_Document interface)

Syntax

```
Function GetState_SheetSizeX : TCoord;
```

Description

Example

See also

ISch_Document interface

GetState_SheetSizeY method

(ISch_Document interface)

Syntax

```
Function GetState_SheetSizeY : TCoord;
```

Description

Example

See also

ISch_Document interface

GetState_SheetStyle method

(ISch_Document interface)

Syntax

```
Function GetState_SheetStyle : TSheetStyle;
```

Description

The SheetStyle property determines the document standard style. One of the document sheet styles are A4, Letter and imperial/metric sized sheets.

The procedure obtains the sheet style and is used in the SheetStyle property.

Example

See also

ISch_Document interface

TSheetStyle type

GetState_SheetZonesX method

(ISch_Document interface)

Syntax

```
Function GetState_SheetZonesX : Integer;
```

Description

Example

See also

ISch_Document interface

GetState_SheetZonesY method

(ISch_Document interface)

Syntax

```
Function GetState_SheetZonesY : Integer;
```

Description

Example

See also

ISch_Document interface

GetState_ShowTemplateGraphics method

(ISch_Document interface)

Syntax

```
Function GetState_ShowTemplateGraphics : Boolean;
```

Description

The template is usually placed on the bottom right of the schematic sheet. The template files have a DOT extension and are located in the `\Templates\` folder of Altium Designer software installation.

The procedure determines whether the template graphics can be displayed or not and is used in the ShowTemplateGraphics property.

Example

See also

ISch_Document interface

GetState_SnapGridOn method

(ISch_Document interface)

Syntax

```
Function GetState_SnapGridOn : Boolean;
```

Description

The snap grid is the grid that the cursor is locked to when placing or manipulating objects on the sheet. This grid should be left on at all times except when specifically placing or moving objects that need to be off grid such as text objects. The visible grid is the grid you see on the grid which acts as a visual grid and typically it is set to be the same as or a multiple of the snap grid.

The procedure gets a boolean value whether the SnapGrid is active or not and is used in the SnapGridOn property.

Example

See also

ISch_Document interface

GetState_SnapGridSize method

(ISch_Document interface)

Syntax

```
Function GetState_SnapGridSize : TCoord;
```

Description

The snap grid is the grid that the cursor is locked to when placing or manipulating objects on the sheet. This grid should be left on at all times except when specifically placing or moving objects that need to be off grid such as text objects. The visible grid is the grid you see on the grid which acts as a visual grid and typically it is set to be the same as or a multiple of the snap grid.

The procedure gets the size value of the snap grid and is used in the SnapGridSize property.

Example

See also

ISch_Document interface

GetState_SystemFont method

(ISch_Document interface)

Syntax

```
Function GetState_SystemFont : TCoord;
```

Description

Example

See also

ISch_Document interface

GetState_TemplateFileName method

(ISch_Document interface)

Syntax

```
Function GetState_TemplateFileName : WideString;
```

Description

Example

See also

ISch_Document interface

GetState_TitleBlockOn method

(ISch_Document interface)

Syntax

```
Function GetState_TitleBlockOn : Boolean;
```

Description

Example

See also

ISch_Document interface

GetState_Unit method

(ISch_Document interface)

Syntax

```
Function GetState_Unit : TUnit;
```

Description

This property determines the system unit used for the schematic project. The available imperial units are Mils, inches, DXP default and Auto imperial as well as available metric units which are mm,cm, metres and auto-metric.

Example

See also

ISch_Document interface

TUnit type

GetState_UnitSystem method

(ISch_Document interface)

Syntax

```
Function GetState_UnitSystem : TUnitSystem;
```

Description

Example

See also

ISch_Document interface

GetState_UseCustomSheet method

(ISch_Document interface)

Syntax

```
Function GetState_UseCustomSheet : Boolean;
```

Description

The property determines whether a custom sheet is used instead of a standard sheet. If the UseCustomSheet is true, then the CustomMarginWidth, CustomSheetStyle, CustomX and CustomY properties can be set for this custom sheet property.

This procedure gets the value whether the custom sheet is used instead of a standard sheet and is used in the UseCustomSheet property.

Example

See also

ISch_Document interface

GetState_VisibleGridOn method

(ISch_Document interface)

Syntax

```
Function GetState_VisibleGridOn : Boolean;
```

Description

The electrical grid supports the Schematic Editor's guided wiring feature. When you are moving an electrical object in the workspace, and when it falls within the electrical grid range of another electrical object that you could connect to, the object you are moving will snap to the fixed object and a hot spot or highlight dot will appear. This dot guides you as to where a valid connection can be made. The electrical grid (hot spot) should be set slightly lower than the current snap grid or else it becomes difficult to position electrical objects one snap grid apart.

Example

See also

ISch_Document interface

GetState_VisibleGridSize method

(ISch_Document interface)

Syntax

```
Function GetState_VisibleGridSize : TCoord;
```

Description

Example

See also

ISch_Document interface

GetState_WorkspaceOrientation method

(ISch_Document interface)

Syntax

```
Function GetState_WorkspaceOrientation : TSheetOrientation;
```

Description

Example

See also

ISch_Document interface

SetState_BorderOn method

(ISch_Document interface)

Syntax

```
Procedure SetState_BorderOn (AValue : Boolean);
```

Description

This BorderOn property determines whether the border on around the outside of the current schematic document will be displayed or not.

The method sets a boolean value whether the Border is displayed or not and is used in the BorderOn property.

Example

See also

ISch_Document interface

SetState_CustomMarginWidth method

(ISch_Document interface)

Syntax

```
Procedure SetState_CustomMarginWidth (AValue : TCoord);
```

Description

The CustomMarginWidth property sets the margin from the bounds of the schematic sheet inwards. This method sets the CustomMarginWidth property.

Notes

The UseCustomSheet property must be set to true before you can massage the attributes for the custom style of the schematic sheet.

Example

See also

ISch_Document interface

SetState_CustomSheetStyle method

(ISch_Document interface)

Syntax

```
Procedure SetState_CustomSheetStyle (AValue : WideString);
```

Description

This property represents custom sheet style property which values can be inherited from one of the standard sheet styles and customized further. This method defines the custom sheet style and then can be customized further.

Example

See also

ISch_Document interface

SetState_CustomX method

(ISch_Document interface)

Syntax

```
Procedure SetState_CustomX (AValue : TCoord);
```

Description

The CustomX property sets the width of the custom sheet for the document. This method sets the CustomX value and is used in the CustomX property.

Example

See also

ISch_Document interface

SetState_CustomXZones method

(ISch_Document interface)

Syntax

```
Procedure SetState_CustomXZones (AValue : TCoord);
```

Description

This property determines the number of regions or reference zones that are displayed along the horizontal and vertical borders. The reference zones form a reference grid along the border of your schematic. This reference grid is only for display purposes and does not affect the Snap, Visible or Electrical Grids that are used when placing schematic objects.

This method sets the CustomXZones property.

Example

See also

ISch_Document interface

SetState_CustomY method

(ISch_Document interface)

Syntax

```
Procedure SetState_CustomY (AValue : TCoord);
```

Description

The CustomY property sets the width of the custom sheet for the document. This method sets the CustomY value and is used in the CustomY property.

Example

See also

ISch_Document interface

SetState_CustomYZones method

(ISch_Document interface)

Syntax

```
Procedure SetState_CustomYZones (AValue : TCoord);
```

Description

This property determines the number of regions or reference zones that are displayed along the horizontal and vertical borders. The reference zones form a reference grid along the border of your schematic. This reference grid is only for display purposes and does not affect the Snap, Visible or Electrical Grids that are used when placing schematic objects.

This method sets the CustomYZones property.

Example

See also

ISch_Document interface

SetState_DocumentBorderStyle method

(ISch_Document interface)

Syntax

```
Procedure SetState_DocumentBorderStyle (AValue : TSheetDocumentBorderStyle);
```

Description

The DocumentBorderStyle property determines the current document/border style for the schematic sheet - ANSI or standard blocks.

The function sets the current document border style and is used in the DocumentBorderStyle property.

Example

See also

ISch_Document interface

SetState_HotSpotGridOn method

(ISch_Document interface)

Syntax

```
Procedure SetState_HotSpotGridOn (AValue : Boolean);
```

Description

The electrical grid supports the Schematic Editor's guided wiring feature. When you are moving an electrical object in the workspace, and when it falls within the electrical grid range of another electrical object that you could connect to, the object you are moving will snap to the fixed object and a hot spot or highlight dot will appear. This dot guides you as to where a valid connection can be made. The electrical grid (hot spot) should be set slightly lower than the current snap grid or else it becomes difficult to position electrical objects one snap grid apart.

Example

See also

ISch_Document interface

SetState_HotSpotGridSize method

(ISch_Document interface)

Syntax

```
Procedure SetState_HotSpotGridSize (AValue : TCoord);
```

Description

The electrical grid supports the Schematic Editor's guided wiring feature. When you are moving an electrical object in the workspace, and when it falls within the electrical grid range of another electrical object that you could connect to, the object you are moving will snap to the fixed object and a hot spot or highlight dot will appear. This dot guides you as to where a valid connection can be made. The electrical grid (hot spot) should be set slightly lower than the current snap grid or else it becomes difficult to position electrical objects one snap grid apart.

The procedure sets the hot spot grid size and is used in the HotSpotGridSize property.

Example

See also

ISch_Document interface

HotSpotGridOn method

TCoord type

SetState_LoadFormat method

(ISch_Document interface)

Syntax

```
Procedure SetState_LoadFormat (AValue : WideString);
```

Description

Example

See also

ISch_Document interface

SetState_ReferenceZonesOn method

(ISch_Document interface)

Syntax

```
Procedure SetState_ReferenceZonesOn (AValue : Boolean);
```

Description

This property determines the number of regions or reference zones that are displayed along the horizontal and vertical borders. The reference zones form a reference grid along the border of your schematic. This reference grid is only for display purposes and does not affect the Snap, Visible or Electrical Grids that are used when placing schematic objects.

The procedure sets whether the reference zones can be displayed or not and is used in the ReferenceZonesOn property.

Example

```
Procedure TurnOffReferenceZones;
Var
    I          : Integer;
    Project    : IProject;
    Doc        : IDocument;
    CurrentSch : ISch_Document;
Begin
    Project := GetWorkspace.DM_FocusedProject;
    If Project = Nil Then Exit;

    For I := 0 to Project.DM_LogicalDocumentCount - 1 Do
    Begin
        Doc := Project.DM_LogicalDocuments(I);
        If Doc.DM_DocumentKind = 'SCH' Then
        Begin
            CurrentSch := SchServer.GetSchDocumentByPath(Doc.DM_FullPath);
            If (CurrentSch <> Nil) And CurrentSch.GetState_ReferenceZonesOn Then
            Begin
             SchServer.RobotManager.SendMessage(CurrentSch.I_ObjectAddress, c_BroadCast,
SCHM_BeginModify, c_NoEventData);
             CurrentSch.SetState_ReferenceZonesOn(False);
             SchServer.RobotManager.SendMessage(CurrentSch.I_ObjectAddress, c_BroadCast,
SCHM_EndModify , c_NoEventData);
             End;
        End;
    End;
End;
```

See also

ISch_Document interface

SetState_SheetMarginWidth method

(ISch_Document interface)

Syntax

```
Procedure SetState_SheetMarginWidth (AValue : TCoord);
```

Description

The SheetMarginWidth property determines the margin from the bounds of the schematic sheet inwards.

The SheetMarginWidth procedure sets the width of the sheet margin and is used in the SheetMarginWidth property.

Notes

The UseCustomSheet property must be set to False before you can massage the attributes for the schematic sheet.

Example

See also

ISch_Document interface

SetState_SheetSizeX method

(ISch_Document interface)

Syntax

```
Procedure SetState_SheetSizeX (AValue : TCoord);
```

Description

Example

See also

ISch_Document interface

SetState_SheetSizeY method

(ISch_Document interface)

Syntax

```
Procedure SetState_SheetSizeY (AValue : TCoord);
```

Description

Example

See also

ISch_Document interface

SetState_SheetStyle method

(ISch_Document interface)

Syntax

```
Procedure SetState_SheetStyle (AValue : TSheetStyle);
```

Description

The SheetStyle property determines the document standard style. One of the document sheet styles are A4, Letter and imperial/metric sized sheets.

The procedure defines the sheet style and is used in the SheetStyle property.

Example

See also

ISch_Document interface

SetState_SheetZonesX method

(ISch_Document interface)

Syntax

```
Procedure SetState_SheetZonesX (AValue : Integer);
```

Description

Example

See also

ISch_Document interface

SetState_SheetZonesY method

(ISch_Document interface)

Syntax

```
Procedure SetState_SheetZonesY (AValue : Integer);
```

Description

Example

See also

ISch_Document interface

SetState_ShowTemplateGraphics method

(ISch_Document interface)

Syntax

```
Procedure SetState_ShowTemplateGraphics(AValue : Boolean);
```

Description

The template is usually placed on the bottom right of the schematic sheet. The template files have a DOT extension and are located in the `in the \Templates\` folder of the Altium Designer software installation.

The procedure sets whether the template graphics can be displayed or not and is used in the ShowTemplateGraphics property.

Example

See also

ISch_Document interface

SetState_SnapGridOn method

(ISch_Document interface)

Syntax

```
Procedure SetState_SnapGridOn (AValue : Boolean);
```

Description

The snap grid is the grid that the cursor is locked to when placing or manipulating objects on the sheet. This grid should be left on at all times except when specifically placing or moving objects that need to be off grid such as text objects. The visible grid is the grid you see on the grid which acts as a visual grid and typically it is set to be the same as or a multiple of the snap grid.

The procedure sets a boolean value whether the SnapGrid is active or not and is used in the SnapGridOn property.

Example

See also

ISch_Document interface

SetState_SnapGridSize method

(ISch_Document interface)

Syntax

```
Procedure SetState_SnapGridSize (AValue : TCoord);
```

Description

The snap grid is the grid that the cursor is locked to when placing or manipulating objects on the sheet. This grid should be left on at all times except when specifically placing or moving objects that need to be off grid such as text objects. The visible grid is the grid you see on the grid which acts as a visual grid and typically it is set to be the same as or a multiple of the snap grid.

The procedure sets the size value of the snap grid and is used in the SnapGridSize property.

Example

See also

ISch_Document interface

SetState_SystemFont method

(ISch_Document interface)

Syntax

```
Procedure SetState_SystemFont (AValue : TFontId);
```

Description

Example

See also

ISch_Document interface

SetState_TemplateFileName method

(ISch_Document interface)

Syntax

```
Procedure SetState_TemplateFileName (AValue : WideString);
```

Description

The template filename is the filename of the template that is placed usually on the bottom right of the schematic sheet. The template files have a `DOT` extension and are located in the `\Templates\` folder of the Altium Designer installation.

The procedure sets the template filename and is used in the `TemplateFilename` property.

Example

See also

ISch_Document interface

SetState_TitleBlockOn method

(ISch_Document interface)

Syntax

```
Procedure SetState_TitleBlockOn (AValue : Boolean);
```

Description

Example

See also

ISch_Document interface

SetState_Unit method

(ISch_Document interface)

Syntax

```
Procedure SetState_Unit (AValue : TUnit);
```

Description

This property determines the system unit used for the schematic project. The available imperial units are Mils, inches, DXP default and Auto imperial as well as available metric units which are mm,cm, metres and auto-metric.

This method sets the Unit system and is used in the DisplayUnit property.

Example

See also

ISch_Document interface

TUnit type

SetState_UseCustomSheet method

(ISch_Document interface)

Syntax

```
Procedure SetState_UseCustomSheet (AValue : Boolean);
```

Description

The property determines whether a custom sheet is used instead of a standard sheet. If the UseCustomSheet is true, then the CustomMarginWidth, CustomSheetStyle, CustomX and CustomY properties can be set for this custom sheet property.

This procedure sets whether the custom sheet is used instead of a standard sheet and is used in the UseCustomSheet property.

Example

See also

ISch_Document interface

SetState_VisibleGridOn method

(ISch_Document interface)

Syntax

```
Procedure SetState_VisibleGridOn (AValue : Boolean);
```

Description

Example

See also

ISch_Document interface

SetState_VisibleGridSize method

(ISch_Document interface)

Syntax

```
Procedure SetState_VisibleGridSize (AValue : TCoord);
```

Description

Example

See also

ISch_Document interface

SetState_WorkspaceOrientation method

(ISch_Document interface)

Syntax

```
Procedure SetState_WorkspaceOrientation(AValue : TSheetOrientation);
```

Description

This procedure sets the orientation of the workspace - either as a portrait or as a landscape format.

Example

See also

ISch_Document interface

TSheetOrientation type

ISch_Document Properties

BorderOn property

(ISch_Document interface)

Syntax

```
Property BorderOn : Boolean Read GetState_BorderOn Write SetState_BorderOn;
```

Description

This BorderOn property determines whether the border on around the outside of the current schematic document will be displayed or not.

Example

See also

ISch_Document interface

CustomMarginWidth property

(ISch_Document interface)

Syntax

```
Property CustomMarginWidth : TCoord Read GetState_CustomMarginWidth Write
SetState_CustomMarginWidth;
```

Description

The CustomMarginWidth property sets the margin from the bounds of the schematic sheet inwards. This property is supported by the GetState_CustomMarginWidth and SetState_CustomMarginWidth methods.

Notes

The UseCustomSheet property must be set to true before you can massage the attributes for the custom style of the schematic sheet.

Example

See also

ISch_Document interface

UseCustomSheet property

CustomSheetStyle property

(ISch_Document interface)

Syntax

```
Property CustomSheetStyle : WideString Read GetState_CustomSheetStyle Write
SetState_CustomSheetStyle;
```

Description

This property represents custom sheet style property which values can be inherited from one of the standard sheet styles and customized further.

This property is supported by the GetState_CustomSheetStyle and SetState_CustomSheetStyle methods.

Notes

The UseCustomSheet property must be set to true before you can massage the attributes for the custom style of the schematic sheet.

Example

See also

ISch_Document interface

CustomX property

(ISch_Document interface)

Syntax

```
Property CustomX : TCoord Read GetState_CustomX Write SetState_CustomX;
```

Description

This property sets the width of the custom sheet for the document. This property is supported by the GetState_CustomX and SetState_CustomX methods.

Notes

The UseCustomSheet property must be set to true before you can massage the attributes for the custom style of the schematic sheet.

Example

See also

ISch_Document interface

CustomXZones property

(ISch_Document interface)

Syntax

```
Property CustomXZones : TCoord Read GetState_CustomXZones Write SetState_CustomXZones;
```

Description

This property determines the number of regions or reference zones that are displayed along the horizontal and vertical borders. The reference zones form a reference grid along the border of your schematic. This reference grid is only for display purposes and does not affect the Snap, Visible or Electrical Grids that are used when placing schematic objects.

This property is supported by the GetState_CustomXZones and SetState_CustomXZones methods.

Notes

The UseCustomSheet property must be set to true before you can massage the attributes for the custom style of the schematic sheet.

Example

See also

ISch_Document interface

CustomY property

(ISch_Document interface)

Syntax

```
Property CustomY : TCoord Read GetState_CustomY Write SetState_CustomY;
```

Description

This property sets the height of the custom sheet for the document. This property is supported by the GetState_CustomY and SetState_CustomY methods.

Notes

The UseCustomSheet property must be set to true before you can massage the attributes for the custom style of the schematic sheet.

Example

See also

ISch_Document interface

CustomYZones property

(ISch_Document interface)

Syntax

```
Property CustomYZones : TCoord Read GetState_CustomYZones Write SetState_CustomYZones;
```

Description

This property determines the number of regions or reference zones that are displayed along the horizontal and vertical borders. The reference zones form a reference grid along the border of your schematic. This reference grid is only for display purposes and does not affect the Snap, Visible or Electrical Grids that are used when placing schematic objects.

This property is supported by the GetState_CustomYZones and SetState_CustomYZones methods.

Notes

The UseCustomSheet property must be set to true before you can massage the attributes for the custom style of the schematic sheet.

Example

See also

ISch_Document interface

DocumentBorderStyle property

(ISch_Document interface)

Syntax

```
Property DocumentBorderStyle : TSheetDocumentBorderStyle Read GetState_DocumentBorderStyle
Write SetState_DocumentBorderStyle;
```

Description

The DocumentBorderStyle property determines the current document/border style for the schematic sheet - whether it is a standard or an ANSI title block.

This property is supported by the GetState_DocumentBorderStyle and SetState_DocumentBorderStyle methods.

Example

See also

ISch_Document interface

TSheetDocumentBorderStyle type

DisplayUnit property

(ISch_Document interface)

Syntax

```
Property DisplayUnit : TUnit Read GetState_Unit Write SetState_Unit;
```

Description

This property determines the system unit used for the schematic project. The available imperial units are Mils, inches, DXP default and Auto imperial as well as available metric units which are mm,cm,metres and autometric.

This DisplayUnit property is supported by the GetState_Unit and SetState_Unit methods.

Example

See also

ISch_Document interface

TUnit type

DocumentName property

(ISch_Document interface)

Syntax

```
Property DocumentName : WideString Read GetState_DocumentName;
```

Description

This read only property determines the schematic document name. This property is supported by the GetState_DocumentName;

Example

See also

ISch_Document interface

HotSpotGridOn property

(ISch_Document interface)

Syntax

```
Property HotSpotGridOn : Boolean Read GetState_HotSpotGridOn Write SetState_HotSpotGridOn;
```

Description

The property determines whether the hot spot grid is displayed or not. The electrical grid supports the Schematic Editor's guided wiring feature. When you are moving an electrical object in the workspace, and when it falls within the electrical grid range of another electrical object that you could connect to, the object you are moving will snap to the fixed object and a hot spot or highlight dot will appear. This dot guides you as to where a valid connection can be made. The electrical grid (hot spot) should be set slightly lower than the current snap grid or else it becomes difficult to position electrical objects one snap grid apart.

This property is supported by the GetState_HotSpotGridOn and SetState_HotSpotGridOn methods.

Example

See also

ISch_Document interface

HotSpotGridSize property

(ISch_Document interface)

Syntax

```
Property HotSpotGridSize : TCoord Read GetState_HotSpotGridSize Write
SetState_HotSpotGridSize;
```

Description

The electrical grid supports the Schematic Editor's guided wiring feature. When you are moving an electrical object in the workspace, and when it falls within the electrical grid range of another electrical object that you could connect to, the object you are moving will snap to the fixed object and a hot spot or highlight dot will appear. This dot guides you as to where a valid connection can be made. The electrical grid (hot spot) should be set slightly lower than the current snap grid or else it becomes difficult to position electrical objects one snap grid apart.

The HotSpotGridSize property determines the size of the hot spot (electrical grid) in TCoord units.

Example

See also

ISch_Document interface

HotSpotGridOn

SnapGridOn

SnapGridSize

TCoord type

InternalTolerance property

(ISch_Document interface)

Syntax

```
Property InternalTolerance : TCoord Read GetState_InternalTolerance;
```

Description

Example

See also

ISch_Document interface

LoadFormat property

(ISch_Document interface)

Syntax

```
Property LoadFormat : WideString Read GetState_LoadFormat Write SetState_LoadFormat;
```

Description

Example

See also

ISch_Document interface

PopupMenuHitTest method

(ISch_Document interface)

Syntax

```
Function PopupMenuHitTest : ISch_HitTest;
```

Description

Example

See also

ISch_Document interface

ISch_HitTest interface

ReferenceZonesOn property

(ISch_Document interface)

Syntax

```
Property ReferenceZonesOn : Boolean Read GetState_ReferenceZonesOn Write
SetState_ReferenceZonesOn;
```

Description

This property determines the number of regions or reference zones that are displayed along the horizontal and vertical borders. The reference zones form a reference grid along the border of your schematic. This reference grid is only for display purposes and does not affect the Snap, Visible or Electrical Grids that are used when placing schematic objects.

This property determines whether the reference zones can be displayed or not and is supported by the GetState_ReferenceZonesOn and SetState_ReferenceZonesOn methods.

Example

```
Procedure TurnOffReferenceZones;
Var
    I          : Integer;
    Project    : IProject;
    Doc        : IDocument;
    CurrentSch : ISch_Document;
Begin
```

```
    Project := GetWorkspace.DM_FocusedProject;
    If Project = Nil Then Exit;

    For I := 0 to Project.DM_LogicalDocumentCount - 1 Do
    Begin
        Doc := Project.DM_LogicalDocuments(I);
        If Doc.DM_DocumentKind = 'SCH' Then
        Begin
            CurrentSch := SchServer.GetSchDocumentByPath(Doc.DM_FullPath);
            If (CurrentSch <> Nil) And CurrentSch.ReferenceZonesOn Then
            Begin
             SchServer.RobotManager.SendMessage(CurrentSch.I_ObjectAddress, c_BroadCast,
SCHM_BeginModify, c_NoEventData);
             CurrentSch.ReferenceZonesOn :=  False;
             SchServer.RobotManager.SendMessage(CurrentSch.I_ObjectAddress, c_BroadCast,
SCHM_EndModify , c_NoEventData);
             End;
        End;
    End;
End;
```

See also

ISch_Document interface

SheetMarginWidth property

(ISch_Document interface)

Syntax

```
Property SheetMarginWidth : TCoord Read GetState_SheetMarginWidth Write
SetState_SheetMarginWidth;
```

Description

The SheetMarginWidth property sets the margin from the bounds of the schematic sheet inwards. This property is supported by the GetState_MarginWidth and SetState_MarginWidth methods.

Notes

The UseCustomSheet property must be set to False before you can massage the attributes for the schematic sheet.

Example

See also

ISch_Document interface

SheetStyle property

(ISch_Document interface)

Syntax

```
Property SheetStyle : TSheetStyle Read GetState_SheetStyle Write SetState_SheetStyle;
```

Description

The SheetStyle property determines the document standard style. One of the document sheet styles are A4, Letter and imperial/metric sized sheets.

This property is supported by the GetState_SheetStyle and SetState_SheetStyle methods.

Example

See also

ISch_Document interface

TSheetStyle type

SheetSizeX property

(ISch_Document interface)

Syntax

```
Property SheetSizeX : TCoord Read GetState_SheetSizeX Write SetState_SheetSizeX;
```

Description

The SheetSizeX property defines the width of the sheet. This property is supported by the GetState_SheetSizeX and GetState_SheetSizeX methods.

Example

See also

ISch_Document interface

SheetSizeY method

SheetSizeY property

(ISch_Document interface)

Syntax

```
Property SheetSizeY : TCoord Read GetState_SheetSizeY Write SetState_SheetSizeY;
```

Description

The SheetSizeY property defines the height of the sheet. This property is supported by the GetState_SheetSizeY and GetState_SheetSizeY methods.

Example

See also

ISch_Document interface

SheetZonesX property

(ISch_Document interface)

Syntax

```
Property SheetZonesX : Integer Read GetState_SheetZonesX Write SetState_SheetZonesX;
```

Description

Example

See also

ISch_Document interface

SheetZonesY property

(ISch_Document interface)

Syntax

```
Property SheetZonesY : Integer Read GetState_SheetZonesY Write SetState_SheetZonesY;
```

Description

Example

See also

ISch_Document interface

ShowTemplateGraphics property

(ISch_Document interface)

Syntax

```
Property ShowTemplateGraphics : Boolean Read GetState_ShowTemplateGraphics Write
SetState_ShowTemplateGraphics;
```

Description

The template is usually placed on the bottom right of the schematic sheet. The template files have a `DOT` extension and are located in the `\Templates\` folder of the Altium Designer software installation.

The property determines whether the template graphics are displayed or not.

Example

See also

ISch_Document interface

SnapGridOn property

(ISch_Document interface)

Syntax

```
Property SnapGridOn : Boolean Read GetState_SnapGridOn Write SetState_SnapGridOn;
```

Description

The snap grid is the grid that the cursor is locked to when placing or manipulating objects on the sheet. This grid should be left on at all times except when specifically placing or moving objects that need to be off grid such as text objects. The visible grid is the grid you see on the grid which acts as a visual grid and typically it is set to be the same as or a multiple of the snap grid.

This property is supported by the GetState_SnapGridOn and SetState_SnapGridOn methods.

Example

See also

ISch_Document interface

SnapGridSize property

(ISch_Document interface)

Syntax

```
Property SnapGridSize : TCoord Read GetState_SnapGridSize Write SetState_SnapGridSize;
```

Description

The snap grid is the grid that the cursor is locked to when placing or manipulating objects on the sheet. This grid should be left on at all times except when specifically placing or moving objects that need to be off grid such as text objects. The visible grid is the grid you see on the grid which acts as a visual grid and typically it is set to be the same as or a multiple of the snap grid.

The property defines the snap grid size and is supported by the GetState_SnapGridSize and SetState_SnapGridSize methods.

Example

See also

ISch_Document interface

SystemFont property

(ISch_Document interface)

Syntax

```
Property SystemFont : TFontId Read GetState_SystemFont Write SetState_SystemFont;
```

Description

Example

See also

ISch_Document interface

TFontID type

TemplateFileName property

(ISch_Document interface)

Syntax

```
Property TemplateFileName : WideString Read GetState_TemplateFileName Write
SetState_TemplateFileName;
```

Description

The template filename is the filename of the template that is placed usually on the bottom right of the schematic sheet. The template files have a DOT extension and are located in the `\Templates\` folder of Altium Designer software installation.

This TemplateFileName property is supported by the `GetState_TemplateFileName` and `SetState_TemplateFileName` methods.

Example

See also

ISch_Document interface

ShowTemplateGraphics method

TitleBlockOn property

(ISch_Document interface)

Syntax

```
Property TitleBlockOn : Boolean Read GetState_TitleBlockOn Write SetState_TitleBlockOn;
```

Description

The property determines whether the title block is displayed or not and is supported by the GetState_TitleBlockOn and SetState_TitleBlockOn methods.

Example

See also

ISch_Document interface

DocumentBorderStyle method

VisibleGridOn property

(ISch_Document interface)

Syntax

```
Property VisibleGridOn : Boolean Read GetState_VisibleGridOn Write SetState_VisibleGridOn;
```

Description

Example

See also

ISch_Document interface

UnitSystem property

(ISch_Document interface)

Syntax

```
Property UnitSystem : TUnitSystem Read GetState_UnitSystem;
```

Description

Example

See also

ISch_Document interface

UseCustomSheet property

(ISch_Document interface)

Syntax

```
Property UseCustomSheet : Boolean Read GetState_UseCustomSheet Write SetState_UseCustomSheet;
```

Description

The property determines whether a custom sheet is used instead of a standard sheet. If the UseCustomSheet is true, then the CustomMarginWidth, CustomSheetStyle, CustomX and CustomY properties can be set for this custom sheet property.

The UseCustomSheet property is supported by the GetState_UseCustomSheet and SetState_UseCustomSheet methods.

Example

See also

ISch_Document interface

CustomX property

CustomY property

CustomSheetStyle property

CustomMarginWidth property

VisibleGridSize property

(ISch_Document interface)

Syntax

```
Property VisibleGridSize : TCoord Read GetState_VisibleGridSize Write
SetState_VisibleGridSize;
```

Description

Example

See also

ISch_Document interface

WorkspaceOrientation property

(ISch_Document interface)

Syntax

```
Property WorkspaceOrientation : TSheetOrientation Read GetState_WorkspaceOrientation Write
SetState_WorkspaceOrientation;
```

Description

Example

See also

ISch_Document interface

ISch_Sheet Interface

Overview

The ISch_Sheet interface represents an existing schematic document open in Altium Designer. A schematic document can have bus and wiring connections which are represented by the IConnectionsArray interface.

You can modify or set the document's preference settings.

You can iterate design objects in a Schematic or library document, see ISch_Iterator interface for details.

You can invoke the ChooseLocationInteractively or ChooseRectangleInteractively methods to obtain coordinates from the Schematic sheet or library sheet.

You can create a library from a project that has components

You can check whether objects exist on a particular point on a schematic or library document.

Notes

The ISch_Sheet interface hierarchy is as follows;

ISch_BasicContainer

ISch_GraphicalObject

ISch_ParameterizedGroup

ISch_Document

ISch_Sheet

ISch_Sheet methods

GetState_WireConnections

GetState_BusConnections

OptimizeUseOfPolylines

GetState_HarnessDefinitionsChanged

Reset_HarnessDefinitionsChanged

Raise_HarnessDefinitionsChanged

ISch_Sheet properties

WireConnections

BusConnections

HarnessDefinitionsChanged

See also

ISch_Document interface

ISch_Lib interface

ISch_Sheet Methods

GetState_BusConnections method

(ISch_Sheet interface)

Syntax

```
Function GetState_BusConnections : IConnectionsArray;
```

Description

This function fetches the connections of the busses on a schematic document. This method is used in the BusConnections property.

Example

See also

ISch_Sheet interface

GetState_WireConnections method

(ISch_Sheet interface)

Syntax

```
Function GetState_WireConnections : IConnectionsArray;
```

Description

This function fetches the connections of the wires on a schematic document. This method is used in the WireConnections property.

Example

See also

ISch_Sheet interface

OptimizeUseOfPolylines method

(ISch_Sheet interface)

Syntax

```
Procedure OptimizeUseOfPolylines;
```

Description

This procedure forces the optimal connection of polylines graphically and in the datastructure.

Example

See also

ISch_Sheet interface

GetState_HarnessDefinitionsChanged

(ISch_Sheet interface)

Syntax

```
Function  GetState_HarnessDefinitionsChanged : Boolean;
```

Description

Example

See also

ISch_Sheet interface

Reset_HarnessDefinitionsChanged

(ISch_Sheet interface)

Syntax

```
Procedure Reset_HarnessDefinitionsChanged;
```

Description

Example

See also

ISch_Sheet interface

Raise_HarnessDefinitionsChanged

(ISch_Sheet interface)

Syntax

```
Procedure Raise_HarnessDefinitionsChanged;
```

Description

Example

See also

ISch_Sheet interface

ISch_Sheet Properties

BusConnections property

(ISch_Sheet interface)

Syntax

```
Property BusConnections : IConnectionsArray Read GetState_BusConnections;
```

Description

This property fetches the connections of busses on the schematic document. This property is supported by the GetState_BusConnections method.

Example

See also

ISch_Sheet interface

WireConnections property

(ISch_Sheet interface)

Syntax

```
Property WireConnections : IConnectionsArray Read GetState_WireConnections;
```

Description

This property fetches the connections of wires on the schematic document. This property is supported by the GetState_WireConnections method.

Example

See also

ISch_Sheet interface

HarnessDefinitionsChanged property

(ISch_Sheet interface)

Syntax

```
Property HarnessDefinitionsChanged : Boolean Read GetState_HarnessDefinitionsChanged;
```

Description

This property is supported by the GetState_HarnessDefinitionsChanged method.

Example

See also

ISch_Sheet interface

ISch_Lib Interface

Overview

This interface represents an existing library document open in Altium Designer. A library is composed of library pages and each page represents the symbol (schematic library component).

You can modify or set the document's preference settings.

You can invoke the ChooseLocationInteractively or ChooseRectangleInteractively methods to obtain coordinates from the Schematic sheet or library sheet.

You can check whether objects exist on a particular point on a schematic or library document.

You can iterate design objects in a library document, with the library iterator. This iterator is created by the SchLibIterator_Create function.

You can invoke the LibIsEmpty method to check if the library is empty (ie no symbols in the library) or not.

Notes

Due to the nature of a library document, all symbols (library components) are displayed on their library pages, so you iterate through the library to fetch symbols.

The ISch_Lib interface hierarchy is as follows;

ISch_BasicContainer

ISch_GraphicalObject

ISch_ParameterizedGroup

ISch_Document

ISch_Lib

ISch_Lib methods

AddSchComponent
LibIsEmpty
RemoveSchComponent
Sch_LibraryRuleChecker_Create
Sch_LibraryRuleChecker_Destroy
SchLibIterator_Create
TransferComponentsPrimitivesBackFromEditor
TransferComponentsPrimitivesToEditor

GetState_Current_SchComponent
GetState_CurrentSchComponentDisplayMode
GetState_CurrentSchComponentPartId
GetState_Description
GetState_ShowHiddenPins

SetState_Current_SchComponent
SetState_CurrentSchComponentAddDisplayMode
SetState_CurrentSchComponentAddPart
SetState_CurrentSchComponentDisplayMode
SetState_CurrentSchComponentPartId
SetState_CurrentSchComponentRemoveDisplayMode
SetState_CurrentSchComponentRemovePart
SetState_Description
SetState_ShowHiddenPins

ISch_Lib properties

CurrentSchComponent
Description
ShowHiddenPins

See also

ISch_Iterator interface

ILibCompInfoReader interface

IComponentINfo interface

ISch_Lib Methods

AddSchComponent method

(ISch_Lib interface)

Syntax

```
Procedure AddSchComponent (Const AComponent : ISch_Component);
```

Description

Example

See also

ISch_Lib interface

LibIsEmpty method

(ISch_Lib interface)

Syntax

```
Function LibIsEmpty : Boolean;
```

Description

Example

See also

ISch_Lib interface

SchLibIterator_Create method

(ISch_Lib interface)

Syntax

```
Function SchLibIterator_Create : ISch_Iterator;
```

Description

Example

See also

ISch_Lib interface

RemoveSchComponent method

(ISch_Lib interface)

Syntax

```
Procedure RemoveSchComponent(Const AComponent : ISch_Component);
```

Description

Example

See also

ISch_Lib interface

Sch_LibraryRuleChecker_Create method

(ISch_Lib interface)

Syntax

```
Function Sch_LibraryRuleChecker_Create : ISch_LibraryRuleChecker;
```

Description

Example

See also

ISch_Lib interface

Sch_LibraryRuleChecker_Destroy method

(ISch_Lib interface)

Syntax

```
Procedure Sch_LibraryRuleChecker_Destroy (Var ARuleChecker : ISch_LibraryRuleChecker);
```

Description

Example

See also

ISch_Lib interface

TransferComponentsPrimitivesToEditor method

(ISch_Lib interface)

Syntax

```
Procedure TransferComponentsPrimitivesToEditor;
```

Description

Example

See also

ISch_Lib interface

TransferComponentsPrimitivesBackFromEditor method

(ISch_Lib interface)

Syntax

```
Procedure TransferComponentsPrimitivesBackFromEditor;
```

Description

Example

See also

ISch_Lib interface

GetState_Current_SchComponent method

(ISch_Lib interface)

Syntax

```
Function GetState_Current_SchComponent: ISch_Component;
```

Description

Example

See also

ISch_Lib interface

GetState_CurrentSchComponentDisplayMode method

(ISch_Lib interface)

Syntax

```
Function GetState_CurrentSchComponentDisplayMode : TDisplayMode;
```

Description

Example

See also

ISch_Lib interface

GetState_CurrentSchComponentPartId method

(ISch_Lib interface)

Syntax

```
Function GetState_CurrentSchComponentPartId : Integer;
```

Description

Example

See also

ISch_Lib interface

GetState_Description method

(ISch_Lib interface)

Syntax

```
Function GetState_Description : WideString;
```

Description

Example

See also

ISch_Lib interface

GetState_ShowHiddenPins method

(ISch_Lib interface)

Syntax

```
Function GetState_ShowHiddenPins : Boolean;
```

Description

Example

See also

ISch_Lib interface

SetState_Current_SchComponent method

(ISch_Lib interface)

Syntax

```
Procedure SetState_Current_SchComponent(AValue : ISch_Component);
```

Description

Example

See also

ISch_Lib interface

SetState_CurrentSchComponentAddDisplayMode method

(ISch_Lib interface)

Syntax

```
Procedure SetState_CurrentSchComponentAddDisplayMode;
```

Description

Example

See also

ISch_Lib interface

SetState_CurrentSchComponentAddPart method

(ISch_Lib interface)

Syntax

```
Procedure SetState_CurrentSchComponentAddPart;
```

Description

Example

See also

ISch_Lib interface

SetState_CurrentSchComponentDisplayMode method

(ISch_Lib interface)

Syntax

```
Procedure SetState_CurrentSchComponentDisplayMode(ADisplayMode : TDisplayMode);
```

Description

Example

See also

ISch_Lib interface

SetState_CurrentSchComponentPartId method

(ISch_Lib interface)

Syntax

```
Procedure SetState_CurrentSchComponentPartId(APartId : Integer);
```

Description

Example

See also

ISch_Lib interface

SetState_CurrentSchComponentRemoveDisplayMode method

(ISch_Lib interface)

Syntax

```
Procedure SetState_CurrentSchComponentRemoveDisplayMode;
```

Description

Example

See also

ISch_Lib interface

SetState_CurrentSchComponentRemovePart method

(ISch_Lib interface)

Syntax

```
Procedure SetState_CurrentSchComponentRemovePart;
```

Description

Example

See also

ISch_Lib interface

SetState_Description method

(ISch_Lib interface)

Syntax

```
Procedure SetState_Description (AValue : WideString);
```

Description

Example

See also

ISch_Lib interface

SetState_ShowHiddenPins method

(ISch_Lib interface)

Syntax

```
Procedure SetState_ShowHiddenPins (AValue : Boolean);
```

Description

Example

See also

ISch_Lib interface

Properties

Description property

(ISch_Lib interface)

Syntax

```
Property Description : WideString Read GetState_Description Write SetState_Description;
```

Description

This property gets or sets the `description` of the library document. This property is supported by its GetState_`Description` and SetState_`Description` methods.

Example

See also

ISch_Lib interface

ShowHiddenPins property

(ISch_Lib interface)

Syntax

```
Property ShowHiddenPins : Boolean Read GetState_ShowHiddenPins Write SetState_ShowHiddenPins;
```

Description

This property gets or sets the visible property of hidden pins of the component in the library document. This property is supported by its GetState_ShowHiddenPins and SetState_ShowHiddenPins methods.

Example

See also

ISch_Lib interface

CurrentSchComponent property

(ISch_Lib interface)

Syntax

```
Property CurrentSchComponent : ISch_Component Read GetState_Current_SchComponent Write
SetState_Current_SchComponent;
```

Description

This property gets or sets the component as the current component in the library document. This property is supported by its GetState_CurrentSchComponent and SetState_CurrentSchComponent methods.

Example

See also

ISch_Lib interface

ISch_BasicContainer Interface

Overview

The ISch_BasicContainer interface represents as a parent object or a child object for a schematic object in Altium Designer.

A sheet symbol object for example is a parent object, and its child objects are sheet entries, thus to fetch the sheet entries, you would create an iterator for the sheet symbol and iterate for sheet entry objects.

A schematic document is a parent object as well thus you also create an iterator for this document and iterate for objects on this document.

Notes

ISch_BasicContainer is the ancestor interface object for schematic object interfaces.

ISch_BasicContainer is the ancestor interface object for ISch_MapDefiner and ISch_Implementation interfaces.

ISch_Document is inherited from ISch_BasicContainer and is a container for storing design objects and in turn each design object is inherited from the ISch_BasicContainer interface.

ISch_Iterator fetches design objects which are inherited from the ISch_BasicContainer interface.

ISch_BasicContainer methods

GetState_ObjectId
GetState_SchBasicContainer
GetState_OwnerSchDocument
GetState_Text
GetState_IdentifierString
GetState_DescriptionString
Setstate_Default
SetState_Text

I_ObjectAddress

AddSchObject
AddAndPositionSchObject
RemoveSchObject

SchIterator_Create
SchIterator_Destroy

DeleteAll
FreeAllContainedObjects
Import_FromUser
Replicate

ISch_BasicContainer properties

Container
ObjectId
OwnerDocument

See also

ISch_GraphicalObject interface
ISch_Document interface
ISch_Implementation interface
ISch_MapDefiner interface

ISch_BasicContainer Methods

AddAndPositionSchObject method

(ISch_BasicContainer interface)

Syntax

```
Procedure AddAndPositionSchObject(AObject : ISch_BasicContainer);
```

Description

The AddSchObject procedure adds and positions a child object into the parent object that the AddSchObject is associated with. For example adding sheet entries in a sheet symbol, you would use this method.

Example

See also

ISch_BasicContainer interface

AddSchObject method

AddSchObject method

(ISch_BasicContainer interface)

Syntax

```
Procedure AddSchObject (AObject : ISch_BasicContainer);
```

Description

The AddSchObject procedure adds a child object into the parent object that the AddSchObject is associated with.

DelphiScript **Example**

```
// Create a parameter object and add it to the new pin object.
Try
    SchServer.ProcessControl.PreProcess(SchDoc, '');
    // Add the parameter to the pin with undo stack also enabled
    Param.Name := 'Added Parameter';
    Param.Text := 'Param added to the pin. Press Undo and this will disappear.  Press undo
twice to remove the component';
    Param.Location := Point(InchesToCoord(3), InchesToCoord(2.4));
    Pin.AddSchObject(Param);
    SchServer.RobotManager.SendMessage(Component.I_ObjectAddress, c_BroadCast,
SCHM_PrimitiveRegistration, Param.I_ObjectAddress);
Finally
    SchServer.ProcessControl.PostProcess(SchDoc, '');
End;
```

See also

ISch_BasicContainer interface

DeleteAll method

(ISch_BasicContainer interface)

Syntax

```
Procedure DeleteAll;
```

Description

The DeleteAll procedure removes the contained objects from the container of ISch_BasicContainer type. For example, if you just want to get a list of contained objects, and make small changes to them and then move them to a new container. In this case, you do not want to free and recreate all the contained objects, so you use the DeleteAll method. To have a clean container, you need to call the FreeAllContainedObjects method instead.

Example

See also

ISch_BasicContainer interface

FreeAllContainedObjects method

FreeAllContainedObjects method

(ISch_BasicContainer interface)

Syntax

```
Procedure FreeAllContainedObjects;
```

Description

The FreeAllContainedObjects procedure removes the contained objects from the container of ISch_BasicContainer type and the container ends up clean. To have container that can be reused with the same elements in another container, you need to call the DeleteAll method instead.

Example

See also

ISch_BasicContainer interface

DeleteAll method

GetState_DescriptionString method

(ISch_BasicContainer interface)

Syntax

```
Function GetState_DescriptionString : WideString;
```

Description

This function returns you the description string for this object.

Example

See also

ISch_BasicContainer interface

GetState_IdentifierString method

(ISch_BasicContainer interface)

Syntax

```
Function GetState_IdentifierString : WideString;
```

Description

This function returns you the identifier string.

Example

See also

ISch_BasicContainer interface

GetState_ObjectId method

(ISch_BasicContainer interface)

Syntax

```
Function GetState_ObjectId : TObjectId;
```

Description

The ObjectID property determines what object type the object in question is. For example when iterating for objects on a schematic document, you would want to modify all objects but update the port objects' locations only, thus you check for the object's ObjectId and if it is a ePort type, then take action.

The function retrieves the ObjectId type and this function is used as a getter in the ObjectID property.

DelphiScript **Example**

```
AnObject := Iterator.FirstSchObject;
While AnObject <> Nil Do
Begin
    SchServer.RobotManager.SendMessage(AnObject.I_ObjectAddress, c_BroadCast,
SCHM_BeginModify, c_NoEventData);

    Case AnObject.ObjectId Of
```

```
       eWire   : AnObject.Color     := $0000FF; //red color in bgr format
       ePort   : AnObject.AreaColor := $00FF00; //green color in bgr format
    End;

    SchServer.RobotManager.SendMessage(AnObject.I_ObjectAddress, c_BroadCast, SCHM_EndModify
, c_NoEventData);
    AnObject := Iterator.NextSchObject;
End;
```

See also

ISch_BasicContainer interface

GetState_OwnerSchDocument method

(ISch_BasicContainer interface)

Syntax

```
Function GetState_OwnerSchDocument : ISch_Document;
```

Description

This property returns the ISch_Document interface that the object is associated with. It is also said that the document owns the object when the Object has a valid OwnerDocument property.

The function returns the ISch_Document interface that the object is associated with.

Example

See also

ISch_BasicContainer interface

ISch_Document interface

ISch_GraphicalObject interface

GetState_SchBasicContainer method

(ISch_BasicContainer interface)

Syntax

```
Function GetState_SchBasicContainer : ISch_BasicContainer;
```

Description

This function obtains the container of child objects from the parent object itself. This function is used in the Container property.

Example

See also

ISch_BasicContainer interface

GetState_Text method

(ISch_BasicContainer interface)

Syntax

```
Function GetState_Text : WideString;
```

Description

This function retrieves the text string for this object.

Example

See also

ISch_BasicContainer interface

I_ObjectAddress method

(ISch_BasicContainer interface)

Syntax

```
Function I_ObjectAddress : TSCHObjectHandle;
```

Description

This function retrieves the object address (a pointer type) of the object in question which is of TSchObjectHandle type. This function is mainly used for the SendMessge method from the ISch_RobotManager interface.

DelphiScript **Example**

```
SchServer.RobotManager.SendMessage(AnObject.I_ObjectAddress, c_BroadCast, SCHM_BeginModify,
c_NoEventData);
AnObject.Color     := $0000FF; //red color in bgr format
SchServer.RobotManager.SendMessage(AnObject.I_ObjectAddress, c_BroadCast, SCHM_EndModify  ,
c_NoEventData);
```

See also

ISch_BasicContainer interface

ISch_RobotManager interface

Import_FromUser method

(ISch_BasicContainer interface)

Syntax

```
Function Import_FromUser : Boolean;
```

Description

The Import_FromUser function invokes the Properties dialog for the object. This is equivalent to when you double click on an object on the schematic document and the Object Properties dialog appears. This function returns a True value when the User clicks okay otherwise a False value is returned.

An example of using this method is to pop up the Properties dialog programmatically so that the user can modify the object and then the script or the server code can do more processing.

Example

See also

ISch_BasicContainer interface

RemoveSchObject method

(ISch_BasicContainer interface)

Syntax

```
Procedure RemoveSchObject (AObject : ISch_BasicContainer);
```

Description

The RemoveSchObject method removes the Schematic object from the database associated with the document or the parent object but it is not removed from memory. Therefore an Undo action will be able to restore this object only if the RobotManager's SendMessage methods are invoked.

DelphiScript **Example**

```
// Initialize the robots in Schematic editor.
SchServer.ProcessControl.PreProcess(CurrentSheet, '');

// Set up iterator to look for Port objects only
Iterator := CurrentSheet.SchIterator_Create;
If Iterator = Nil Then Exit;
Iterator.AddFilter_ObjectSet(MkSet(ePort));
```

```
Try
    Port := Iterator.FirstSchObject;
    While Port <> Nil Do
    Begin
        OldPort := Port;
        Port    := Iterator.NextSchObject;
        CurrentSheet.RemoveSchObject(OldPort);

        SchServer.RobotManager.SendMessage(CurrentSheet.I_ObjectAddress,c_BroadCast,
                                SCHM_PrimitiveRegistration,OldPort.I_ObjectAddress);
    End;
Finally
    CurrentSheet.SchIterator_Destroy(Iterator);
End;
// Clean up robots in Schematic editor.
SchServer.ProcessControl.PostProcess(CurrentSheet, '');
```

See also

ISch_BasicContainer interface

Replicate method

(ISch_BasicContainer interface)

Syntax

```
Function Replicate : ISch_BasicContainer;
```

Description

This functions makes another copy of this object but with an unique object address (a new memory location) but with same attributes as this object.

Example

See also

ISch_BasicContainer interface

SchIterator_Create method

(ISch_BasicContainer interface)

Syntax

```
Function SchIterator_Create : ISch_Iterator;
```

Description

The SchIterator_Create function creates an iterator for the parent object (such as the document, component or the sheet symbol) and with this iterator, you have the ability to iterate the child objects within, such as pins of a component. Once you have finished using the iterator, invoke the SchIterator_Destroy method to free the iterator from memory.

Example

```
Try
    SheetSymbol := ParentIterator.FirstSchObject;
    While SheetSymbol <> Nil Do
    Begin
        // Look for sheet entries (child objects) within a sheet symbol object.
        ChildIterator := SheetSymbol.SchIterator_Create;
        If ChildIterator <> Nil Then
        Begin
```

```
            ChildIterator.AddFilter_ObjectSet(MkSet(eSheetEntry));
            Try
                SheetEntry := ChildIterator.FirstSchObject;
                While SheetEntry <> Nil Do
                Begin
                    EntriesNames := SheetEntry.Name + #13 + EntriesNames;
                    SheetEntry := ChildIterator.NextSchObject;
                End;
            Finally
                SheetSymbol.SchIterator_Destroy(ChildIterator);
            End;
        End;
        SheetSymbol := ParentIterator.NextSchObject;
    End;
Finally
    CurrentSheet.SchIterator_Destroy(ParentIterator);
End;
```

See also

ISch_BasicContainer interface

ISch_Iterator interface

SchIterator_Destroy

SchIterator_Destroy method

(ISch_BasicContainer interface)

Syntax

```
Procedure SchIterator_Destroy(Var AIterator : ISch_Iterator);
```

Description

The SchIterator_Destroy function destroys the iterator from the parent object (such as the document, component or the sheet symbol). This iterator once created with the SchIterator_Create method, has the ability to iterate the child objects within, such as pins of a component.

DelphiScript **Example**

```
Try
    SheetSymbol := ParentIterator.FirstSchObject;
    While SheetSymbol <> Nil Do
    Begin
        // Look for sheet entries (child objects) within a sheet symbol object.
        ChildIterator := SheetSymbol.SchIterator_Create;
        If ChildIterator <> Nil Then
        Begin
            ChildIterator.AddFilter_ObjectSet(MkSet(eSheetEntry));
            Try
                SheetEntry := ChildIterator.FirstSchObject;
                While SheetEntry <> Nil Do
                Begin
                    EntriesNames := SheetEntry.Name + #13 + EntriesNames;
                    SheetEntry := ChildIterator.NextSchObject;
                End;
```

```
            Finally
                SheetSymbol.SchIterator_Destroy(ChildIterator);
            End;
        End;
        SheetSymbol := ParentIterator.NextSchObject;
    End;
Finally
    CurrentSheet.SchIterator_Destroy(ParentIterator);
End;
```

See also

ISch_BasicContainer interface

SchIterator_Create;

Setstate_Default method

(ISch_BasicContainer interface)

Syntax

```
Procedure Setstate_Default(AUnit : TUnitSystem);
```

Description

This procedure sets the default unit system for this object.

Example

See also

ISch_BasicContainer interface

TUnitSystem type

SetState_Text method

(ISch_BasicContainer interface)

Syntax

```
Procedure SetState_Text (AValue : WideString);
```

Description

This procedure sets the text string for this object.

Example

See also

ISch_BasicContainer interface

ISch_BasicContainer Properties

Container property

(ISch_BasicContainer interface)

Syntax

```
Property Container : ISch_BasicContainer Read GetState_SchBasicContainer;
```

Description

This property represents the container within the parent object (such as a document, component or sheet symbol). This property is supported by the GetState_SchBasicContainer method. If the container is empty it implies that this object itself is a standalone or child object.

Example

See also

ISch_BasicContainer interface

ObjectId property

(ISch_BasicContainer interface)

Syntax

```
Property ObjectId : TObjectId Read GetState_ObjectId;
```

Description

The ObjectID property determines what object type the object in question is. For example when iterating for objects on a schematic document, you would want to modify all objects but update the port objects' locations only, thus you check for the object's ObjectId and if it is a ePort type, then take action.

DelphiScript **Example**

```
AnObject := Iterator.FirstSchObject;
While AnObject <> Nil Do
Begin
    SchServer.RobotManager.SendMessage(AnObject.I_ObjectAddress, c_BroadCast,
SCHM_BeginModify, c_NoEventData);

    Case AnObject.ObjectId Of
       eWire   : AnObject.Color     := $0000FF; //red color in bgr format
       ePort   : AnObject.AreaColor := $00FF00; //green color in bgr format
    End;

    SchServer.RobotManager.SendMessage(AnObject.I_ObjectAddress, c_BroadCast, SCHM_EndModify
, c_NoEventData);
    AnObject := Iterator.NextSchObject;
End;
```

See also

ISch_BasicContainer interface

TObjectID type

OwnerDocument property

(ISch_BasicContainer interface)

Syntax

```
Property OwnerDocument : ISch_Document Read GetState_OwnerSchDocument;
```

Description

This property returns the ISch_Document interface that the object is associated with. It is also said that the document owns the object when the Object has a valid OwnerDocument property.

This property is supported by the GetState_OwnerSchDocument method.

Example

See also

ISch_BasicContainer interface

ISch_Document interface

ISch_GraphicalObject Interface

Overview

The ISch_GraphicalObject interface represents the ancestor interface for an object that has graphical properties on a schematic document.

All graphic objects such as arcs, ports, rectangles etc have bounding rectangles of TCoordRect type.

Notes

ISch_BasicContainer interface

ISch_GraphicalObject interface

The ISch_GraphicalObject interface hierarchy is as follows;

ISch_GraphicalObject methods

GetState_AreaColor
GetState_Color
GetState_CompilationMasked
GetState_Dimmed
GetState_Disabled
GetState_DisplayError
GetState_EnableDraw
GetState_ErrorColor
GetState_ErrorKind
GetState_ErrorString
GetState_LiveHighlightValue
GetState_Location
GetState_OwnerPartDisplayMode
GetState_OwnerPartId
GetState_Selection
SetState_AreaColor
SetState_Color
SetState_CompilationMasked
SetState_Dimmed
SetState_Disabled
SetState_DisplayError
SetState_EnableDraw
SetState_ErrorColor
SetState_ErrorKind
SetState_ErrorString
SetState_LiveHighlightValue
SetState_Location
SetState_OwnerPartDisplayMode
SetState_OwnerPartId
SetState_Selection

AddErrorString
BoundingRectangle
BoundingRectangle_Full
GraphicallyInvalidate
Mirror
MoveByXY
MoveToXY
ResetErrorFields

ISch_GraphicalObject properties

AreaColor
Color
CompilationMasked
Dimmed
Disabled
DisplayError
EnableDraw
ErrorColor
ErrorKind
ErrorString
LiveHighlightValue
Location
OwnerPartDisplayMode
OwnerPartId
Selection

RotateBy90

SetState_xSizeySize

ISch_GraphicalObject Methods

AddErrorString method

(ISch_GraphicalObject interface)

Syntax

```
Procedure AddErrorString(Const AErrorString : WideString; AtEnd : LongBool);
```

Description

This procedure adds an error string to the string whether it is at end or not.

Example

See also

ISch_GraphicalObject interface

GetState_AreaColor method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_AreaColor : TColor;
```

Description

The AreaColor property denotes the filled color region of a closed object. The AreaColor value is defined as a TColor type from the Borland Delphi's Graphics Unit and has a color range from $00000000 (black) to $00FFFFFF (white).

This method obtains the color for the area color of an object and is used in the AreaColor property.

Example

```
Case AnObject.ObjectId Of
    eWire   : AnObject.Color     := $0000FF; //red color in bgr format
    ePort   : AnObject.AreaColor := $00FF00; //green color in bgr format
End;
```

See also

ISch_GraphicalObject interface

TColor type

GetState_Color method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_Color : TColor;
```

Description

The Color property denotes the color region of a closed object which is usually the border. The Color value is defined as a TColor type from the Borland Delphi's Graphics Unit and has a color range from $00000000 (black) to $00FFFFFF (white).

This method obtains the color for the color of the boundary of an object and is used in the Color property.

Example

```
Case AnObject.ObjectId Of
    eWire   : AnObject.Color     := $0000FF; //red color in bgr format
    ePort   : AnObject.AreaColor := $00FF00; //green color in bgr format
End;
```

See also

ISch_GraphicalObject interface

TColor type

GetState_CompilationMasked method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_CompilationMasked : Boolean;
```

Description

The CompilationMasked property determines whether the object is masked by the Compiler. The CompileMask object can be placed on a group of objects on the schematic sheet, and these objects have their CompilationMasked property set to true.

This method obtains the boolean value whether the CompilationMasked is true or not and is used in the CompilationMasked property.

Example

See also

ISch_GraphicalObject interface

GetState_Dimmed method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_Dimmed : Boolean;
```

Description

This Dimmed property is true when this object is not part of the filter mechanism (by the Filter panel for example). When objects are found by the Filter mechanism, they stay as is (Dimmed is false), and the objects that are not found are dimmed (Dimmed is true).

This procedure gets the boolean value of the Dimmed property and is this method used in the Dimmed property.

Example

See also

ISch_GraphicalObject interface

GetState_Disabled method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_Disabled : Boolean;
```

Description

This Disabled property is true when this object is not part of the filter mechanism (by the Filter panel for example). When objects are found by the Filter mechanism, they stay as is (Disabled is false), and the objects that are not found are disabled (Disabled is true).

Example

See also

ISch_GraphicalObject interface

GetState_DisplayError method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_DisplayError : Boolean;
```

Description

This property determines whether the DisplayError is displayed or not. When true, the red squiggly line underneath the graphical object appears when it is subject to a compilation error in Altium Designer.

This procedure gets the boolean value for the DisplayError property and is used in the DisplayError property.

Example

See also

ISch_GraphicalObject interface

GetState_EnableDraw method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_EnableDraw : Boolean;
```

Description

This property merely determines whether the object can be drawn on the screen or not. This procedure gets the value for the EnableDraw property and is used as a getter for the EnableDraw property.

Example

See also

ISch_GraphicalObject interface

GetState_ErrorColor method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_ErrorColor : TColor;
```

Description

The ErrorColor property determines the error color value that the object is associated with. The Color value is defined as a TColor type from the Borland Delphi's Graphics Unit and has a color range from $00000000 (black) to $00FFFFFF (white).

The function sets the color for the ErrorColor property and is also used as a setter function in the ErrorColor property.

Example

See also

ISch_GraphicalObject interface

GetState_ErrorKind method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_ErrorKind : TErrorKind;
```

Description

This property determines the error kind that the object is associated with, when it is subject to the Compiler in Altium Designer. This procedure is used for the ErrorKind property.

Example

See also

ISch_GraphicalObject interface

GetState_ErrorString method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_ErrorString : WideString;
```

Description

This property returns the Error string that the object is associated with when it is subject to the Compiler in Altium Designer.

This procedure is used for the ErrorString property.

Example

See also

ISch_GraphicalObject interface

GetState_LiveHighlightValue method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_LiveHighlightValue : WideString;
```

Description

This property toggles the highlight value (text string) of the object when it is subject to the probe process in Altium Designer during the Live Design mode. This method is used for the LiveHighlightValue property.

Example

See also

ISch_GraphicalObject interface

GetState_Location method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_Location : TLocation;
```

Description

The Location property defines the reference point of the object (not necessarily the center of the object). Use the BoundingRectangle and BoundingRectangle_Full methods to determine the bounding regions of the object.

This procedure retrieves the location or the reference point of the object. This method is used for the Location property.

Example

See also

ISch_GraphicalObject interface

TLocation type

GetState_OwnerPartDisplayMode method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_OwnerPartDisplayMode : TDisplayMode;
```

Description

This property represents schematic components in various graphical representations only. A schematic component can have up to 255 different graphical representations and a component can be composed of different parts that make up the whole. A child object is part of the parent object and thus the child object's owner part display mode fetches the parent's (in this case the component) part display mode.

This procedure gets the owner display mode (one of the existing modes only) for the component.

Example

See also

ISch_GraphicalObject interface

GetState_OwnerPartId method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_OwnerPartId : Integer;
```

Description

The OwnerPartId property determines the child object's parent object's part id. A component can be composed of multiple parts. Each part is composed of schematic primitives and thus each primitive associated with the part can be queried for its OwnerPartId property. The owner of the child object is the parent object.

This procedure gets the OwnerPartId from the object as part of the component object.

Example

See also

ISch_GraphicalObject interface

GetState_Selection method

(ISch_GraphicalObject interface)

Syntax

```
Function GetState_Selection : Boolean;
```

Description

This property determines whether the object is selected or not. When an object is selected, a crossed line boundary appears around the object. This object can then be moved or edited graphically.

This method can define the selection state of the object and is used for the Selection property.

Example

See also

ISch_GraphicalObject interface

SetState_AreaColor method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_AreaColor (AColor : TColor);
```

Description

The AreaColor property denotes the filled color region of a closed object. The AreaColor value is defined as a TColor type from the Borland Delphi's Graphics Unit and has a color range from $00000000 (black) to $00FFFFFF (white).

This method defines the color for the area color of an object and is used in the AreaColor property.

Example

```
Case AnObject.ObjectId Of
    eWire   : AnObject.Color     := $0000FF; //red color in bgr format
    ePort   : AnObject.AreaColor := $00FF00; //green color in bgr format
End;
```

See also

ISch_GraphicalObject interface

TColor type

SetState_Color method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_Color (AColor : TColor);
```

Description

The Color property denotes the color region of a closed object which is usually the border. The Color value is defined as a TColor type from the Borland Delphi's Graphics Unit and has a color range from $00000000 (black) to $00FFFFFF (white).

This method defines the color for the color of the boundary of an object and is used in the Color property.

Example

```
Case AnObject.ObjectId Of
    eWire   : AnObject.Color     := $0000FF; //red color in bgr format
    ePort   : AnObject.AreaColor := $00FF00; //green color in bgr format
End;
```

See also

ISch_GraphicalObject interface

TColor type

SetState_CompilationMasked method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_CompilationMasked (AValue : Boolean);
```

Description

The CompilationMasked property determines whether the object is masked by the Compiler. The CompileMask object can be placed on a group of objects on the schematic sheet, and these objects have their CompilationMasked property set to true.

This method sets the CompilationMasked to true or not and is used in the CompilationMasked property.

Example

See also

ISch_GraphicalObject interface

SetState_Dimmed method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_Dimmed (B : Boolean);
```

Description

This Dimmed property is true when a parent object is not part of the navigation mechanism (Navigator panel). When objects are found by the Navigation mechanism, they stay as is (Dimmed is false), and the objects that are not part of the Navigation are dimmed (Dimmed is true).

This procedure sets the boolean value of the Dimmed property and is this method used in the Dimmed property.

Example

See also

ISch_GraphicalObject interface

SetState_Disabled method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_Disabled (B : Boolean);
```

Description

This Disabled property is true when this object is not part of the filter mechanism (by the Filter panel for example). When objects are found by the Filter mechanism, they stay as is (Disabled is false), and the objects that are not found are disabled (Disabled is true).

Example

See also

ISch_GraphicalObject interface

SetState_DisplayError method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_DisplayError (AValue : Boolean);
```

Description

This property determines whether the DisplayError is displayed or not. When true, the red squiggly line underneath the graphical object appears when it is subject to a compilation error in Altium Designer.

This procedure sets the boolean value for the DisplayError property and is used in the DisplayError property.

Example

See also

ISch_GraphicalObject interface

SetState_EnableDraw method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_EnableDraw (B : Boolean);
```

Description

This property merely determines whether the object can be drawn on the screen or not. This procedure sets the value for the EnableDraw property and is used as a setter for the EnableDraw property.

Example

See also

ISch_GraphicalObject interface

SetState_ErrorColor method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_ErrorColor (AValue : TColor);
```

Description

The ErrorColor property determines the error color value that the object is associated with.

The Color value is defined as a TColor type from the Borland Delphi's Graphics Unit and has a color range from $00000000 (black) to $00FFFFFF (white).

This procedure obtains the color of the error and this procedure is used as a getter method for the ErrorColor property.

Example

See also

ISch_GraphicalObject interface

SetState_ErrorKind method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_ErrorKind (AValue : TErrorKind);
```

Description

This property determines the error kind that the object is associated with, when it is subject to the Compiler in Altium Designer. This procedure is used for the ErrorKind property.

Example

See also

ISch_GraphicalObject interface

SetState_ErrorString method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_ErrorString (Const AValue : WideString);
```

Description

This property returns the Error string that the object is associated with when it is subject to the Compiler in Altium Designer.

This procedure is used for the ErrorString property.

Example

See also

ISch_GraphicalObject interface

SetState_LiveHighlightValue method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_LiveHighlightValue (AValue : WideString);
```

Description

This property toggles the highlight value (text string) of the object when it is subject to the probe process in Altium Designer during the Live Design mode. This method is used for the LiveHighlightValue property.

Example

See also

ISch_GraphicalObject interface

SetState_Location method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_Location (ALocation : TLocation);
```

Description

The Location property defines the reference point of the object (not necessarily the center of the object). Use the BoundingRectangle and BoundingRectangle_Full methods to determine the bounding regions of the object.

This procedure sets the location or the reference point of the object. This method is used for the Location property.

Example

See also

ISch_GraphicalObject interface

SetState_OwnerPartDisplayMode method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_OwnerPartDisplayMode (AValue : TDisplayMode);
```

Description

This property represents schematic components in various graphical representations only. A schematic component can have up to 255 different graphical representations and a component can be composed of different parts that make up the whole. A child object is part of the parent object and thus the child object's owner part display mode fetches the parent's (in this case the component) part display mode.

This procedure sets the display mode (one of the existing modes only) for the component.

Example

See also

ISch_GraphicalObject interface

ISch_Component interface

SetState_OwnerPartId method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_OwnerPartId (AValue : Integer);
```

Description

The OwnerPartId property determines the child object's parent object's part id. A component can be composed of multiple parts. Each part is composed of schematic primitives and thus each primitive associated with the part can be queried for its OwnerPartId property. The owner of the child object is the parent object.

This procedure sets the OwnerPartId for the object as part of the component object.

Example

See also

ISch_GraphicalObject interface

SetState_Selection method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_Selection (B : Boolean);
```

Description

This property determines whether the object is selected or not. When an object is selected, a crossed line boundary appears around the object. This object can then be moved or edited graphically.

This method can define the selection state of the object and is used for the Selection property.

Example

See also

ISch_GraphicalObject interface

SetState_xSizeySize method

(ISch_GraphicalObject interface)

Syntax

```
Procedure SetState_xSizeySize;
```

Description

This method sets the X size and the ySize of the graphical bounds of the object.

Example

See also

ISch_GraphicalObject interface

BoundingRectangle method

(ISch_GraphicalObject interface)

Syntax

```
Function BoundingRectangle : TCoordRect;
```

Description

This function returns the coordinates of the bounds of the parent object itself (not including the children objects if any). To determine the full bounding rectangle of the object (including the children object), invoke the BoundingRectangle_Full method instead.

For example a Schematic component would typically have a rectangle as the outline, the pins and parameters as the children objects.

Example

See also

ISch_GraphicalObject interface

BoundingRectangle_Full method

TCoordRect type

BoundingRectangle_Full method

(ISch_GraphicalObject interface)

Syntax

```
Function BoundingRectangle_Full : TCoordRect;
```

Description

This function returns the coordinates of the bounds of the parent object itself and including the children objects if any.. To determine the bounding rectangle of the parent object (excluding the children object), invoke the BoundingRectangle method instead.

For example a Schematic component would typically have a rectangle as the outline, the pins and parameters as the children objects.

Example

See also

ISch_GraphicalObject interface

BoundingRectangle method

TCoordRect type

GraphicallyInvalidate method

(ISch_GraphicalObject interface)

Syntax

```
Procedure GraphicallyInvalidate;
```

Description

This procedure when invoked invalidates the object graphically prompting the system to do a system re-draw to refresh the screen.

Example

See also

ISch_GraphicalObject interface

Mirror method

(ISch_GraphicalObject interface)

Syntax

```
Procedure Mirror (Axis : TLocation);
```

Description

The Mirror method flips the object across the axis (TLocaiton Type)

Example

See also

ISch_GraphicalObject interface
ISch_Label interface
ISch_Component interface
TLocation Type

MoveByXY method

(ISch_GraphicalObject interface)

Syntax

```
Procedure MoveByXY (x,y : TCoord);
```

Description

This MoveByXY procedure moves the object in a linear distance specified by the X,Y coordinates relative to the reference point of the object.

Example

```
// Add rectangle and pin objects to the component object.
Component.AddSchObject(Rect);
Component.AddSchObject(Pin);

// Add the new component to the schematic document.
SchDoc.AddSchObject(Component);
Component.Comment.IsHidden := True;
Component.Designator.IsHidden := True;

// Move component by 1,1 inch in respect to document's origin.
Component.MoveByXY(InchesToCoord(1), InchesToCoord(1));
```

See also

ISch_GraphicalObject interface
TCoord type
UndoRedo script example in **Example**s\Scripts\DelphiScript Scripts\Sch folder.

MoveToXY method

(ISch_GraphicalObject interface)

Syntax

```
Procedure MoveToXY (x,y : TCoord);
```

Description

This MoveToXY procedure moves the object to a new location specified by the X,Y coordinates.

Example

```
// Add rectangle and pin objects to the component object.
Component.AddSchObject(Rect);
Component.AddSchObject(Pin);

// Add the new component to the schematic document.
SchDoc.AddSchObject(Component);
Component.Comment.IsHidden := True;
Component.Designator.IsHidden := True;

// Move component to 1,1 inch in respect to document's origin.
Component.MoveToXY(InchesToCoord(1), InchesToCoord(1));
```

See also

ISch_GraphicalObject interface

TCoord type

UndoRedo script example in **Example**s\Scripts\DelphiScript Scripts\Sch folder.

ResetErrorFields method

(ISch_GraphicalObject interface)

Syntax

```
Procedure ResetErrorFields;
```

Description

This procedure resets the error fields of the object.

Example

See also

ISch_GraphicalObject interface

RotateBy90 method

(ISch_GraphicalObject interface)

Syntax

```
Procedure RotateBy90(Center : TLocation; A : TRotationBy90);
```

Description

The RotateBy90 procedure forces the rotation of the object by its center or a defined location on the schematic sheet and the rotation is done in 90 degree increments (0, 90, 180, 270).

Example

See also

ISch_GraphicalObject interface

TLocation type

TRotationBy90 type

ISch_GraphicalObject Properties

AreaColor property

(ISch_GraphicalObject interface)

Syntax

```
Property AreaColor : TColor Read GetState_AreaColor Write SetState_AreaColor;
```

Description

The AreaColor property denotes the filled color region of a closed object. The AreaColor value is defined as a TColor type from the Borland Delphi's Graphics Unit and has a color range from $00000000 (black) to $00FFFFFF (white).

This property is supported by the GetState_AreaColor and SetState_AreaColor methods.

Example

```
Case AnObject.ObjectId Of
    eWire   : AnObject.Color     := $0000FF; //red color in bgr format
    ePort   : AnObject.AreaColor := $00FF00; //green color in bgr format
End;
```

See also

ISch_GraphicalObject interface

ISch_Port interface

ISch_Pie interface

ISch_Rectangle interface

ISch_RoundRectangle interface

ISch_TextFrame interface

Color property

(ISch_GraphicalObject interface)

Syntax

```
Property Color : TColor Read GetState_Color Write SetState_Color;
```

Description

The Color property denotes the color region of a closed object which is usually the border outline. The Color value is defined as a TColor type from the Borland Delphi's Graphics Unit and has a color range from $00000000 (black) to $00FFFFFF (white).

The Color property is supported by the GetState_Color and SetState_Color methods.

Notes

The color format is in blue,green,red (b,g,r) primary color format and each primary color has a value of 0 to 255.

Example

```
Case AnObject.ObjectId Of
    eWire   : AnObject.Color     := $0000FF; //red color in bgr format
    ePort   : AnObject.AreaColor := $00FF00; //green color in bgr format
End;
```

See also

ISch_GraphicalObject interface

TColor type

CompilationMasked property

(ISch_GraphicalObject interface)

Syntax

```
Property CompilationMasked : Boolean Read GetState_CompilationMasked Write
SetState_CompilationMasked;
```

Description

The CompilationMasked property determines whether the object is masked by the Compiler. The CompileMask object can be placed on a group of objects on the schematic sheet, and these objects have their CompilationMasked property set to true.

This property is supported by the GetState_CompilationMasked and SetState_CompilationMasked methods.

Example

See also

ISch_GraphicalObject interface

Dimmed property

(ISch_GraphicalObject interface)

Syntax

```
Property Dimmed : Boolean Read GetState_Dimmed Write SetState_Dimmed;
```

Description

This Dimmed property is true when a parent object is not part of the navigation mechanism (Navigator panel). When objects are found by the Navigation mechanism, they stay as is (Dimmed is false), and the objects that are not part of the Navigation are dimmed (Dimmed is true).

This property is supported by the GetState_Dimmed and SetState_Dimmed methods.

Notes

The Disabled / Dimmed states of a parent object (say a component), all its children (pins, lines, etc...) will be also set to this state. Thus when the Disabled/Dimmed property of a child object is being queried, the Disabled/Dimmed state of the parent object will be returned.

Example

See also

ISch_GraphicalObject interface

Disabled property

(ISch_GraphicalObject interface)

Syntax

```
Property Disabled : Boolean Read GetState_Disabled Write SetState_Disabled;
```

Description

The Disabled property determines whether the object is disabled (due to not being part of the collected objects by the filter mechanism ie the Filter panel)

Notes

The Disabled / Dimmed states of a parent object (say a component), all its children (pins, lines, etc...) will be also set to this state. Thus when the Disabled/Dimmed property of a child object is being queried, the Disabled/Dimmed state of the parent object will be returned.

Example

See also

ISch_GraphicalObject interface

DisplayError property

(ISch_GraphicalObject interface)

Syntax

```
Property DisplayError : Boolean Read GetState_DisplayError Write SetState_DisplayError;
```

Description

This property determines whether the DisplayError is displayed or not. When true, the red squiggly line underneath the graphical object appears when it is subject to a compilation error in Altium Designer.

This property is supported by the GetState_DisplayError and SetState_DisplayError methods.

Example

See also

ISch_GraphicalObject interface

EnableDraw property

(ISch_GraphicalObject interface)

Syntax

```
Property EnableDraw : Boolean Read GetState_EnableDraw Write SetState_EnableDraw;
```

Description

This property merely determines whether the object can be drawn on the screen or not. This property is supported by the GetState_EnableDraw and SetState_EnableDraw methods.

Example

See also

ISch_GraphicalObject interface

ErrorColor property

(ISch_GraphicalObject interface)

Syntax

```
Property ErrorColor : TColor Read GetState_ErrorColor Write SetState_ErrorColor;
```

Description

The ErrorColor property determines the error color value that the object is associated with.

The Color value is defined as a TColor type from the Borland Delphi's Graphics Unit and has a color range from $00000000 (black) to $00FFFFFF (white).

The Color property is supported by the GetState_ErrorColor and SetState_ErrorColor methods.

Example

See also

ISch_GraphicalObject interface

ErrorKind property

(ISch_GraphicalObject interface)

Syntax

```
Property ErrorKind : TErrorKind Read GetState_ErrorKind Write SetState_ErrorKind;
```

Description

This property determines the error kind that the object is associated with, when it is subject to the Compiler in Altium Designer. This property is supported by the GetState_ErrorKind and the SetState_ErrorKind methods.

Example

See also

ISch_GraphicalObject interface

TErrorKind type from Workspace Manager API

ErrorString property

(ISch_GraphicalObject interface)

Syntax

```
Property ErrorString : WideString Read GetState_ErrorString Write SetState_ErrorString;
```

Description

This property returns the Error string that the object is associated with when it is subject to the Compiler in Altium Designer. This property is supported by the GetState_ErrorString and SetState_ErrorString methods.

Example

See also

ISch_GraphicalObject interface

LiveHighlightValue property

(ISch_GraphicalObject interface)

Syntax

```
Property LiveHighlightValue : WideString Read GetState_LiveHighlightValue Write
SetState_LiveHighlightValue;
```

Description

This property toggles the highlight value (text string) of the object when it is subject to the probe process in Altium Designer during the Live Design mode. This property is supported by the GetState_LiveHighlightValue and SetState_LIveHighlightValue methods.

Example

See also

ISch_GraphicalObject interface

Location property

(ISch_GraphicalObject interface)

Syntax

```
Property Location : TLocation Read GetState_Location Write SetState_Location;
```

Description

The Location property defines the reference point of the object (not necessarily the center of the object). Use the BoundingRectangle and BoundingRectangle_Full methods to determine the bounding regions of the object.

This property is supported by the GetState_Location and SetState_Location methods.

Example

See also

ISch_GraphicalObject interface

BoundingRectangle method

BoundingRectangle_Full method

TLocation type

OwnerPartDisplayMode property

(ISch_GraphicalObject interface)

Syntax

```
Property OwnerPartDisplayMode : TDisplayMode Read GetState_OwnerPartDisplayMode Write
SetState_OwnerPartDisplayMode;
```

Description

This property represents schematic components in various graphical representations only. A schematic component can have up to 255 different graphical representations and a component can be composed of different parts that make up the whole. A child object is part of the parent object and thus the child object's owner part display mode fetches the parent's (in this case the component) part display mode.

This property is supported by the GetState_OwnerPartDisplayMode and SetState_OwnerPartDisplayMode methods.

Example

See also

ISch_GraphicalObject interface

ISch_Component interface

TDisplayMode type (byte type) from Workspace Manager API

OwnerPartId property

(ISch_GraphicalObject interface)

Syntax

```
Property OwnerPartId : Integer Read GetState_OwnerPartId Write SetState_OwnerPartId;
```

Description

The OwnerPartId property determines the child object's parent object's part id. A component can be composed of multiple parts. Each part is composed of schematic primitives and thus each primitive associated with the part can be queried for its OwnerPartId property. The owner of the child object is the parent object. This property is supported by the GetState_OwnerPartId and SetState_OwnerPartId methods.

Example

See also

ISch_GraphicalObject interface

Selection property

(ISch_GraphicalObject interface)

Syntax

```
Property Selection : Boolean Read GetState_Selection Write SetState_Selection;
```

Description

This property determines whether the object is selected or not. When an object is selected, a crossed line boundary appears around the object. This object can then be moved or edited graphically.

This property is supported by the GetState_Selection and SetState_Selection methods.

Example

See also

ISch_GraphicalObject interface

ISch_RobotManager Interface

Overview

The ISch_RobotManager interface represents an object that can send Schematic messages into the Schematic Editor server from a script to update the sub-systems such as the Undo system.

Notes

Part of ISch_ServerInterface object interface

MessageID table

```
SCHM_NullMessage             = 0;
SCHM_PrimitiveRegistration   = 1;
SCHM_BeginModify             = 2;
SCHM_EndModify               = 3;
SCHM_YieldToRobots           = 4;
SCHM_CancelModify            = 5;
SCHM_Create                  = 6;
SCHM_Destroy                 = 7;
SCHM_ProcessStart            = 8;
SCHM_ProcessEnd              = 9;
SCHM_ProcessCancel           = 10;
SCHM_CycleEnd                = 11;
SCHM_CycleStart              = 12;
SCHM_SystemInvalid           = 13;
SCHM_SystemValid             = 14;
```

Message types table

```
c_BroadCast     = Nil;
c_NoEventData   = Nil;
c_FromSystem    = Nil;
```

The ISch_RobotManager interface hierarchy is as follows;

ISch_RobotManager methods

SendMessage

ISch_RobotManager properties

See also

ISch_ServerInterface interface

SendMessage method

(ISch_RobotManager interface)

Syntax

```
Procedure SendMessage(Source,Destination : Pointer; MessageID : Word; MessageData : Pointer);
```

Description

The SendMessage method sends a message into Schematic Editor notifying that the data structures need to be updated and synchronized. It could be an object being modified, added or deleted from the schematic document.

Normally when an object is being modified,

The Source parameter, the current sheet's I_ObjectAddress value.

The Destination parameter has the c_Broadcast value

The MessageID parameter has the SchM_PrimitiveRegistration value

The MessageData parameter has the new object's I_ObjectAddress value.

Normally when a new object is being added,

The Source parameter, the I_ObjectAddress of an object needs to be invoked.

The Destination parameter has the c_Broadcast value

The MessageID parameter has the SchM_BeginModify and SchM_EndModify values.

The MessageData parameter has the c_noEventData value

Normally when an object is being removed,

The Source parameter, the current sheet's I_ObjectAddress value.

The Destination parameter normally has the c_Broadcast value

The MessageID parameter has the SchM_PrimitiveRegistration value.

The MessageData parameter has the deleted object's I_ObjectAddress value.

DelphiScript example of an object being modified

```
// Initialize the robots in Schematic editor.
SchServer.ProcessControl.PreProcess(Doc, '');
Iterator        := Doc.SchIterator_Create;
Iterator.AddFilter_ObjectSet(MkSet(ePort, eWire));
If Iterator = Nil Then Exit;
Try
    AnObject := Iterator.FirstSchObject;
    While AnObject <> Nil Do
    Begin
        Case AnObject.ObjectId Of
        SchServer.RobotManager.SendMessage(AnObject.I_ObjectAddress, c_BroadCast,
SCHM_BeginModify, c_NoEventData);
           eWire   : AnObject.Color     := $0000FF; //red color in bgr format
        SchServer.RobotManager.SendMessage(AnObject.I_ObjectAddress, c_BroadCast,
SCHM_EndModify  , c_NoEventData);
        End;
        AnObject := Iterator.NextSchObject;
    End;
Finally
    Doc.SchIterator_Destroy(Iterator);
End;
// Clean up the robots in Schematic editor
SchServer.ProcessControl.PostProcess(Doc, '');
```

DelphiScript example of an object being removed

```
Try
    Port := Iterator.FirstSchObject;
    While Port <> Nil Do
    Begin
        OldPort := Port;
        Port    := Iterator.NextSchObject;
        CurrentSheet.RemoveSchObject(OldPort);

        SchServer.RobotManager.SendMessage
                               (CurrentSheet.I_ObjectAddress,
                                c_BroadCast,
                                SCHM_PrimitiveRegistration,
                                OldPort.I_ObjectAddress);
     End;
Finally
     CurrentSheet.SchIterator_Destroy(Iterator);
End;
```

See also

ISch_RobotManager interface

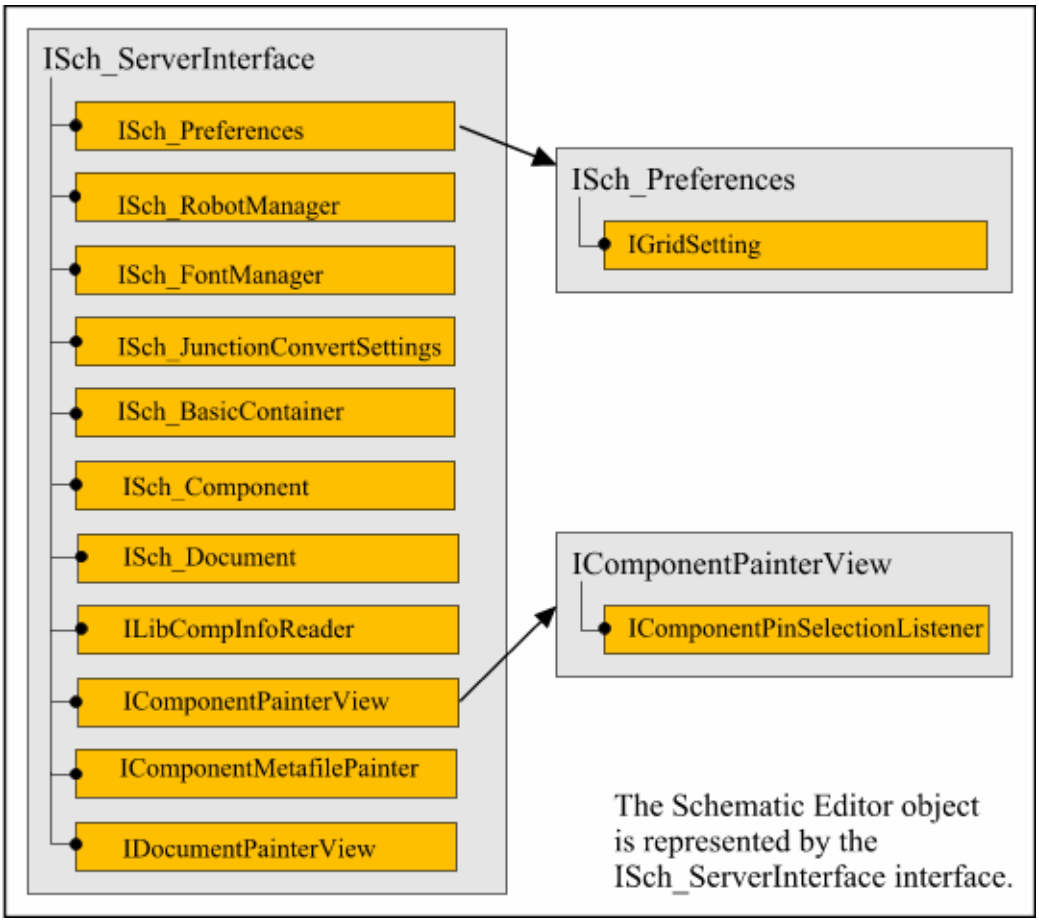
ISch_ServerInterface Interface

Overview

This interface is an entry interface to the schematic server loaded in Altium Designer. You can fetch the Preferences, Robot Manager (for sending messages into the schematic system), the font manager for managing fonts on a schematic document. You can also create or delete schematic design objects from this interface.

The `Sch_Server` function in the `Rt_Schematic` unit (which is embedded in the scripting engine) returns the `ISch_ServerInterface` interface.

The ISch_ServerInterface as the composite interface has the following aggregate object interfaces:



Example

```
    // Grab current schematic document.
    SchDoc := SchServer.GetCurrentSchDocument;
    If SchDoc = Nil Then Exit;

    // Component is a container that has child objects
    // Create component, and its rectangle, pin and parameter objects.
    Component := SchServer.SchObjectFactory (eSchComponent, eCreate_Default);
```

Example 2

```
   Try
       SchServer.ProcessControl.PreProcess(SchDoc, '');

       // Add the parameter to the pin with undo stack also enabled
       Param.Name := 'Added Parameter';
       Param.Text := 'Param added to the pin. Press Undo and this will disappear.  Press undo
twice to remove the component';
       Param.Location := Point(InchesToCoord(3), InchesToCoord(2.4));

       Pin.AddSchObject(Param);
       SchServer.RobotManager.SendMessage(Component.I_ObjectAddress, c_BroadCast,
SCHM_PrimitiveRegistration, Param.I_ObjectAddress);
   Finally
       SchServer.ProcessControl.PostProcess(SchDoc, '');
   End;
```

Notes

Note that these `IServerModule` interfaces represent loaded servers in Altium Designer. This application manages single instances of different server modules. Each server can have multiple server document kinds, for example the Schematic server supports two server document kinds – SCH and SCHLIB design documents. A loaded server typically hosts documents and each document in turn hosts a document view and panel views. Thus a Schematic Editor server also has the `IServerModule` interface along with the `ISch_ServerInterface` interface.

Invoke the `SchServer` function to obtain `the ISch_ServerInterface` object interface which represents the Schematic Editor server.

ISch_ServerInterface methods

GetState_SchPreferences
GetState_RobotManager
GetState_FontManager
GetState_ProbesTimerEnabled
SetState_ProbesTimerEnabled
GetState_JunctionConvertSettings

GetSchDocumentByPath
GetCurrentSchDocument
SchObjectFactory
LoadComponentFromLibrary
LoadComponentFromDatabaseLibrary
DestroySchObject
ReportSchObjectsDifferences
CreateLibCompInfoReader
DestroyCompInfoReader
CreateComponentPainter
CreateComponentMetafilePainter
CreateDocumentPainter
UpdateSignalValueDisplay

ISch_ServerInterface properties

Preferences
RobotManager
FontManager
JunctionConvertSettings
ProbesTimerEnabled

Example

See also

Sch_Server function
ISch_Preferences interface
ISch_RobotManager interface
ISch_FontManager interface
ILibCompInfoReader interface
IServerModule interface

ISch_ServerInterface Methods

CreateComponentMetafilePainter method

(ISch_ServerInterface interface)

Syntax

```
Function CreateComponentMetafilePainter : IComponentMetafilePainter;
```

Description

Example

See also

ISch_ServerInterface interface

IComponentMetafilePainter interface

CreateComponentPainter method

(ISch_ServerInterface interface)

Syntax

```
Function CreateComponentPainter : IComponentPainterView;
```

Description

A IComponentPainterView interface represents the surface that a component can be painted on.

This interface is a IExternalForm type which represents the TExternalFormComponent object. The TExternalForm class is defined in the ExternalForm unit from the DXP Run Time Library.

Notes

This IComponentPainterView interface is not supported in the scripting system.

This IComponentPainterView interface is for server development purposes and you need to have RT_IntegratedLIbrary, RT_Schematic, ExternalForms and the RT_ClientServerINterfaces units in a server project.

Example

See also

ISch_ServerInterface interface

IComponentPainterView interface

CreateDocumentPainter method

(ISch_ServerInterface interface)

Syntax

```
Function CreateDocumentPainter : IDocumentPainterView;
```

Description

This function retrieves the IDocumentPainterView interface that represents the Mini Viewer object in the Schematic Editor.

Example

See also

ISch_ServerInterface interface

IDocumentPainterView interface

CreateLibCompInfoReader method

(ISch_ServerInterface interface)

Syntax

```
Function CreateLibCompInfoReader (ALibFileName : WideString) : ILibCompInfoReader;
```

Description

The function returns a ILibCompInfoReader interface that represents a library component information reader object.

Invoke the CreateLibCompInfoReader function with the path to a schematic library and to obtain the number of components in this library, invoke the ILibCompInfoReader.NumComponentsInfos method and then to obtain the information for each component in this library invoke the ComponentInfos[] method. When you are done, invoke the DestroyCompInfoReader method.

DelphiScript **Example**

```
Procedure LibraryCompInfoReader;
Var
```

```
    CurrentLib     : ISch_Lib;
    ALibCompReader : ILibCompInfoReader;
    CompInfo       : IComponentInfo;
    FileName       : String;
    CompNum, J     : Integer;
    ReportInfo     : TStringList;
    Document       : IServerDocument;
Begin
    If SchServer = Nil Then Exit;
    CurrentLib := SchServer.GetCurrentSchDocument;
    If CurrentLib = Nil Then Exit;
    // CHeck if CurrentLib is a Library document or not
    If CurrentLib.ObjectID <> eSchLib Then
    Begin
         ShowError('Please open schematic library.');
         Exit;
    End;
    FileName := CurrentLib.DocumentName;
    // Set up Library Component Reader object.
    ALibCompReader := SchServer.CreateLibCompInfoReader(FileName);
    If ALibCompReader = Nil Then Exit;
    ALibCompReader.ReadAllComponentInfo;
    ReportInfo := TStringList.Create;
    // Obtain the number of components in the specified sch library.
    CompNum := ALibCompReader.NumComponentInfos;
    // Go thru each component obtained by the LibCompReader interface.
    For J := 0 To CompNum - 1 Do
    Begin
        ReportInfo.Add(FileName);
        CompInfo := ALibCompReader.ComponentInfos[J];
        ReportInfo.Add(' Name : '         + CompInfo.CompName);
        ReportInfo.Add('  Alias Name : '  + CompInfo.AliasName);
        ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.PartCount));
        ReportInfo.Add('  Description : ' + CompInfo.Description);
        ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.Offset));
        ReportInfo.Add('');
    End;
    SchServer.DestroyCompInfoReader(ALibCompReader);

    ReportInfo.Add('');
    ReportInfo.Insert(0,'Schematic Libraries and Their Components Report');
    ReportInfo.Insert(1,'-----------------------------------------');
    ReportInfo.Insert(2,'');
    ReportInfo.SaveToFile('C:\SchLibCompReport.txt');
    // Open and display the Component data in DXP.
    If Client = Nil Then Exit;
```

```
    Document := Client.OpenDocument('Text','c:\SchLibCompReport.txt');
    If Document <> Nil Then
        Client.ShowDocument(Document);
    ReportInfo.Free;
End;
```

See also

ISch_ServerInterface interface

ILibCompInfoReader interface

DestroyCompInfoReader method

(ISch_ServerInterface interface)

Syntax

```
Procedure DestroyCompInfoReader (Var ALibCompReader : ILibCompInfoReader);
```

Description

The function destroys an library component information reader object that is represented by the ILibCompInfoReader interface.

Example

See also

ISch_ServerInterface interface

CreateLibCompInfoReader method

ILibCompInfoReader interface

GetCurrentSchDocument method

(ISch_ServerInterface interface)

Syntax

```
Function GetCurrentSchDocument : ISch_Document;
```

Description

This function returns the ISch_Document interface that represents the current schematic document open in Altium Designer.

Example

See also

ISch_ServerInterface interface

ISch_Document interface

GetSchDocumentByPath method

(ISch_ServerInterface interface)

Syntax

```
Function GetSchDocumentByPath(APath : WideString) : ISch_Document;
```

Description

Example

See also

ISch_ServerInterface interface

GetState_FontManager method

(ISch_ServerInterface interface)

Syntax

```
Function GetState_FontManager : ISch_FontManager;
```

Description

This function retrieves the ISch_Font interface which represents the Font Manager object in the Schematic Editor.

Example

See also

ISch_ServerInterface interface

ISch_Font interface

GetState_JunctionConvertSettings method

(ISch_ServerInterface interface)

Syntax

```
Function GetState_JunctionConvertSettings : ISch_JunctionConvertSettings;
```

Description

The JunctionConvertSettings property represents a crossing of wiring on a schematic sheet. When an addition of a wire would create a four-way junction, this is converted to into two adjacent three way junctions. If it is disabled and when a four way junction is created, the two wires crossing at the intersection are not joined electrically and if the Display Cross Overs option is enabled, a cross over is shown on this intersection.

This property is supported by the GetState_JunctionConvertSettings method.

Example

See also

ISch_ServerInterface interface

GetState_ProbesTimerEnabled method

(ISch_ServerInterface interface)

Syntax

```
Function GetState_ProbesTimerEnabled : Boolean;
```

Description

The ProbesTimerEnabled property determines whether the Probes are active or not. This feature is used in the LiveDesign process in Altium Designer.

This property is supported by the GetState_ProbesTimerEnabled and SetState_ProbesTimerEnabled methods.

Example

See also

ISch_ServerInterface interface

GetState_RobotManager method

(ISch_ServerInterface interface)

Syntax

```
Function GetState_RobotManager : ISch_RobotManager;
```

Description

The RobotManager property returns the ISch_RobotManager interface. This interface deals with sending Schematic notification messages in the system. To have the ability to send a specific message when a specific event in the Schematic Editor occurs can be achieved with the ISch_RobotManager interface.

This property is supported by the GetState_RobotManager method.

Example

See also

ISch_ServerInterface interface

GetState_SchPreferences method

(ISch_ServerInterface interface)

Syntax

```
Function GetState_SchPreferences : ISch_Preferences;
```

Description

The Preferences property retrieves the ISch_Preferences interface which represents the Preferences object for the Schematic Editor.

This read only property is supported by the GetState_SchPreference method.

Example

See also

ISch_ServerInterface interface

LoadComponentFromLibrary method

(ISch_ServerInterface interface)

Syntax

```
Function LoadComponentFromLibrary(ALibReference : WideString;ALibraryName : WideString) :
ISch_Component;
```

Description

Example

See also

ISch_ServerInterface interface

LoadComponentFromDatabaseLibrary method

(ISch_ServerInterface interface)

Syntax

```
Function  LoadComponentFromDatabaseLibrary(ALibraryName       : WideString;
                                           ADatabaseTableName : WideString;
                                           ADatabaseKeys      : WideString) : ISch_Component;
```

Description

Example

See also

ISch_ServerInterface interface

ReportSchObjectsDifferences method

(ISch_ServerInterface interface)

Syntax

```
Function ReportSchObjectsDifferences(Const AObject1, AObject2 :
ISch_BasicContainer;AIgnoreSpatialAttributes : Boolean;ADiffDescription : PChar) : Integer;
```

Description

Example

See also

ISch_ServerInterface interface

SchObjectFactory method

(ISch_ServerInterface interface)

Syntax

```
Function SchObjectFactory(AObjectId : TObjectId;ACreationMode : TObjectCreationMode) :
ISch_BasicContainer;
```

Description

The `SchObjectFactory` function creates a new object based on `TObjectID` and `TObjectCreationMode` values.

When you wish to create a new design object with the `ISch_ServerInterface`'s `SchObjectFactory` method, you will need to have a specific design object type, assign this object with new attribute values and register this object with in the schematic document with the ISch_Document's RegisterSchObjectInContainer method.

Example

```
Var
    SchPort     : ISch_Port;
    FSchDoc     : ISch_Document;
    CurView     : IServerDocumentView;
Begin
    // Check if Schematic server exists or not.
    If SchServer = Nil Then Exit;

    // Obtain the Schematid sheet interfac.e
    FSchDoc := SchServer.GetCurrentSchDocument;
    If FSchDoc = Nil Then Exit;

    // Create a new port object
    SchPort := SchServer.SchObjectFactory(ePort,eCreate_GlobalCopy);
    If SchPort = Nil Then Exit;

    // Set up parameters for the port object.
    // the port is placed at 500,500 mils respectively.
    SchPort.Location  := Point(MilsToCoord(500),MilsToCoord(500));
    SchPort.Style     := ePortRight;
    SchPort.IOType    := ePortBidirectional;
    SchPort.Alignment := eHorizontalCentreAlign;

    SchPort.Width     := MilsToCoord(1000);

    SchPort.AreaColor := 0;
    SchPort.TextColor := $FFFFFF;
    SchPort.Name      := 'A new port with no net.';

    // Add a port object onto the existing schematic document
    FSchDoc.RegisterSchObjectInContainer(SchPort);

    // Refresh the schematic sheet.
    FSchDoc.GraphicallyInvalidate;
End;
```

See also

ISch_ServerInterface interface

TObjectCreationMode type

DestroySchObject method

(ISch_ServerInterface interface)

Syntax

```
Procedure DestroySchObject(Var ASchObject : ISch_BasicContainer);
```

Description

Example

See also

ISch_ServerInterface interface

SetState_ProbesTimerEnabled method

(ISch_ServerInterface interface)

Syntax

```
Procedure SetState_ProbesTimerEnabled(AValue : Boolean);
```

Description

The ProbesTimerEnabled property determines whether the Probes are active or not. This feature is used in the LiveDesign process in Altium Designer.

This property is supported by the GetState_ProbesTimerEnabled and SetState_ProbesTimerEnabled methods.

Example

See also

ISch_ServerInterface interface

UpdateSignalValueDisplay method

(ISch_ServerInterface interface)

Syntax

```
Function UpdateSignalValueDisplay(DMObject : IDMObject; Value : Integer; BitIndex : Integer) :
LongBool;
```

Description

Example

See also

ISch_ServerInterface interface

ISch_ServerInterface Properties

FontManager property

(ISch_ServerInterface interface)

Syntax

```
Property FontManager : ISch_FontManager Read GetState_FontManager;
```

Description

This property retrieves the Font manager object which is represented by the ISch_FontManager interface. The property is supported by the GetState_FontManager method.

Example

See also

ISch_Font interface

ISch_FontManager2 interface

ISch_ServerInterface interface

JunctionConvertSettings property

(ISch_ServerInterface interface)

Syntax

```
Property JunctionConvertSettings : ISch_JunctionConvertSettings Read
GetState_JunctionConvertSettings;
```

Description

The JunctionConvertSettings property represents a crossing of wiring on a schematic sheet. When an addition of a wire would create a four-way junction, this is converted to into two adjacent three way junctions. If it is disabled and when a four way junction is created, the two wires crossing at the intersection are not joined electrically and if the Display Cross Overs option is enabled, a cross over is shown on this intersection.

This property is supported by the GetState_JunctionConvertSettings method.

Example

See also

ISch_ServerInterface interface

ISch_JunctionConvertSettings interface

Preferences property

(ISch_ServerInterface interface)

Syntax

```
Property Preferences : ISch_Preferences Read GetState_SchPreferences;
```

Description

This Preferences property retrieves the ISch_Preferences interface which represents the Preferences object for the Schematic Editor. This read only property is supported by the GetState_SchPreference method.

Example

```
Preferences := SchServer.Preferences;
Preferences.WatermarkDeviceSheet.True;
Preferences.WatermarkReadOnlySheet := True;
```

See also

ISch_Preferences interface

ISch_ServerInterface interface

ProbesTimerEnabled property

(ISch_ServerInterface interface)

Syntax

```
Property ProbesTimerEnabled : Boolean Read GetState_ProbesTimerEnabled Write
SetState_ProbesTimerEnabled;
```

Description

The ProbesTimerEnabled property determines whether the Probes are active or not. This feature is used in the LiveDesign process in Altium Designer.

This property is supported by the GetState_ProbesTimerEnabled and SetState_ProbesTimerEnabled methods.

Example

See also

ISch_ServerInterface interface

RobotManager property

(ISch_ServerInterface interface)

Syntax

```
Property RobotManager : ISch_RobotManager Read GetState_RobotManager;
```

Description

This property returns the ISch_RobotManager interface. This interface deals with sending Schematic notification messages in the system. To have the ability to send a specific message when a specific event in the Schematic Editor occurs can be achieved with the ISch_RobotManager interface.

This property is supported by the GetState_RobotManager method.

DelphiScript **Example**

```
SchPort := SchServer.SchObjectFactory(ePort,eCreate_GlobalCopy);
If SchPort = Nil Then Exit;
SchPort.Location  := Point(MilsToCoord(2500),MilsToCoord(2500));
SchPort.Style     := ePortRight;
SchPort.IOType    := ePortBidirectional;
SchPort.Alignment := eHorizontalCentreAlign;
SchPort.Width     := MilsToCoord(500);
SchPort.AreaColor := 0;
SchPort.TextColor := $FF00FF;
SchPort.Name      := 'New Port 4';

// Add a new port object in the existing Schematic document.
Doc.RegisterSchObjectInContainer(SchPort);
SchServer.RobotManager.SendMessage(Doc.I_ObjectAddress,c_BroadCast,
                                   SCHM_PrimitiveRegistration,SchPort.I_ObjectAddress);
```

See also

ISch_ServerInterface interface

ISch_RobotManager interface

ISch_Preferences Interface

Overview

The ISch_Preferences interface represents the global preferences for the Schematic Editor and the settings are the same for any PCB project that has schematics in Altium Designer.

The ISch_ServerInterface interface represents the Schematic Editor and this interface has an ISch_Preferences aggregate object interface.

ISch_Preferences Methods and Properties Table

ISch_Preferences methods	ISch_Preferences properties
Import_FromUser	SelectionColor
Get_SelectionColor	MultiSelectionColor
Get_MultiSelectionColor	ResizeColor
Get_ResizeColor	TranslateRotateColor
Get_TranslateRotateColor	VisibleGridColor
Get_VisibleGridColor	VisibleGridStyle
Get_VisibleGridStyle	GraphicsCursorStyle
Get_GraphicsCursorStyle	OrcadFootPrint
Get_OrcadFootPrint	SnapToCenter

Get_SnapToCenter
Get_UseOrcadPortWidth
Get_AutoBackupTime
Get_AutoBackupFileCount
Get_SelectionReference
Get_UndoRedoStackSize
Get_ConvertSpecialStrings
Get_MaintainOrthogonal
Get_DisplayPrinterFonts
Get_AutoZoom
Get_HotSpotGridDistance
Get_SnapToHotSpot
Get_OptimizePolylines
Get_ComponentsCutWires
Get_AddTemplateToClipBoard
Get_AutoPanStyle
Get_AutoPanJumpDistance
Get_AutoPanShiftJumpDistance
Get_PinNameMargin
Get_PinNumberMargin
Get_DefaultPrimsPermanent
Get_IgnoreSelection
Get_ClickClearsSelection
Get_DoubleClickRunsInspector
Get_MultiPartNamingMethod
Get_Sensitivity
Get_SingleSlashNegation
Get_RunInPlaceEditing
Get_DefaultPowerGndName
Get_DefaultSignalGndName
Get_DefaultEarthName
Get_DefaultTemplateFileName
Get_BufferedPainting
Get_Metafile_NoERCMarkers
Get_Metafile_ParameterSets
Get_Metafile_Probes
Get_DocumentScope
Get_LibraryScope
Get_ConfirmSelectionMemoryClear
Get_LastModelType
Get_StringIncA
Get_StringIncB
Get_MarkManualParameters
Get_CtrlDbleClickGoesDown
Get_SheetStyle_XSize
Get_SheetStyle_YSize

UseOrcadPortWidth
AutoBackupTime
AutoBackupFileCount
SelectionReference
UndoRedoStackSize
ConvertSpecialStrings
MaintainOrthogonal
DisplayPrinterFonts
AutoZoom
HotSpotGridDistance
SnapToHotSpot
OptimizePolylines
ComponentsCutWires
AddTemplateToClipBoard
AutoPanStyle
AutoPanJumpDistance
AutoPanShiftJumpDistance
PinNameMargin
PinNumberMargin
DefaultPrimsPermanent
IgnoreSelection
ClickClearsSelection
DoubleClickRunsInspector
MultiPartNamingMethod
Sensitivity
SingleSlashNegation
RunInPlaceEditing
DefaultPowerGndName
DefaultSignalGndName
DefaultEarthName
DefaultTemplateFileName
BufferedPainting
Metafile_NoERCMarkers
Metafile_ParameterSets
Metafile_Probes
DocumentScope
LibraryScope
ConfirmSelectionMemoryClear
LastModelType
StringIncA
StringIncB
MarkManualParameters
CtrlDbleClickGoesDown
SheetStyle_XSize
SheetStyle_YSize
SheetStyle_XZones

Get_SheetStyle_XZones
Get_SheetStyle_YZones
Get_SheetStyle_MarginWidth
Get_PolylineCutterMode
Get_CutterGridSizeMultiple
Get_CutterFixedLength
Get_ShowCutterBoxMode
Get_ShowCutterMarkersMode
Get_ViolationDisplayByLevel
Get_ViolationColorByLevel
Get_AlwaysDrag
Get_DocMenuID
Get_LibMenuID
Get_DefaultSheetStyle
Get_WireAutoJunctionsColor
Get_ManualJunctionsColor
Get_BusAutoJunctionsColor
Get_DefaultUnit
Get_DefaultUnitSystem
Set_SelectionColor
Set_MultiSelectionColor
Set_ResizeColor
Set_TranslateRotateColor
Set_VisibleGridColor
Set_VisibleGridStyle
Set_GraphicsCursorStyle
Set_OrcadFootPrint
Set_SnapToCenter
Set_UseOrcadPortWidth
Set_AutoBackupTime
Set_AutoBackupFileCount
Set_SelectionReference
Set_UndoRedoStackSize
Set_ConvertSpecialStrings
Set_MaintainOrthogonal
Set_DisplayPrinterFonts
Set_AutoZoom
Set_HotSpotGridDistance
Set_SnapToHotSpot
Set_OptimizePolylines
Set_ComponentsCutWires
Set_AddTemplateToClipBoard
Set_AutoPanStyle
Set_AutoPanJumpDistance
Set_AutoPanShiftJumpDistance
Set_PinNameMargin

SheetStyle_YZones
SheetStyle_MarginWidth
PolylineCutterMode
CutterGridSizeMultiple
CutterFixedLength
ShowCutterBoxMode
ShowCutterMarkersMode
ViolationDisplay
ViolationColor
AlwaysDrag
DocMenuID
LibMenuID
DefaultSheetStyle
WireAutoJunctionsColor
ManualJunctionsColor
BusAutoJunctionsColor
DefaultDisplayUnit
DefaultUnitSystem

Set_PinNumberMargin
Set_DefaultPrimsPermanent
Set_IgnoreSelection
Set_ClickClearsSelection
Set_DoubleClickRunsInspector
Set_MultiPartNamingMethod
Set_Sensitivity
Set_SingleSlashNegation
Set_RunInPlaceEditing
Set_DefaultPowerGndName
Set_DefaultSignalGndName
Set_DefaultEarthName
Set_DefaultTemplateFileName
Set_BufferedPainting
Set_Metafile_NoERCMarkers
Set_Metafile_ParameterSets
Set_Metafile_Probes
Set_DocumentScope
Set_LibraryScope
Set_ConfirmSelectionMemoryClear
Set_LastModelType
Set_StringIncA
Set_StringIncB
Set_MarkManualParameters
Set_CtrlDbleClickGoesDown
Set_PolylineCutterMode
Set_CutterGridSizeMultiple
Set_CutterFixedLength
Set_ShowCutterBoxMode
Set_ShowCutterMarkersMode
Set_ViolationDisplayByLevel
Set_ViolationColorByLevel
Set_AlwaysDrag
Set_DocMenuID
Set_LibMenuID
Set_DefaultSheetStyle
Set_WireAutoJunctionsColor
Set_ManualJunctionsColor
Set_BusAutoJunctionsColor
Set_DefaultUnit

GridPresetsCount
GridPresetAt

See also

ISch_ServerInterface interface

ISch_Document interface

ISch_Preferences Methods

Get_AddTemplateToClipBoard method

(ISch_Preferences interface)

Syntax

```
Function Get_AddTemplateToClipBoard : Boolean;
```

Description

The Get_AddTemplateToClipBoard function when true, adds the current sheet template to the clipboard when you copy or cut from the current schematic sheet.

Example

```
AddTemp := Prefs.Get_AddTemplateToClipBoard;
```

See also

ISch_Preferences interface

Get_AlwaysDrag method

(ISch_Preferences interface)

Syntax

```
Function Get_AlwaysDrag : Boolean;
```

Description

The Get_AlwaysDrag function returns true if you can drag a group of objects on a schematic document and the electrical wiring stay connected. Note, to keep the connections clean while dragging, press the spacebar to cycle through the different corner modes in Altium Designer.

The function returns false if if wiring are left alone and become disconnected when previously connected objects are being dragged.

Example

```
AlwaysDrag := Prefs.Get_AlwaysDrag;
```

See also

ISch_Preferences interface

Get_AutoPanJumpDistance method

(ISch_Preferences interface)

Syntax

```
Function Get_AutoPanJumpDistance : TCoord;
```

Description

The Get_AutoPanJumpDistance function gets the size of each auto-panning step. The step size determines how fast the document pans when auto-panning is enabled. The smaller the value, the slower or finer the auto-panning movement.

Example

```
PanJumpDist := CoordToDxps(Prefs.Get_AutoPanJumpDistance);
```

See also

ISch_Preferences interface

Get_AutoPanShiftJumpDistance method

(ISch_Preferences interface)

Syntax

```
Function Get_AutoPanShiftJumpDistance : TCoord;
```

Description

The Get_AutoPanShiftJumpDistance function returns a value of TCoord type which determines the size of each step when the SHIFT key is held during auto-panning in Altium Designer. The shift step size determines how fast the document pans when auto-panning is enabled and the SHIFT key is pressed. The smaller the value, the slower or finer the auto-panning movement.

Example

```
JumpDist := Prefs.GetAutoPanShiftJumpDistance;
```

See also

ISch_Preferences interface

Get_AutoPanStyle method

(ISch_Preferences interface)

Syntax

```
Function Get_AutoPanStyle : TAutoPanStyle;
```

Description

Example

See also

ISch_Preferences interface

Get_AutoZoom method

(ISch_Preferences interface)

Syntax

```
Function Get_AutoZoom : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_BufferedPainting method

(ISch_Preferences interface)

Syntax

```
Function Get_BufferedPainting : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_BusAutoJunctionsColor method

(ISch_Preferences interface)

Syntax

```
Function Get_BusAutoJunctionsColor : TColor;
```

Description

Example

See also

ISch_Preferences interface

Get_ClickClearsSelection method

(ISch_Preferences interface)

Syntax

```
Function Get_ClickClearsSelection : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_ComponentsCutWires method

(ISch_Preferences interface)

Syntax

```
Function Get_ComponentsCutWires : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_ConfirmSelectionMemoryClear method

(ISch_Preferences interface)

Syntax

```
Function Get_ConfirmSelectionMemoryClear : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_ConvertSpecialStrings method

(ISch_Preferences interface)

Syntax

```
Function Get_ConvertSpecialStrings : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_CtrlDbleClickGoesDown method

(ISch_Preferences interface)

Syntax

```
Function Get_CtrlDbleClickGoesDown : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_CutterFixedLength method

(ISch_Preferences interface)

Syntax

```
Function Get_CutterFixedLength : TCoord;
```

Description

Example

See also

ISch_Preferences interface

Get_CutterGridSizeMultiple method

(ISch_Preferences interface)

Syntax

```
Function Get_CutterGridSizeMultiple : Integer;
```

Description

Example

See also

ISch_Preferences interface

Get_DefaultEarthName method

(ISch_Preferences interface)

Syntax

```
Function Get_DefaultEarthName : WideString;
```

Description

The DefaultEarthName property denotes the default signal ground name to be used for objects on the schematic document. The default name is EARTH.

The Get_DefaultEarthName function retrieves the earth name string.

Example

See also

ISch_Preferences interface

Get_DefaultPowerGndName method

(ISch_Preferences interface)

Syntax

```
Function Get_DefaultPowerGndName : WideString;
```

Description

The DefaultPowerGndName property denotes the default power ground name to be used for objects on the schematic document. The default name is GND.

The Get_DefaultPowerGndName function retrieves the power ground name string.

Example

See also

ISch_Preferences interface

Get_DefaultPrimsPermanent method

(ISch_Preferences interface)

Syntax

```
Function Get_DefaultPrimsPermanent : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_DefaultSheetStyle method

(ISch_Preferences interface)

Syntax

```
Function Get_DefaultSheetStyle : TSheetStyle;
```

Description

Example

See also

ISch_Preferences interface

Get_DefaultSignalGndName method

(ISch_Preferences interface)

Syntax

```
Function Get_DefaultSignalGndName : WideString;
```

Description

The DefaultSignalGndName property denotes the default signal ground name to be used for objects on the schematic document. The default name is SGND.

The Get_DefaultSignalGndName function retrieves the signal ground name string.

Example

See also

ISch_Preferences interface

Get_DefaultTemplateFileName method

(ISch_Preferences interface)

Syntax

```
Function Get_DefaultTemplateFileName : WideString;
```

Description

Example

See also

ISch_Preferences interface

Get_DefaultUnit method

(ISch_Preferences interface)

Syntax

```
Function Get_DefaultUnit : TUnit;
```

Description

Example

See also

ISch_Preferences interface

Get_DefaultUnitSystem method

(ISch_Preferences interface)

Syntax

```
Function Get_DefaultUnitSystem : TUnitSystem;
```

Description

Example

See also

ISch_Preferences interface

Get_DisplayPrinterFonts method

(ISch_Preferences interface)

Syntax

```
Function Get_DisplayPrinterFonts : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_DocMenuID method

(ISch_Preferences interface)

Syntax

```
Function Get_DocMenuID : Widestring;
```

Description

The DocMenuID property determines which pop up menu to pop up depending on whether it is a schematic or a library document. The property returns a widestring format which can be either PUSCHMENU or PUSCHLIBMENU strings and they correspond to the entries in the Schematic Editor's resources file (ADVSCH.RCS file).

Example

See also

ISch_Preferences interface

Get_DocumentScope method

(ISch_Preferences interface)

Syntax

```
Function Get_DocumentScope : TChosenDocumentScope;
```

Description

The DocumentScope property determines the scope for filtering and selection to be applied to the current document or to any open document in Altium Designer. The Get_DocumentScope method sets the Chosen Document scope.

Example

See also

ISch_Preferences interface

Get_DoubleClickRunsInspector method

(ISch_Preferences interface)

Syntax

```
Function Get_DoubleClickRunsInspector : Boolean;
```

Description

This method represents the option to bring up the Inspector dialog instead of the design object's properties dialog when you double click on a design object.

Invoke this function to check if design object's properties dialog is invoked (False) or the Inspector dialog (True) when you double click on a design object.

Example

See also

ISch_Preferences interface

Get_GraphicsCursorStyle method

(ISch_Preferences interface)

Syntax

```
Function Get_GraphicsCursorStyle : TCursorShape;
```

Description

Example

See also

ISch_Preferences interface

Get_HotSpotGridDistance method

(ISch_Preferences interface)

Syntax

```
Function Get_HotSpotGridDistance : Integer;
```

Description

Example

See also

ISch_Preferences interface

Get_IgnoreSelection method

(ISch_Preferences interface)

Syntax

```
Function Get_IgnoreSelection : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_LastModelType method

(ISch_Preferences interface)

Syntax

```
Function Get_LastModelType : WideString;
```

Description

Example

See also

ISch_Preferences interface

Get_LibMenuID method

(ISch_Preferences interface)

Syntax

```
Function Get_LibMenuID : Widestring;
```

Description

Example

See also

ISch_Preferences interface

Get_LibraryScope method

(ISch_Preferences interface)

Syntax

```
Function Get_LibraryScope : TLibraryScope;
```

Description

Example

See also

ISch_Preferences interface

Get_MaintainOrthogonal method

(ISch_Preferences interface)

Syntax

```
Function Get_MaintainOrthogonal : Boolean;
```

Description

The MaintainOrthogonal property if set to true then when you drag components, any wiring that is dragged with the component is kept orthogonal (i.e. corners at 90 degrees). If this option is disabled, wiring dragged with a component will be repositioned obliquely.

This method gets the property true or false and is used in the MaintainOrthogonal property.

Example

See also

ISch_Preferences interface

Get_ManualJunctionsColor method

(ISch_Preferences interface)

Syntax

```
Function Get_ManualJunctionsColor : TColor;
```

Description

Example

See also

ISch_Preferences interface

Get_MarkManualParameters method

(ISch_Preferences interface)

Syntax

```
Function Get_MarkManualParameters : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_Metafile_NoERCMarkers method

(ISch_Preferences interface)

Syntax

```
Function Get_Metafile_NoERCMarkers : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_Metafile_ParameterSets method

(ISch_Preferences interface)

Syntax

```
Function Get_Metafile_ParameterSets : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_MetaFile_Probes method

(ISch_Preferences interface)

Syntax

```
Function Get_Metafile_Probes : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_MultiPartNamingMethod method

(ISch_Preferences interface)

Syntax

```
Function Get_MultiPartNamingMethod : Integer;
```

Description

Example

See also

ISch_Preferences interface

Get_MultiSelectionColor method

(ISch_Preferences interface)

Syntax

```
Function Get_MultiSelectionColor : TColor;
```

Description

Example

See also

ISch_Preferences interface

Get_OptimizePolylines method

(ISch_Preferences interface)

Syntax

```
Function Get_OptimizePolylines : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_OrcadFootPrint method

(ISch_Preferences interface)

Syntax

```
Function Get_OrcadFootPrint : TOrcadFootPrint;
```

Description

Example

See also

ISch_Preferences interface

Get_PinNameMargin method

(ISch_Preferences interface)

Syntax

```
Function Get_PinNameMargin : Integer;
```

Description

Example

See also

ISch_Preferences interface

Get_PinNumberMargin method

(ISch_Preferences interface)

Syntax

```
Function Get_PinNumberMargin : Integer;
```

Description

Example

See also

ISch_Preferences interface

Get_PolylineCutterMode method

(ISch_Preferences interface)

Syntax

```
Function Get_PolylineCutterMode : TPolylineCutterMode;
```

Description

Example

See also

ISch_Preferences interface

Get_ResizeColor method

(ISch_Preferences interface)

Syntax

```
Function Get_ResizeColor : TColor;
```

Description

Example

See also

ISch_Preferences interface

Get_RunInPlaceEditing method

(ISch_Preferences interface)

Syntax

```
Function Get_RunInPlaceEditing : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_SelectionColor method

(ISch_Preferences interface)

Syntax

```
Function Get_SelectionColor : TColor;
```

Description

Example

See also

ISch_Preferences interface

Get_SelectionReference method

(ISch_Preferences interface)

Syntax

```
Function Get_SelectionReference : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_Sensitivity method

(ISch_Preferences interface)

Syntax

```
Function Get_Sensitivity : Integer;
```

Description

Example

See also

ISch_Preferences interface

Get_SheetStyle_MarginWidth method

(ISch_Preferences interface)

Syntax

```
Function Get_SheetStyle_MarginWidth (S : TSheetStyle) : TCoord;
```

Description

Example

See also

ISch_Preferences interface

Get_SheetStyle_XSize method

(ISch_Preferences interface)

Syntax

```
Function Get_SheetStyle_XSize (S : TSheetStyle) : TCoord;
```

Description

Example

See also

ISch_Preferences interface

Get_SheetStyle_XZones method

(ISch_Preferences interface)

Syntax

```
Function Get_SheetStyle_XZones (S : TSheetStyle) : TCoord;
```

Description

Example

See also

ISch_Preferences interface

Get_SheetStyle_YSize method

(ISch_Preferences interface)

Syntax

```
Function Get_SheetStyle_YSize (S : TSheetStyle) : TCoord;
```

Description

Example

See also

ISch_Preferences interface

Get_SheetStyle_YZones method

(ISch_Preferences interface)

Syntax

```
Function Get_SheetStyle_YZones (S : TSheetStyle) : TCoord;
```

Description

Example

See also

ISch_Preferences interface

Get_ShowCutterBoxMode method

(ISch_Preferences interface)

Syntax

```
Function Get_ShowCutterBoxMode : TShowCutterBoxMode;
```

Description

Example

See also

ISch_Preferences interface

Get_ShowCutterMarkersMode method

(ISch_Preferences interface)

Syntax

```
Function Get_ShowCutterMarkersMode : TShowCutterMarkersMode;
```

Description

Example

See also

ISch_Preferences interface

Get_SingleSlashNegation method

(ISch_Preferences interface)

Syntax

```
Function Get_SingleSlashNegation : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_SnapToCenter method

(ISch_Preferences interface)

Syntax

```
Function Get_SnapToCenter : Boolean;
```

Description

This property represents the action where you hold the object being moved or dragged by its reference point (for objects that have one, such as library components or ports), or its center (for objects which do not have a reference point such as a rectangle).

This function returns a boolean value whether the you can snap to the center of a object or not before being moved or dragged by its reference point.

Example

See also

ISch_Preferences interface

Get_SnapToHotSpot method

(ISch_Preferences interface)

Syntax

```
Function Get_SnapToHotSpot : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_StringIncA method

(ISch_Preferences interface)

Syntax

```
Function Get_StringIncA : WideString;
```

Description

Example

See also

ISch_Preferences interface

Get_StringIncB method

(ISch_Preferences interface)

Syntax

```
Function Get_StringIncB : WideString;
```

Description

Example

See also

ISch_Preferences interface

Get_TranslateRotateColor method

(ISch_Preferences interface)

Syntax

```
Function Get_TranslateRotateColor : TColor;
```

Description

Example

See also

ISch_Preferences interface

Get_UndoRedoStackSize method

(ISch_Preferences interface)

Syntax

```
Function Get_UndoRedoStackSize : Integer;
```

Description

Example

See also

ISch_Preferences interface

Get_UseOrcadPortWidth method

(ISch_Preferences interface)

Syntax

```
Function Get_UseOrcadPortWidth : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_ViolationColorByLevel method

(ISch_Preferences interface)

Syntax

```
Function Get_ViolationColorByLevel (ALevel : TErrorLevel) : TColor;
```

Description

Example

See also

ISch_Preferences interface

Get_ViolationDisplayByLevel method

(ISch_Preferences interface)

Syntax

```
Function Get_ViolationDisplayByLevel (ALevel : TErrorLevel) : Boolean;
```

Description

Example

See also

ISch_Preferences interface

Get_VisibleGridColor method

(ISch_Preferences interface)

Syntax

```
Function Get_VisibleGridColor : TColor;
```

Description

Example

See also

ISch_Preferences interface

Get_VisibleGridStyle method

(ISch_Preferences interface)

Syntax

```
Function Get_VisibleGridStyle : TVisibleGrid;
```

Description

Example

See also

ISch_Preferences interface

Get_WireAutoJunctionsColor method

(ISch_Preferences interface)

Syntax

```
Function Get_WireAutoJunctionsColor : TColor;
```

Description

Example

See also

ISch_Preferences interface

GridPresetsCount method

(ISch_Preferences interface)

Syntax

```
Function GridPresetsCount(AUnit : TUnitSystem) : Integer;
```

Description

Example

See also

ISch_Preferences interface

GridPresetAt method

(ISch_Preferences interface)

Syntax

```
Function GridPresetAt (AUnit : TUnitSystem; AnIndex : Integer) : IGridSetting;
```

Description

Example

See also

ISch_Preferences interface

Set_AddTemplateToClipBoard method

(ISch_Preferences interface)

Syntax

```
Procedure Set_AddTemplateToClipBoard (AValue : Boolean);
```

Description

The Set_AddTemplateToClipBoard procedure adds the current sheet template to the clipboard when you copy or cut from the current schematic sheet if the True value is passed in as a parameter. Otherwise the template is not copied ot the clipboard when the value is False.

Example

```
Prefs.Set_AddTemplateToClipBoard(True);
```

See also

ISch_Preferences interface

Set_AlwaysDrag method

(ISch_Preferences interface)

Syntax

```
Procedure Set_AlwaysDrag (AValue : Boolean);
```

Description

The Set_AlwaysDrag procedure if set true you can drag a group of objects on a schematic document and the electrical wiring stay connected. Note, to keep the connections clean while dragging, press the spacebar to cycle through the different corner modes in Altium Designer. Set a false value to leave wiring alone and become disconnected when previously connected objects are being dragged.

Example

```
Prefs.Set_AlwaysDrag(True);
```

See also

ISch_Preferences interface

Set_AutoBackupFileCount method

(ISch_Preferences interface)

Syntax

```
Procedure Set_AutoBackupFileCount (AValue : Integer);
```

Description

Example

See also

ISch_Preferences interface

Set_AutoBackupTime method

(ISch_Preferences interface)

Syntax

```
Procedure Set_AutoBackupTime (AValue : Integer);
```

Description

Example

See also

ISch_Preferences interface

Set_AutoPanJumpDistance method

(ISch_Preferences interface)

Syntax

```
Procedure Set_AutoPanJumpDistance (AValue : TCoord);
```

Description

The Set_AutoPanJumpDistance function sets the size of each auto-panning step with a TCoord value. The step size determines how fast the document pans when auto-panning is enabled. The smaller the value, the slower or finer the auto-panning movement.

Example

```
Prefs.Set_AutoPanJumpDistance(CoordToDxps(Value));
```

See also

ISch_Preferences interface

Set_AutoPanShiftJumpDistance method

(ISch_Preferences interface)

Syntax

```
Procedure Set_AutoPanShiftJumpDistance (AValue : TCoord);
```

Description

The Set_AutoPanShiftJumpDistance sets a value of TCoord type which determines the size of each step when the SHIFT key is held during auto-panning in Altium Designer. The shift step size determines how fast the document pans when auto-panning is enabled and the SHIFT key is pressed. The smaller the value, the slower or finer the auto-panning movement.

Example

```
Prefs.Set_AutoPanShiftJumpDistance(DxpsToCoord(100));
```

See also

ISch_Preferences interface

Set_AutoPanStyle method

(ISch_Preferences interface)

Syntax

```
Procedure Set_AutoPanStyle (AValue : TAutoPanStyle);
```

Description

Example

See also

ISch_Preferences interface

Set_AutoZoom method

(ISch_Preferences interface)

Syntax

```
Procedure Set_AutoZoom (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_BufferedPainting method

(ISch_Preferences interface)

Syntax

```
Procedure Set_BufferedPainting (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_BusAutoJunctionsColor method

(ISch_Preferences interface)

Syntax

```
Procedure Set_BusAutoJunctionsColor (AValue : TColor);
```

Description

Example

See also

ISch_Preferences interface

Set_ClickClearsSelection method

(ISch_Preferences interface)

Syntax

```
Procedure Set_ClickClearsSelection (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_ComponentsCutWires method

(ISch_Preferences interface)

Syntax

```
Procedure Set_ComponentsCutWires (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_ConfirmSelectionMemoryClear method

(ISch_Preferences interface)

Syntax

```
Procedure Set_ConfirmSelectionMemoryClear (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_ConvertSpecialStrings method

(ISch_Preferences interface)

Syntax

```
Procedure Set_ConvertSpecialStrings (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_CtrlDbleClickGoesDown method

(ISch_Preferences interface)

Syntax

```
Procedure Set_CtrlDbleClickGoesDown (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_CutterFixedLength method

(ISch_Preferences interface)

Syntax

```
Procedure Set_CutterFixedLength (AValue : TCoord);
```

Description

Example

See also

ISch_Preferences interface

Set_CutterGridSizeMultiple method

(ISch_Preferences interface)

Syntax

```
Procedure Set_CutterGridSizeMultiple (AValue : Integer);
```

Description

Example

See also

ISch_Preferences interface

Set_DefaultEarthName method

(ISch_Preferences interface)

Syntax

```
Procedure Set_DefaultEarthName (AValue : WideString);
```

Description

Example

See also

ISch_Preferences interface

Set_DefaultPowerGndName method

(ISch_Preferences interface)

Syntax

```
Procedure Set_DefaultPowerGndName (AValue : WideString);
```

Description

Example

See also

ISch_Preferences interface

Set_DefaultPrimsPermanent method

(ISch_Preferences interface)

Syntax

```
Procedure Set_DefaultPrimsPermanent (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_DefaultSheetStyle method

(ISch_Preferences interface)

Syntax

```
Procedure Set_DefaultSheetStyle (AValue : TSheetStyle);
```

Description

Example

See also

ISch_Preferences interface

Set_DefaultSignalGndName method

(ISch_Preferences interface)

Syntax

```
Procedure Set_DefaultSignalGndName (AValue : WideString);
```

Description

Example

See also

ISch_Preferences interface

Set_DefaultTemplateFileName method

(ISch_Preferences interface)

Syntax

```
Procedure Set_DefaultTemplateFileName (AValue : WideString);
```

Description

Example

See also

ISch_Preferences interface

Set_DefaultUnit method

(ISch_Preferences interface)

Syntax

```
Procedure Set_DefaultUnit (AValue : TUnit);
```

Description

Example

See also

ISch_Preferences interface

Set_DisplayPrinterFonts method

(ISch_Preferences interface)

Syntax

```
Procedure Set_DisplayPrinterFonts (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_DocMenuID method

(ISch_Preferences interface)

Syntax

```
Procedure Set_DocMenuID (Const AValue : Widestring);
```

Description

The DocMenuID property determines which pop up menu to pop up depending on whether it is a schematic or a library document. The property returns a widestring format which can be either PUSCHMENU or PUSCHLIBMENU strings and they correspond to the entries in the Schematic Editor's resources file (ADVSCH.RCS file).

The procedure sets the new Document Menu ID value.

Example

See also

ISch_Preferences interface

Set_DocumentScope method

(ISch_Preferences interface)

Syntax

```
Procedure Set_DocumentScope (AValue : TChosenDocumentScope);
```

Description

The DocumentScope property determines the scope for filtering and selection to be applied to the current document or to any open document in Altium Designer. The Set_DocumentScope method sets the Chosen Document scope.

Example

See also

ISch_Preferences interface

Set_DoubleClickRunsInspector method

(ISch_Preferences interface)

Syntax

```
Procedure Set_DoubleClickRunsInspector (AValue : Boolean);
```

Description

This method represents the option to bring up the Inspector dialog instead of the design object's properties dialog when you double click on a design object.

Assign false to this AValue parameter to disable this option if you want to see the design object's properties dialog when you double click on a design object.

Example

See also

ISch_Preferences interface

Set_GraphicsCursorStyle method

(ISch_Preferences interface)

Syntax

```
Procedure Set_GraphicsCursorStyle (AValue : TCursorShape);
```

Description

Example

See also

ISch_Preferences interface

Set_HotSpotGridDistance method

(ISch_Preferences interface)

Syntax

```
Procedure Set_HotSpotGridDistance (AValue : Integer);
```

Description

Example

See also

ISch_Preferences interface

Set_IgnoreSelection method

(ISch_Preferences interface)

Syntax

```
Procedure Set_IgnoreSelection (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_LastModelType method

(ISch_Preferences interface)

Syntax

```
Procedure Set_LastModelType (AValue : WideString);
```

Description

Example

See also

ISch_Preferences interface

Set_LibMenuID method

(ISch_Preferences interface)

Syntax

```
Procedure Set_LibMenuID (Const AValue : Widestring);
```

Description

Example

See also

ISch_Preferences interface

Set_LibraryScope method

(ISch_Preferences interface)

Syntax

```
Procedure Set_LibraryScope (AValue : TLibraryScope);
```

Description

Example

See also

ISch_Preferences interface

Set_MaintainOrthogonal method

(ISch_Preferences interface)

Syntax

```
Procedure Set_MaintainOrthogonal (AValue : Boolean);
```

Description

The MaintainOrthogonal property if set to true then when you drag components, any wiring that is dragged with the component is kept orthogonal (i.e. corners at 90 degrees). If this option is disabled, wiring dragged with a component will be repositioned obliquely.

This method sets the property true or false and is used in the MaintainOrthogonal property.

Example

See also

ISch_Preferences interface

Set_ManualJunctionsColor method

(ISch_Preferences interface)

Syntax

```
Procedure Set_ManualJunctionsColor (AValue : TColor);
```

Description

Example

See also

ISch_Preferences interface

Set_MarkManualParameters method

(ISch_Preferences interface)

Syntax

```
Procedure Set_MarkManualParameters (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_Metafile_NoERCMarkers method

(ISch_Preferences interface)

Syntax

```
Procedure Set_Metafile_NoERCMarkers (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_Metafile_ParameterSets method

(ISch_Preferences interface)

Syntax

```
Procedure Set_Metafile_ParameterSets (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_MetaFile_Probes method

(ISch_Preferences interface)

Syntax

```
Procedure Set_Metafile_Probes(AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_MultiPartNamingMethod method

(ISch_Preferences interface)

Syntax

```
Procedure Set_MultiPartNamingMethod (AValue : Integer);
```

Description

Example

See also

ISch_Preferences interface

Set_MultiSelectionColor method

(ISch_Preferences interface)

Syntax

```
Procedure Set_MultiSelectionColor (AValue : TColor);
```

Description

Example

See also

ISch_Preferences interface

Set_OptimizePolylines method

(ISch_Preferences interface)

Syntax

```
Procedure Set_OptimizePolylines (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_OrcadFootPrint method

(ISch_Preferences interface)

Syntax

```
Procedure Set_OrcadFootPrint (AValue : TOrcadFootPrint);
```

Description

Example

See also

ISch_Preferences interface

Set_PinNameMargin method

(ISch_Preferences interface)

Syntax

```
Procedure Set_PinNameMargin (AValue : Integer);
```

Description

Example

See also

ISch_Preferences interface

Set_PinNumberMargin method

(ISch_Preferences interface)

Syntax

```
Procedure Set_PinNumberMargin (AValue : Integer);
```

Description

Example

See also

ISch_Preferences interface

Set_PolylineCutterMode method

(ISch_Preferences interface)

Syntax

```
Procedure Set_PolylineCutterMode (AValue : TPolylineCutterMode);
```

Description

Example

See also

ISch_Preferences interface

Set_ResizeColor method

(ISch_Preferences interface)

Syntax

```
Procedure Set_ResizeColor (AValue : TColor);
```

Description

Example

See also

ISch_Preferences interface

Set_RunInPlaceEditing method

(ISch_Preferences interface)

Syntax

```
Procedure Set_RunInPlaceEditing (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_SelectionColor method

(ISch_Preferences interface)

Syntax

```
Procedure Set_SelectionColor (AValue : TColor);
```

Description

Example

See also

ISch_Preferences interface

Set_SelectionReference method

(ISch_Preferences interface)

Syntax

```
Procedure Set_SelectionReference (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_Sensitivity method

(ISch_Preferences interface)

Syntax

```
Procedure Set_Sensitivity (AValue : Integer);
```

Description

Example

See also

ISch_Preferences interface

Set_ShowCutterBoxMode method

(ISch_Preferences interface)

Syntax

```
Procedure Set_ShowCutterBoxMode (AValue : TShowCutterBoxMode);
```

Description

Example

See also

ISch_Preferences interface

Set_ShowCutterMarkersMode method

(ISch_Preferences interface)

Syntax

```
Procedure Set_ShowCutterMarkersMode (AValue : TShowCutterMarkersMode);
```

Description

Example

See also

ISch_Preferences interface

Set_SingleSlashNegation method

(ISch_Preferences interface)

Syntax

```
Procedure Set_SingleSlashNegation (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_SnapToCenter method

(ISch_Preferences interface)

Syntax

```
Procedure Set_SnapToCenter (AValue : Boolean);
```

Description

This SnapToCenter property represents the action where you hold the object being moved or dragged by its reference point (for objects that have one, such as library components or ports), or its center (for objects which do not have a reference point such as a rectangle).

The procedure sets whether you can snap to center of the objects or not.

Example

See also

ISch_Preferences interface

Set_SnapToHotSpot method

(ISch_Preferences interface)

Syntax

```
Procedure Set_SnapToHotSpot (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_StringIncA method

(ISch_Preferences interface)

Syntax

```
Procedure Set_StringIncA (AValue : WideString);
```

Description

The Set_StringIncA method represents a value to auto-increment on pin designators of a component when you are placing pins for a component. This is used for building components in the Library editor. Normally you would use a positive increment value for pin designators and negative increment value for pin names. Eg 1, 2,3 for pin designators and D8, D7, D6 for pin names. Thus Primary = 1 and Secondary = -1 and set Display Name to D8 and Designator to 1 in the Pin Properties dialog before you place the first pin.

This method sets the increment value for the pin designators and the StringIncB method sets the increment value for the pin names.

This method is used by the StringIncA property.

Example

See also

ISch_Preferences interface

Set_StringIncB method

(ISch_Preferences interface)

Syntax

```
Procedure Set_StringIncB (AValue : WideString);
```

Description

The Set_StringIncB method represents a value to auto-increment on pin designators of a component when you are placing pins for a component. This is used for building components in the Library editor. Normally you would use a positive increment value for pin designators and negative increment value for pin names. Eg 1, 2,3 for pin designators and D8, D7, D6 for pin names. Thus Primary = 1 and Secondary = -1 and set Display Name to D8 and Designator to 1 in the Pin Properties dialog before you place the first pin.

This method sets the increment value for the pin names and the StringIncA method sets the increment value for the pin designators.

This method is used by the StringIncB property.

Example

See also

ISch_Preferences interface

Set_TranslateRotateColor method

(ISch_Preferences interface)

Syntax

```
Procedure Set_TranslateRotateColor (AValue : TColor);
```

Description

Example

See also

ISch_Preferences interface

Set_UndoRedoStackSize method

(ISch_Preferences interface)

Syntax

```
Procedure Set_UndoRedoStackSize (AValue : Integer);
```

Description

Example

See also

ISch_Preferences interface

Set_UseOrcadPortWidth method

(ISch_Preferences interface)

Syntax

```
Procedure Set_UseOrcadPortWidth (AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_ViolationColorByLevel method

(ISch_Preferences interface)

Syntax

```
Procedure Set_ViolationColorByLevel (ALevel : TErrorLevel;AValue : TColor);
```

Description

Example

See also

ISch_Preferences interface

Set_ViolationDisplayByLevel method

(ISch_Preferences interface)

Syntax

```
Procedure Set_ViolationDisplayByLevel (ALevel : TErrorLevel;AValue : Boolean);
```

Description

Example

See also

ISch_Preferences interface

Set_VisibleGridColor method

(ISch_Preferences interface)

Syntax

```
Procedure Set_VisibleGridColor (AValue : TColor);
```

Description

Example

See also

ISch_Preferences interface

Set_VisibleGridStyle method

(ISch_Preferences interface)

Syntax

```
Procedure Set_VisibleGridStyle (AValue : TVisibleGrid);
```

Description

Example

See also

ISch_Preferences interface

Set_WireAutoJunctionsColor method

(ISch_Preferences interface)

Syntax

```
Procedure Set_WireAutoJunctionsColor (AValue : TColor);
```

Description

Example

See also

ISch_Preferences interface

ISch_Preferences Properties

WireAutoJunctionsColor property

(ISch_Preferences interface)

Syntax

```
Property WireAutoJunctionsColor : TColor Read Get_WireAutoJunctionsColor Write
Set_WireAutoJunctionsColor;
```

Description

This property determines the color of the auto generated junctions on the schematic document. This property is supported by the GetState_WireAutoJunctionsColor and SetState_WireAutoJunctionsColor methods.

Example

See also

ISch_Preferences interface

TColor type

VisibleGridStyle property

(ISch_Preferences interface)

Syntax

```
Property VisibleGridStyle : TVisibleGrid Read Get_VisibleGridStyle Write Set_VisibleGridStyle
;
```

Description

This property determines the lined or dotted style of the visible grid on the schematic document.

Example

See also

ISch_Preferences interface

TVisibleGrid type

VisibleGridColor property

(ISch_Preferences interface)

Syntax

```
Property VisibleGridColor : TColor Read Get_VisibleGridColor Write Set_VisibleGridColor ;
```

Description

This property determines the color of the visible grid on schematic sheets.

Example

See also

ISch_Preferences interface

TColor type

ViolationDisplay property

(ISch_Preferences interface)

Syntax

```
Property ViolationDisplay [L : TErrorLevel] : Boolean Read Get_ViolationDisplayByLevel Write
Set_ViolationDisplayByLevel;
```

Description

This ViolationDisplay property determines the error level for the violation display.

Example

See also

ISch_Preferences interface

TErrorLevel type from Workspace Manager API

ViolationColor property

(ISch_Preferences interface)

Syntax

```
Property ViolationColor [L : TErrorLevel] : TColor Read Get_ViolationColorByLevel Write
Set_ViolationColorByLevel ;
```

Description

This ViolationColor property determines the color of the violation depending on the error level. This property is supported by the Get_ViolationColorByLevel and Set_ViolationColorByLevel methods.

Example

See also

ISch_Preferences interface

TColor type

TErrorLevel type in Workspace Manager API

UseOrcadPortWidth property

(ISch_Preferences interface)

Syntax

```
Property UseOrcadPortWidth : Boolean Read Get_UseOrcadPortWidth Write Set_UseOrcadPortWidth;
```

Description

The UseOrcadPortWidth property determines whether the ports can be re-sized in the Schematic Editor. This is important if the design has to go back to Orcad(TM) (which does not support re-sizing ports).

This property is supported by the Get_UseOrcadPortWidth and Set_UseOrcadPortWidth methods.

Example

See also

ISch_Preferences interface

UndoRedoStackSize property

(ISch_Preferences interface)

Syntax

```
Property UndoRedoStackSize : Integer Read Get_UndoRedoStackSize Write Set_UndoRedoStackSize ;
```

Description

This property shows the number of actions held in the Undo Buffer. The default value is 50. Define a value to set the Undo Buffer size. There is no limit to the size of the Undo Buffer, however, the larger the size, the more main memory is used to store undo information.

Example

See also

ISch_Preferences interface

TranslateRotateColor property

(ISch_Preferences interface)

Syntax

```
Property TranslateRotateColor : TColor Read Get_TranslateRotateColor Write
Set_TranslateRotateColor ;
```

Description

This property sets or gets the color associated with translation or rotation.

Example

See also

ISch_Preferences interface

TColor type

StringIncB property

(ISch_Preferences interface)

Syntax

```
Property StringIncB : WideString Read Get_StringIncB Write Set_StringIncB ;
```

Description

This property represents a value to auto-increment on pin names of a component when you are placing pins for a component. This can be used for building components in the Library editor.

Normally you would use a positive increment value for pin designators and negative increment value for pin names. Eg 1, 2,3 for pin designators and D8, D7, D6 for pin names. Thus Primary = 1 and Secondary = -1 and set Display Name to D8 and Designator to 1 in the Pin Properties dialog before you place the first pin.

This property is supported by the Get_StringIncB and Set_StringIncB methods.

Example

See also

ISch_Preferences interface

StringIncA property

(ISch_Preferences interface)

Syntax

```
Property StringIncA : WideString Read Get_StringIncA Write Set_StringIncA ;
```

Description

This property represents a value to auto-increment on pin designators of a component when you are placing pins for a component. This is used for building components in the Library editor. Normally you would use a positive increment value for pin designators and negative increment value for pin names. Eg 1, 2,3 for pin designators and D8, D7, D6 for pin names. Thus Primary = 1 and Secondary = -1 and set Display Name to D8 and Designator to 1 in the Pin Properties dialog before you place the first pin.

This property is supported by the Get_StringIncA and Set_StringIncA methods.

Example

See also

ISch_Preferences interface

SnapToHotSpot property

(ISch_Preferences interface)

Syntax

```
Property SnapToHotSpot : Boolean Read Get_SnapToHotSpot Write Set_SnapToHotSpot ;
```

Description

This property represents the action where you hold the object being moved or dragged by the nearest electrical hot spot (eg, the end of a pin) when moving or dragging.

Example

See also

ISch_Preferences interface

SnapToCenter property

(ISch_Preferences interface)

Syntax

```
Property SnapToCenter : Boolean Read Get_SnapToCenter Write Set_SnapToCenter ;
```

Description

This property represents the action where you hold the object being moved or dragged by its reference point (for objects that have one, such as library components or ports), or its center (for objects which do not have a reference point such as a rectangle).

Example

See also

ISch_Preferences interface

SingleSlashNegation property

(ISch_Preferences interface)

Syntax

```
Property SingleSlashNegation : Boolean Read Get_SingleSlashNegation Write
Set_SingleSlashNegation ;
```

Description

Example

See also

ISch_Preferences interface

ShowCutterMarkersMode property

(ISch_Preferences interface)

Syntax

```
Property ShowCutterMarkersMode : TShowCutterMarkersMode Read Get_ShowCutterMarkersMode Write
Set_ShowCutterMarkersMode ;
```

Description

Example

See also

ISch_Preferences interface

ShowCutterBoxMode property

(ISch_Preferences interface)

Syntax

```
Property ShowCutterBoxMode : TShowCutterBoxMode Read Get_ShowCutterBoxMode Write
Set_ShowCutterBoxMode ;
```

Description

Example

See also

ISch_Preferences interface

SheetStyle_YZones property

(ISch_Preferences interface)

Syntax

```
Property SheetStyle_YZones [S : TSheetStyle]: TCoord Read Get_SheetStyle_YZones;
```

Description

Example

See also

ISch_Preferences interface

SheetStyle_YSize property

(ISch_Preferences interface)

Syntax

```
Property SheetStyle_YSize [S : TSheetStyle]: TCoord Read Get_SheetStyle_YSize;
```

Description

Example

See also

ISch_Preferences interface

SheetStyle_XZones property

(ISch_Preferences interface)

Syntax

```
Property SheetStyle_XZones [S : TSheetStyle]: TCoord Read Get_SheetStyle_XZones;
```

Description

Example

See also

ISch_Preferences interface

SheetStyle_XSize property

(ISch_Preferences interface)

Syntax

```
Property SheetStyle_XSize [S : TSheetStyle]: TCoord Read Get_SheetStyle_XSize;
```

Description

Example

See also

ISch_Preferences interface

SheetStyle_MarginWidth[S property

(ISch_Preferences interface)

Syntax

```
Property SheetStyle_MarginWidth[S : TSheetStyle]: TCoord Read Get_SheetStyle_MarginWidth;
```

Description

Example

See also

ISch_Preferences interface

Sensitivity property

(ISch_Preferences interface)

Syntax

```
Property Sensitivity : Integer Read Get_Sensitivity Write Set_Sensitivity ;
```

Description

Example

See also

ISch_Preferences interface

SelectionReference property

(ISch_Preferences interface)

Syntax

```
Property SelectionReference : Boolean Read Get_SelectionReference Write Set_SelectionReference
;
```

Description

Example

See also

ISch_Preferences interface

SelectionColor property

(ISch_Preferences interface)

Syntax

```
Property SelectionColor : TColor Read Get_SelectionColor Write Set_SelectionColor ;
```

Description

Example

See also

ISch_Preferences interface

RunInPlaceEditing property

(ISch_Preferences interface)

Syntax

```
Property RunInPlaceEditing : Boolean Read Get_RunInPlaceEditing Write Set_RunInPlaceEditing ;
```

Description

This property if set to true, then the focused text field may be directly edited within the Schematic Editor, rather than in a dialog box. After focusing the field you wish to modify, clicking upon it again or pressing the F2 shortcut key will open the field for editing.

If this property is set to false, you cannot edit the text directly and you have to edit it from the Parameter Properties dialog. You can just graphically move this text field.

Example

See also

ISch_Preferences interface

ResizeColor property

(ISch_Preferences interface)

Syntax

```
Property ResizeColor : TColor Read Get_ResizeColor Write Set_ResizeColor ;
```

Description

Example

See also

ISch_Preferences interface

TColor type

PolylineCutterMode property

(ISch_Preferences interface)

Syntax

```
Property PolylineCutterMode : TPolylineCutterMode Read Get_PolylineCutterMode Write
Set_PolylineCutterMode ;
```

Description

Example

See also

ISch_Preferences interface

PinNumberMargin property

(ISch_Preferences interface)

Syntax

```
Property PinNumberMargin : Integer Read Get_PinNumberMargin Write Set_PinNumberMargin ;
```

Description

Normally, component pin numbers are displayed outside the body of the component, directly above the corresponding pin line. This property controls the placement of the pin numbers. It specifies the distance (in hundredths of an inch) from the component outline to the start of the pin number text. The default is 8.

Example

See also

ISch_Preferences interface

PinNameMargin property

(ISch_Preferences interface)

Syntax

```
Property PinNameMargin : Integer Read Get_PinNameMargin Write Set_PinNameMargin ;
```

Description

Normally, component pin names are displayed inside the body of the component, adjacent to the corresponding pin. This property controls the placement of component pin names. It specifies the distance (in hundredths of an inch) from the component outline to the start of the pin name text. The default is 5.

Example

See also

ISch_Preferences interface

OrcadFootPrint property

(ISch_Preferences interface)

Syntax

```
Property OrcadFootPrint : TOrcadFootPrint Read Get_OrcadFootPrint Write Set_OrcadFootPrint ;
```

Description

Example

See also

ISch_Preferences interface

OptimizePolylines property

(ISch_Preferences interface)

Syntax

```
Property OptimizePolylines : Boolean Read Get_OptimizePolylines Write Set_OptimizePolylines ;
```

Description

If this property is set to true, then extra wires, poly-lines or buses are prevented from overlapping on top of each other and the overlapping wires, poly-lines or busses are removed automatically.

Note: You need to enable this option to have the ability to automatically cut a wire and terminate onto any two pins of this component when this component is dropped onto this wire.

Example

See also

ISch_Preferences interface

MultiSelectionColor property

(ISch_Preferences interface)

Syntax

```
Property MultiSelectionColor : TColor Read Get_MultiSelectionColor Write
Set_MultiSelectionColor ;
```

Description

This property determines the color of the multi_selection, that is multiple objects on the schematic object is being selected.

Example

See also

ISch_Preferences interface

TColor type

MultiPartNamingMethod property

(ISch_Preferences interface)

Syntax

```
Property MultiPartNamingMethod : Integer Read Get_MultiPartNamingMethod Write
Set_MultiPartNamingMethod ;
```

Description

Example

See also

ISch_Preferences interface

Metafile_ParameterSets property

(ISch_Preferences interface)

Syntax

```
Property Metafile_ParameterSets : Boolean Read Get_Metafile_ParameterSets Write
Set_Metafile_ParameterSets ;
```

Description

This property if set to true includes Parameter Sets design objects when copying to the clipboard or when printing a schematic document.

Example

See also

ISch_Preferences interface

Metafile_NoERCMarkers property

(ISch_Preferences interface)

Syntax

```
Property Metafile_NoERCMarkers : Boolean Read Get_Metafile_NoERCMarkers Write
Set_Metafile_NoERCMarkers ;
```

Description

Example

See also

ISch_Preferences interface

MarkManualParameters property

(ISch_Preferences interface)

Syntax

```
Property MarkManualParameters : Boolean Read Get_MarkManualParameters Write
Set_MarkManualParameters;
```

Description

The MarkManualParameters property denotes whether the dots will be displayed or not when parameters of components for example are auto positioned. If true, the dot for the parameter will appear when its associated component has been rotated/moved on the schematic document.

This property is supported by the Get_MarkManualParameters and Set_MarkManualParameters methods.

Example

See also

ISch_Preferences interface

ManualJunctionsColor property

(ISch_Preferences interface)

Syntax

```
Property ManualJunctionsColor : TColor Read Get_ManualJunctionsColor Write
Set_ManualJunctionsColor;
```

Description

Example

See also

ISch_Preferences interface

TColor type

MaintainOrthogonal property

(ISch_Preferences interface)

Syntax

```
Property MaintainOrthogonal : Boolean Read Get_MaintainOrthogonal Write Set_MaintainOrthogonal
;
```

Description

This property if set to true then when you drag components, any wiring that is dragged with the component is kept orthogonal (i.e. corners at 90 degrees). If this option is disabled, wiring dragged with a component will be repositioned obliquely.

This property is supported by the Get_MaintainOrthogonal and Set_MaintainOrthogonal methods.

Example

See also

ISch_Preferences interface

LibraryScope property

(ISch_Preferences interface)

Syntax

```
Property LibraryScope : TLibraryScope Read Get_LibraryScope Write Set_LibraryScope ;
```

Description

This property represents scope for filtering and selection to be applied to the current component on a library sheet or to all components of an open library in Altium Designer.

Example

See also

ISch_Preferences interface

TLibraryScope type

LibMenuID property

(ISch_Preferences interface)

Syntax

```
Property LibMenuID : Widestring Read Get_LibMenuID Write Set_LibMenuID;
```

Description

Example

See also

ISch_Preferences interface

LastModelType property

(ISch_Preferences interface)

Syntax

```
Property LastModelType : WideString Read Get_LastModelType Write Set_LastModelType ;
```

Description

Example

See also

ISch_Preferences interface

Import_FromUser method

(ISch_Preferences interface)

Syntax

```
Function Import_FromUser : Boolean;
```

Description

Example

See also

ISch_Preferences interface

IgnoreSelection property

(ISch_Preferences interface)

Syntax

```
Property IgnoreSelection : Boolean Read Get_IgnoreSelection Write Set_IgnoreSelection ;
```

Description

Example

See also

ISch_Preferences interface

HotSpotGridDistance property

(ISch_Preferences interface)

Syntax

```
Property HotSpotGridDistance : Integer Read Get_HotSpotGridDistance Write
Set_HotSpotGridDistance ;
```

Description

Example

See also

ISch_Preferences interface

GraphicsCursorStyle property

(ISch_Preferences interface)

Syntax

```
Property GraphicsCursorStyle : TCursorShape Read Get_GraphicsCursorStyle Write
Set_GraphicsCursorStyle ;
```

Description

Example

See also

ISch_Preferences interface

AddTemplateToClipBoard property

(ISch_Preferences interface)

Syntax

```
Property AddTemplateToClipBoard : Boolean Read Get_AddTemplateToClipBoard Write
Set_AddTemplateToClipBoard ;
```

Description

The AddTemplateToClipBoard property determines whether the current sheet template can be added to to the clipboard when you copy or cut from the current schematic sheet.

Example

```
Prefs.AddTemplateToClipBoard := True;
```

See also

ISch_Preferences interface

AlwaysDrag property

(ISch_Preferences interface)

Syntax

```
Property AlwaysDrag : Boolean Read Get_AlwaysDrag Write Set_AlwaysDrag;
```

Description

This property represents the AlwaysDrag option and every time you are dragging a group of objects on a schematic document, the electrical wiring stay connected if it is true. Note, to keep the connections clean while dragging, press the spacebar to cycle through the different corner modes.

Set it to false and the wiring are left alone and become disconnected when previously connected objects are being dragged.

Example

```
Prefs.AlwaysDrag := True;
```

See also

ISch_Preferences interface

AutoPanJumpDistance property

(ISch_Preferences interface)

Syntax

```
Property AutoPanJumpDistance : TCoord Read Get_AutoPanJumpDistance Write
Set_AutoPanJumpDistance ;
```

Description

This property represents the value to set/get the size of each auto-panning step. The step size determines how fast the document pans when auto-panning is enabled. The smaller the value, the slower or finer the auto-panning movement.

This property is supported by the GetState_AutoPanJumpDistance and SetState_AutoPanJumpDistance methods.

Example

```
Prefs.AutoPanJumpDistance := CoordToDxps(10);
```

See also

ISch_Preferences interface

AutoPanShiftJumpDistance property

(ISch_Preferences interface)

Syntax

```
Property AutoPanShiftJumpDistance : TCoord Read Get_AutoPanShiftJumpDistance Write
Set_AutoPanShiftJumpDistance ;
```

Description

This property represents a value to get/set the size of each step when the SHIFT key is held during auto-panning. The shift step size determines how fast the document pans when auto-panning is enabled and the SHIFT key is pressed. The smaller the value, the slower or finer the auto-panning movement. This property is supported by the Get_AutoPanShiftJumpDistance and Set_AutoPanShiftJumpDistance methods.

Example

```
Prefs.AutoPanShiftJumpDistance := DxpsToCoord(100);
```

See also

ISch_Preferences interface

AutoPanStyle property

(ISch_Preferences interface)

Syntax

```
Property AutoPanStyle : TAutoPanStyle Read Get_AutoPanStyle Write Set_AutoPanStyle ;
```

Description

Example

See also

ISch_Preferences interface

AutoZoom property

(ISch_Preferences interface)

Syntax

```
Property AutoZoom : Boolean Read Get_AutoZoom Write Set_AutoZoom ;
```

Description

This property if set to true the schematic sheet is automatically zoomed when jumping to a component. Zoom level remains as it was if this option is not enabled.

Example

See also

ISch_Preferences interface

BufferedPainting property

(ISch_Preferences interface)

Syntax

```
Property BufferedPainting : Boolean Read Get_BufferedPainting Write Set_BufferedPainting ;
```

Description

Example

See also

ISch_Preferences interface

BusAutoJunctionsColor property

(ISch_Preferences interface)

Syntax

```
Property BusAutoJunctionsColor : TColor Read Get_BusAutoJunctionsColor Write
Set_BusAutoJunctionsColor;
```

Description

Example

See also

ISch_Preferences interface

TColor type

ClickClearsSelection property

(ISch_Preferences interface)

Syntax

```
Property ClickClearsSelection : Boolean Read Get_ClickClearsSelection Write
Set_ClickClearsSelection ;
```

Description

If this property is set to true, then all design objects are de-selected by clicking any where on the schematic workspace. Set this property to false if you do not want to have this click anywhere to deselect all ability and the selection is cumulative.

Note: regardless of the setting, you can de-select a selected design object by clicking on it.

Example

See also

ISch_Preferences interface

ComponentsCutWires property

(ISch_Preferences interface)

Syntax

```
Property ComponentsCutWires : Boolean Read Get_ComponentsCutWires Write Set_ComponentsCutWires
;
```

Description

Set the property to true so you can drop a component onto a schematic wire and then the wire is cut into two segments and the segments are terminated onto any two hot pins of this component automatically. You will need to set the Optimize Wires & Buses option to true first.

Example

See also

ISch_Preferences interface

ConfirmSelectionMemoryClear property

(ISch_Preferences interface)

Syntax

```
Property ConfirmSelectionMemoryClear : Boolean Read Get_ConfirmSelectionMemoryClear Write
Set_ConfirmSelectionMemoryClear;
```

Description

The selection memories can be used to store the selection state of a set of objects. To prevent inadvertent overwriting of a selection memory, set the property to true.

Example

See also

ISch_Preferences interface

ConvertSpecialStrings property

(ISch_Preferences interface)

Syntax

```
Property ConvertSpecialStrings : Boolean Read Get_ConvertSpecialStrings Write
Set_ConvertSpecialStrings ;
```

Description

This property when set to true, the contents of the special strings on screen are displayed, as they appear on a printout.

Example

See also

ISch_Preferences interface

CtrlDbleClickGoesDown property

(ISch_Preferences interface)

Syntax

```
Property CtrlDbleClickGoesDown : Boolean Read Get_CtrlDbleClickGoesDown Write
Set_CtrlDbleClickGoesDown ;
```

Description

This property when set to true, the sub-sheet of its associated sheet symbol by double clicking on this sheet symbol opens in Altium Designer.

Set it to false and when you double-click on a sheet symbol, the change properties dialog is displayed instead.

Example

See also

ISch_Preferences interface

CutterFixedLength property

(ISch_Preferences interface)

Syntax

```
Property CutterFixedLength : TCoord Read Get_CutterFixedLength Write Set_CutterFixedLength ;
```

Description

Example

See also

ISch_Preferences interface

CutterGridSizeMultiple property

(ISch_Preferences interface)

Syntax

```
Property CutterGridSizeMultiple : Integer Read Get_CutterGridSizeMultiple Write
Set_CutterGridSizeMultiple ;
```

Description

Example

See also

ISch_Preferences interface

DefaultDisplayUnit property

(ISch_Preferences interface)

Syntax

```
Property DefaultDisplayUnit : TUnit Read Get_DefaultUnit Write Set_DefaultUnit;
```

Description

Example

See also

ISch_Preferences interface

DefaultEarthName property

(ISch_Preferences interface)

Syntax

```
Property DefaultEarthName : WideString Read Get_DefaultEarthName Write Set_DefaultEarthName ;
```

Description

The DefaultEarthName denotes the default signal ground name to be used for objects on the schematic document. The default name is EARTH.

This property is supported by the Get_DefaultEarthName and Set_DefaultEarthName methods.

Example

See also

ISch_Preferences interface

DefaultPowerGndName property

(ISch_Preferences interface)

Syntax

```
Property DefaultPowerGndName : WideString Read Get_DefaultPowerGndName Write
Set_DefaultPowerGndName ;
```

Description

Example

See also

ISch_Preferences interface

DefaultPrimsPermanent property

(ISch_Preferences interface)

Syntax

```
Property DefaultPrimsPermanent : Boolean Read Get_DefaultPrimsPermanent Write
Set_DefaultPrimsPermanent ;
```

Description

Example

See also

ISch_Preferences interface

DefaultSheetStyle property

(ISch_Preferences interface)

Syntax

```
Property DefaultSheetStyle : TSheetStyle Read Get_DefaultSheetStyle Write
Set_DefaultSheetStyle;
```

Description

The DefaultSheetStyle property denotes the sheet style used for the workspace.

There are various sheet styles; A4,A3,A2,A1,A0, A,C,D,E,Letter, Legal, Tabloid, Orcad A, Orcad B, Orcad C, Orcad D, Orcad E.

Example

See also

ISch_Preferences interface

TSheetStyle type

DefaultSignalGndName property

(ISch_Preferences interface)

Syntax

```
Property DefaultSignalGndName : WideString Read Get_DefaultSignalGndName Write
Set_DefaultSignalGndName ;
```

Description

The DefaultSignalGndName denotes the default signal ground name to be used for objects on the schematic document. The default name is SGND.

Example

See also

ISch_Preferences interface

DefaultTemplateFileName property

(ISch_Preferences interface)

Syntax

```
Property DefaultTemplateFileName : WideString Read Get_DefaultTemplateFileName Write
Set_DefaultTemplateFileName ;
```

Description

Example

See also

ISch_Preferences interface

DefaultUnitSystem property

(ISch_Preferences interface)

Syntax

```
Property DefaultUnitSystem : TUnitSystem Read Get_DefaultUnitSystem;
```

Description

Example

See also

ISch_Preferences interface

DisplayPrinterFonts property

(ISch_Preferences interface)

Syntax

```
Property DisplayPrinterFonts : Boolean Read Get_DisplayPrinterFonts Write
Set_DisplayPrinterFonts ;
```

Description

The DisplayPrinterFonts property denotes whether the printer fonts can be displayed or not.

Example

See also

ISch_Preferences interface

DocMenuID property

(ISch_Preferences interface)

Syntax

```
Property DocMenuID : Widestring Read Get_DocMenuID Write Set_DocMenuID;
```

Description

The DocMenuID property determines which pop up menu to pop up depending on whether it is a schematic or a library document. The property returns a widestring format which can be either PUSCHMENU or PUSCHLIBMENU strings and they correspond to the entries in the Schematic Editor's resources file (ADVSCH.RCS file).

Example

See also

ISch_Preferences interface

DocumentScope property

(ISch_Preferences interface)

Syntax

```
Property DocumentScope : TChosenDocumentScope Read Get_DocumentScope Write Set_DocumentScope ;
```

Description

The DocumentScope property determines the scope for filtering and selection to be applied to the current document or to any open document in Altium Designer.

Example

See also

ISch_Preferences interface

TChosenDocumentScope type

DoubleClickRunsInspector property

(ISch_Preferences interface)

Syntax

```
Property DoubleClickRunsInspector : Boolean Read Get_DoubleClickRunsInspector Write
Set_DoubleClickRunsInspector ;
```

Description

This property represents the option to bring up the Inspector dialog instead of the design object's properties dialog when you double click on a design object.

Assign false to this property to disable this option if you want to see the design object's properties dialog when you double click on a design object. Invoke this property to check if design object's properties dialog is invoked (False) or the Inspector dialog (True) when you double click on a design object.

Example

See also

ISch_Preferences interface

IGridSetting interface

Overview

The IGridSetting interface represents the grid settings for the Schematic documents part of a project.

The IGridSetting interface hierarchy is a standalone.

IGridSetting methods

GetState_SnapGridOn
GetState_HotspotGridOn
GetState_VisibleGridOn
GetState_SnapGridSize
GetState_HotspotGridSize
GetState_VisibleGridSize
SetState_SnapGridOn
SetState_HotspotGridOn
SetState_VisibleGridOn
SetState_SnapGridSize
SetState_HotspotGridSize
SetState_VisibleGridSize

I_ObjectAddress
CopyTo
SameAs

IGridSetting properties

SnapGridOn
HotspotGridOn
VisibleGridOn
SnapGridSize
HotspotGridSize
VisibleGridSize

See also

ISch_Preferences interface

IGridSetting Methods

CopyTo method

(IGridSetting interface)

Syntax

```
Procedure CopyTo(AGridSetting : IGridSetting);
```

Description

Example

See also

IGridSetting interface

GetState_HotspotGridOn method

(IGridSetting interface)

Syntax

```
Function GetState_HotspotGridOn : Boolean;
```

Description

This function determines whether the hot spot grid is enabled or not and returns a True or False value.

Example

```
If GridSetting.GetState_HotspotGridOn = True Then
    HotspotGridSize := MilsToCoord(4);
```

See also

IGridSetting interface

GetState_HotspotGridSize method

(IGridSetting interface)

Syntax

```
Function GetState_HotspotGridSize : TCoord;
```

Description

This function determines the size of the hot spot grid size.

Example

```
If GridSetting.GetState_HotspotGridOn = True Then
    HotspotGridSize := MilsToCoord(4);
```

See also

IGridSetting interface

GetState_SnapGridOn method

(IGridSetting interface)

Syntax

```
Function GetState_SnapGridOn : Boolean;
```

Description

Example

See also

IGridSetting interface

GetState_SnapGridSize method

(IGridSetting interface)

Syntax

```
Function GetState_SnapGridSize : TCoord;
```

Description

Example

See also

IGridSetting interface

GetState_VisibleGridOn method

(IGridSetting interface)

Syntax

```
Function GetState_VisibleGridOn : Boolean;
```

Description

Example

See also

IGridSetting interface

GetState_VisibleGridSize method

(IGridSetting interface)

Syntax

```
Function GetState_VisibleGridSize : TCoord;
```

Description

Example

See also

IGridSetting interface

I_ObjectAddress method

(IGridSetting interface)

Syntax

```
Function I_ObjectAddress : Pointer;
```

Description

This function returns the object address of the IGridSetting interface as a pointer type.

Example

```
If GridSetting.I_ObjectAddress <> Nil Then ShowMessage(IntToStr(GridSetting.I_ObjectAddress));
```

See also

IGridSetting interface

SameAs method

(IGridSetting interface)

Syntax

```
Function SameAs(AGridSetting : IGridSetting) : Boolean;
```

Description

Example

See also

IGridSetting interface

SetState_HotspotGridOn method

(IGridSetting interface)

Syntax

```
Procedure SetState_HotspotGridOn (B : Boolean);
```

Description

Example

See also

IGridSetting interface

SetState_HotspotGridSize method

(IGridSetting interface)

Syntax

```
Procedure SetState_HotspotGridSize (C : TCoord);
```

Description

Example

See also

IGridSetting interface

SetState_SnapGridOn method

(IGridSetting interface)

Syntax

```
Procedure SetState_SnapGridOn (B : Boolean);
```

Description

Example

See also

IGridSetting interface

SetState_SnapGridSize method

(IGridSetting interface)

Syntax

```
Procedure SetState_SnapGridSize (C : TCoord);
```

Description

Example

See also

IGridSetting interface

SetState_VisibleGridOn method

(IGridSetting interface)

Syntax

```
Procedure SetState_VisibleGridOn (B : Boolean);
```

Description

Example

See also

IGridSetting interface

SetState_VisibleGridSize method

(IGridSetting interface)

Syntax

```
Procedure SetState_VisibleGridSize (C : TCoord);
```

Description

Example

See also

IGridSetting interface

IGridSetting Properties

HotspotGridOn property

(IGridSetting interface)

Syntax

```
Property HotspotGridOn : Boolean Read GetState_HotspotGridOn Write SetState_HotspotGridOn ;
```

Description

Example

See also

IGridSetting interface

HotspotGridSize property

(IGridSetting interface)

Syntax

```
Property HotspotGridSize : TCoord Read GetState_HotspotGridSize Write SetState_HotspotGridSize
;
```

Description

Example

See also

IGridSetting interface

SnapGridOn property

(IGridSetting interface)

Syntax

```
Property SnapGridOn : Boolean Read GetState_SnapGridOn Write SetState_SnapGridOn ;
```

Description

Example

See also

IGridSetting interface

SnapGridSize property

(IGridSetting interface)

Syntax

```
Property SnapGridSize : TCoord Read GetState_SnapGridSize Write SetState_SnapGridSize ;
```

Description

Example

See also

IGridSetting interface

VisibleGridOn property

(IGridSetting interface)

Syntax

```
Property VisibleGridOn : Boolean Read GetState_VisibleGridOn Write SetState_VisibleGridOn ;
```

Description

Example

See also

IGridSetting interface

VisibleGridSize property

(IGridSetting interface)

Syntax

```
Property VisibleGridSize : TCoord Read GetState_VisibleGridSize Write SetState_VisibleGridSize
;
```

Description

Example

See also

IGridSetting interface

ISch_FontManager

ISch_FontManager Interface

Overview

The ISch_FontManager interface represents the internal font manager in Schematic Editor that manages fonts for text based objects on schematic documents.

To have access to the ISch_FontManager interface, you need to invoke the SchServer function;

FontManager := SchServer.FontManager;

ISch_FontManager methods	ISch_FontManager properties
GetState_DefaultHorizontalSysFontId	DefaultHorizontalSysFontId
GetState_DefaultVerticalSysFontId	DefaultVerticalSysFontId
GetState_FontCount	FontCount
GetState_Rotation	Rotation
GetState_Size	Size
GetState_Italic	Italic
GetState_Bold	Bold
GetState_UnderLine	UnderLine
GetState_StrikeOut	StrikeOut
GetState_SaveFlag	SaveFlag
GetState_FontName	FontName
GetFontHandle	
GetFontID	
GetFontSpec	
GetFontSize	
IsFontVertical	
Import_FromUser	

Example

```
SchLabel.Orientation := eRotate90;
```

```
SchLabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,False,False,False,'Times
New Roman');
```

See also

ISch_Label interface

ISch_FontManager Methods

GetFontHandle method

(ISch_FontManager interface)

Syntax

```
Function GetFontHandle (AnId: Integer; Const CurrentLogFont : TLogFont; ScreenSize : Integer):
THandle;
```

Description

This function retrieves the handle of the font.

Example

See also

ISch_FontManager interface

GetFontID method

(ISch_FontManager interface)

Syntax

```
Function GetFontID (Size,Rotation : Integer; Underline,Italic,Bold,StrikeOut : Boolean; Const
FontName : WideString) : TFontID;
```

Description

This function retrieves the font ID of TFontID type that can be used to set the font style of a text based object such as a ISch_Label object.

Example

```
ALabel.FontId := SchServer.FontManager.GetFontID(14,90,False,False,False,False,'Arial');
```

See also

ISch_FontManager interface

TFontID type

GetFontSpec method

(ISch_FontManager interface)

Syntax

```
Procedure GetFontSpec (FontID : TFontID; Var Size,Rotation : Integer; Var
Underline,Italic,Bold,StrikeOut : Boolean; Var FontName : WideString);
```

Description

Every font used in the Schematic document has its own FontID. You can invoke the GetFontSpec function to retrieve font specifications for the supplied Font ID.

Example

See also

ISch_FontManager interface

GetFontSize method

(ISch_FontManager interface)

Syntax

```
Function GetFontSize (FontID : TFontID) : Integer;
```

Description

Example

See also

ISch_FontManager interface

GetState_Bold method

(ISch_FontManager interface)

Syntax

```
Function GetState_Bold (AnId : Integer) : Boolean;
```

Description

This Bold property determines the Bold style for the font. This property is supported by the GetState_Bold method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,False,True,False,'Times New Roman');
```

See also

ISch_FontManager interface

GetState_DefaultHorizontalSysFontId method

(ISch_FontManager interface)

Syntax

```
Function GetState_DefaultHorizontalSysFontId : Integer;
```

Description

Example

See also

ISch_FontManager interface

GetState_DefaultVerticalSysFontId method

(ISch_FontManager interface)

Syntax

```
Function GetState_DefaultVerticalSysFontId : Integer;
```

Description

Example

See also

ISch_FontManager interface

GetState_FontCount method

(ISch_FontManager interface)

Syntax

```
Function GetState_FontCount : Integer;
```

Description

The FontCount property returns the number of fonts used in the Altium Designer. This property is supported by the GetState_FontCount method.

Example

See also

ISch_FontManager interface

GetState_FontName method

(ISch_FontManager interface)

Syntax

```
Function GetState_FontName (AnId : Integer) : TFontName;
```

Description

This indexed FontName property returns the name of an indexed font as a string. Every computer could have a different table of fonts used. The FontName property is supported by the GetState_FontName method.

Example

See also

ISch_FontManager interface

GetState_Italic method

(ISch_FontManager interface)

Syntax

```
Function GetState_Italic (AnId : Integer) : Boolean;
```

Description

This Italic property determines the Italic style for the font. This property is supported by the GetState_Italic method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New Roman'
```

See also

ISch_FontManager interface

GetState_Rotation method

(ISch_FontManager interface)

Syntax

```
Function GetState_Rotation (AnId : Integer) : Integer;
```

Description

The Rotation property determines the orientation of the text object. For ISch_Labels, it is necessary to set the Orientation property of these ISch_Labels as well as the Rotation property for the FontID variables. This property is supported by the GetState_Rotation method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,False,False,False,'Times New Roman');
// Note eRotate90 for the Orientation property, and a 90 value as a parameter for the GetFontID method.
```

See also

ISch_FontManager interface

GetState_SaveFlag method

(ISch_FontManager interface)

Syntax

```
Function GetState_SaveFlag (AnId : Integer) : Boolean;
```

Description

Example

See also

ISch_FontManager interface

GetState_Size method

(ISch_FontManager interface)

Syntax

```
Function GetState_Size (AnId : Integer) : Integer;
```

Description

The Size property determines the font size. This property is supported by the GetState_Size method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New Roman');
// Times New Roman Font size to 14 points - 1st parameter
```

See also

ISch_FontManager interface

GetState_StrikeOut method

(ISch_FontManager interface)

Syntax

```
Function GetState_StrikeOut (AnId : Integer) : Boolean;
```

Description

The StrikeOut property determines whether the font is striked out or not. This property is supported by the GetState_StrikeOut method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New Roman');
// Strikeout set to false (sixth parameter)
```

See also

ISch_FontManager interface

GetState_UnderLine method

(ISch_FontManager interface)

Syntax

```
Function GetState_UnderLine (AnId : Integer) : Boolean;
```

Description

This UnderLine property determines whether the font is underlined or not. This property is supported by the GetState_UnderLine method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New Roman');
// Strikeout set to false (third parameter)
```

See also

ISch_FontManager interface

IsFontVertical method

(ISch_FontManager interface)

Syntax

```
Function IsFontVertical(FontID : TFontID) : Boolean;
```

Description

This function determines whether the font is vertically orientated or not.

Example

See also

ISch_FontManager interface

ISch_FontManager Properties

Bold property

(ISch_FontManager interface)

Syntax

```
Property Bold [Id : Integer] : Boolean Read GetState_Bold ;
```

Description

This Bold property determines the Bold style for the font. This property is supported by the GetState_Bold method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,False,True,False,'Times New Roman');
```

See also

ISch_FontManager interface

GetFontID method

DefaultHorizontalSysFontId property

(ISch_FontManager interface)

Syntax

```
Property DefaultHorizontalSysFontId : Integer Read GetState_DefaultHorizontalSysFontId;
```

Description

Example

See also

ISch_FontManager interface

DefaultVerticalSysFontId property

(ISch_FontManager interface)

Syntax

```
Property DefaultVerticalSysFontId : Integer Read GetState_DefaultVerticalSysFontId;
```

Description

Example

See also

ISch_FontManager interface

FontCount property

(ISch_FontManager interface)

Syntax

```
Property FontCount : Integer Read GetState_FontCount;
```

Description

The FontCount property returns the number of fonts used in the computer system that the Altium Designer is currently residing on. This property is supported by the GetState_FontCount method.

Example

See also

ISch_FontManager interface

FontName property

(ISch_FontManager interface)

Syntax

```
Property FontName [Id : Integer] : TFontName Read GetState_FontName ;
```

Description

This indexed FontName property returns the name of an indexed font as a string. Every computer could have a different table of fonts used. The FontName property is supported by the GetState_FontName method.

Example

See also

ISch_FontManager interface

Italic property

(ISch_FontManager interface)

Syntax

```
Property Italic [Id : Integer] : Boolean Read GetState_Italic ;
```

Description

This Italic property determines the Italic style for the font. This property is supported by the GetState_Italic method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New Roman');
```

See also

ISch_FontManager interface

GetFontID method

Rotation property

(ISch_FontManager interface)

Syntax

```
Property Rotation [Id : Integer] : Integer Read GetState_Rotation ;
```

Description

The Rotation property determines the orientation of the text object. For ISch_Labels, it is necessary to set the Orientation property of these ISch_Labels as well as the Rotation property for the FontID variables. This property is supported by the GetState_Rotation method.

Example

```
ALabel.Orientation := eRotate90;
```

```
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,False,False,False,'Times New
Roman');
// Note eRotate90 for the Orientation property, and a 90 value as a parameter for the
GetFontID method.
```

See also

ISch_FontManager interface

SaveFlag property

(ISch_FontManager interface)

Syntax

```
Property SaveFlag [Id : Integer] : Boolean Read GetState_SaveFlag ;
```

Description

Example

See also

ISch_FontManager interface

Size property

(ISch_FontManager interface)

Syntax

```
Property Size [Id : Integer] : Integer Read GetState_Size ;
```

Description

The Size property determines the font size. This property is supported by the GetState_Size method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New
Roman');
// Times New Roman Font size to 14 points - 1st parameter
```

See also

ISch_FontManager interface

GetFontID method

StrikeOut property

(ISch_FontManager interface)

Syntax

```
Property StrikeOut [Id : Integer] : Boolean Read GetState_StrikeOut;
```

Description

The StrikeOut property determines whether the font is striked out or not. This property is supported by the GetState_StrikeOut method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New
Roman');
// Strikeout set to false (sixth parameter)
```

See also

ISch_FontManager interface

GetFontID method

UnderLine property

(ISch_FontManager interface)

Syntax

```
Property UnderLine [Id : Integer] : Boolean Read GetState_UnderLine;
```

Description

This UnderLine property determines whether the font is underlined or not. This property is supported by the GetState_UnderLine method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New Roman');
// Strikeout set to false (third parameter)
```

See also

ISch_FontManager interface

GetFontID method

ISch_FontManager2 Interface

Overview

The ISch_FontManager2 interface represents the internal font manager in Schematic Editor that manages fonts for text based objects on schematic documents. The ISch_FontManager2 is the same as ISch_FontManager, but all the methods have the Safecall calling convention which is important for SDK purposes.

To have access to the ISch_FontManager interface, you need to invoke the SchServer function;

FontManager := SchServer.FontManager;

ISch_FontManager2 methods	**ISch_FontManage2r properties**
GetState_DefaultHorizontalSysFontId	DefaultHorizontalSysFontId
GetState_DefaultVerticalSysFontId	DefaultVerticalSysFontId
GetState_FontCount	FontCount
GetState_Rotation	Rotation
GetState_Size	Size
GetState_Italic	Italic
GetState_Bold	Bold
GetState_UnderLine	UnderLine
GetState_StrikeOut	StrikeOut
GetState_SaveFlag	SaveFlag
GetState_FontName	FontName
GetFontHandle	
GetFontID	
GetFontSpec	
GetFontSize	
IsFontVertical	
Import_FromUser	

Example

```
SchLabel.Orientation := eRotate90;
```

```
SchLabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,False,False,False,'Times
New Roman');
```

See also

ISch_Label interface

ISch_FontManager2 Methods

GetFontHandle method

(ISch_FontManager2 interface)

Syntax

```
Function GetFontHandle (AnId: Integer; Const CurrentLogFont : TLogFont; ScreenSize : Integer):
THandle;
```

Description

This function retrieves the handle of the font.

Example

See also

ISch_FontManager2 interface

GetFontID method

(ISch_FontManager2 interface)

Syntax

```
Function GetFontID (Size,Rotation : Integer; Underline,Italic,Bold,StrikeOut : Boolean; Const
FontName : WideString) : TFontID;
```

Description

This function retrieves the font ID of TFontID type that can be used to set the font style of a text based object such as a ISch_Label object.

Example

```
ALabel.FontId := SchServer.FontManager.GetFontID(14,90,False,False,False,False,'Arial');
```

See also

ISch_FontManager2 interface

TFontID type

GetFontSpec method

(ISch_FontManager2 interface)

Syntax

```
Procedure GetFontSpec (FontID : TFontID; Var Size,Rotation : Integer; Var
Underline,Italic,Bold,StrikeOut : Boolean; Var FontName : WideString);
```

Description

Every font used in the Schematic document has its own FontID. You can invoke the GetFontSpec function to retrieve font specifications for the supplied Font ID.

Example

See also

ISch_FontManager2 interface

GetFontSize method

(ISch_FontManager2 interface)

Syntax

```
Function GetFontSize (FontID : TFontID) : Integer;
```

Description

Example

See also

ISch_FontManager2 interface

GetState_Bold method

(ISch_FontManager2 interface)

Syntax

```
Function GetState_Bold (AnId : Integer) : Boolean;
```

Description

This Bold property determines the Bold style for the font. This property is supported by the GetState_Bold method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,False,True,False,'Times New Roman');
```

See also

ISch_FontManager2 interface

GetState_DefaultHorizontalSysFontId method

(ISch_FontManager2 interface)

Syntax

```
Function GetState_DefaultHorizontalSysFontId : Integer;
```

Description

Example

See also

ISch_FontManager2 interface

GetState_DefaultVerticalSysFontId method

(ISch_FontManager2 interface)

Syntax

```
Function GetState_DefaultVerticalSysFontId : Integer;
```

Description

Example

See also

ISch_FontManager2 interface

GetState_FontCount method

(ISch_FontManager2 interface)

Syntax

```
Function GetState_FontCount : Integer;
```

Description

The FontCount property returns the number of fonts used in the Altium Designer. This property is supported by the GetState_FontCount method.

Example

See also

ISch_FontManager2 interface

GetState_FontName method

(ISch_FontManager interface)

Syntax

```
Function GetState_FontName (AnId : Integer) : TFontName;
```

Description

This indexed FontName property returns the name of an indexed font as a string. Every computer could have a different table of fonts used. The FontName property is supported by the GetState_FontName method.

Example

See also

ISch_FontManager2 interface

GetState_Italic method

(ISch_FontManager2 interface)

Syntax

```
Function GetState_Italic (AnId : Integer) : Boolean;
```

Description

This Italic property determines the Italic style for the font. This property is supported by the GetState_Italic method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New Roman'
```

See also

ISch_FontManager2 interface

GetState_Rotation method

(ISch_FontManager2 interface)

Syntax

```
Function GetState_Rotation (AnId : Integer) : Integer;
```

Description

The Rotation property determines the orientation of the text object. For ISch_Labels, it is necessary to set the Orientation property of these ISch_Labels as well as the Rotation property for the FontID variables. This property is supported by the GetState_Rotation method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,False,False,False,'Times New Roman');
// Note eRotate90 for the Orientation property, and a 90 value as a parameter for the GetFontID method.
```

See also

ISch_FontManager2 interface

GetState_SaveFlag method

(ISch_FontManager2 interface)

Syntax

```
Function GetState_SaveFlag (AnId : Integer) : Boolean;
```

Description

Example

See also

ISch_FontManager2 interface

GetState_Size method

(ISch_FontManager2 interface)

Syntax

```
Function GetState_Size (AnId : Integer) : Integer;
```

Description

The Size property determines the font size. This property is supported by the GetState_Size method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New Roman');
// Times New Roman Font size to 14 points - 1st parameter
```

See also

ISch_FontManager2 interface

GetState_StrikeOut method

(ISch_FontManager2 interface)

Syntax

```
Function GetState_StrikeOut (AnId : Integer) : Boolean;
```

Description

The StrikeOut property determines whether the font is striked out or not. This property is supported by the GetState_StrikeOut method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New Roman');
// Strikeout set to false (sixth parameter)
```

See also

ISch_FontManager2 interface

GetState_UnderLine method

(ISch_FontManager2 interface)

Syntax

```
Function GetState_UnderLine (AnId : Integer) : Boolean;
```

Description

This UnderLine property determines whether the font is underlined or not. This property is supported by the GetState_UnderLine method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New Roman');
// Strikeout set to false (third parameter)
```

See also

ISch_FontManager2 interface

IsFontVertical method

(ISch_FontManager2 interface)

Syntax

```
Function IsFontVertical(FontID : TFontID) : Boolean;
```

Description

This function determines whether the font is vertically orientated or not.

Example

See also

ISch_FontManager2 interface

ISch_FontManager2 Properties

Bold property

(ISch_FontManager2 interface)

Syntax

```
Property Bold [Id : Integer] : Boolean Read GetState_Bold ;
```

Description

This Bold property determines the Bold style for the font. This property is supported by the GetState_Bold method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,False,True,False,'Times New Roman');
```

See also

ISch_FontManager interface

GetFontID method

DefaultHorizontalSysFontId property

(ISch_FontManager2 interface)

Syntax

```
Property DefaultHorizontalSysFontId : Integer Read GetState_DefaultHorizontalSysFontId;
```

Description

Example

See also

ISch_FontManager2 interface

DefaultVerticalSysFontId property

(ISch_FontManager2 interface)

Syntax

```
Property DefaultVerticalSysFontId : Integer Read GetState_DefaultVerticalSysFontId;
```

Description

Example

See also

ISch_FontManager2 interface

FontCount property

(ISch_FontManager2 interface)

Syntax

```
Property FontCount : Integer Read GetState_FontCount;
```

Description

The FontCount property returns the number of fonts used in the computer system that the Altium Designer is currently residing on. This property is supported by the GetState_FontCount method.

Example

See also

ISch_FontManager interface

FontName property

(ISch_FontManager2 interface)

Syntax

```
Property FontName [Id : Integer] : TFontName Read GetState_FontName ;
```

Description

This indexed FontName property returns the name of an indexed font as a string. Every computer could have a different table of fonts used. The FontName property is supported by the GetState_FontName method.

Example

See also

ISch_FontManager2 interface

Italic property

(ISch_FontManager2 interface)

Syntax

```
Property Italic [Id : Integer] : Boolean Read GetState_Italic ;
```

Description

This Italic property determines the Italic style for the font. This property is supported by the GetState_Italic method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New Roman');
```

See also

ISch_FontManager2 interface

GetFontID method

Rotation property

(ISch_FontManager2 interface)

Syntax

```
Property Rotation [Id : Integer] : Integer Read GetState_Rotation ;
```

Description

The Rotation property determines the orientation of the text object. For ISch_Labels, it is necessary to set the Orientation property of these ISch_Labels as well as the Rotation property for the FontID variables. This property is supported by the GetState_Rotation method.

Example

```
ALabel.Orientation := eRotate90;
```

```
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,False,False,False,'Times New
Roman');
// Note eRotate90 for the Orientation property, and a 90 value as a parameter for the
GetFontID method.
```

See also

ISch_FontManager2 interface

SaveFlag property

(ISch_FontManager2 interface)

Syntax

```
Property SaveFlag [Id : Integer] : Boolean Read GetState_SaveFlag ;
```

Description

Example

See also

ISch_FontManager2 interface

Size property

(ISch_FontManager2 interface)

Syntax

```
Property Size [Id : Integer] : Integer Read GetState_Size ;
```

Description

The Size property determines the font size. This property is supported by the GetState_Size method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New
Roman');
// Times New Roman Font size to 14 points - 1st parameter
```

See also

ISch_FontManager interface

GetFontID method

StrikeOut property

(ISch_FontManager2 interface)

Syntax

```
Property StrikeOut [Id : Integer] : Boolean Read GetState_StrikeOut;
```

Description

The StrikeOut property determines whether the font is striked out or not. This property is supported by the GetState_StrikeOut method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New
Roman');
// Strikeout set to false (sixth parameter)
```

See also

ISch_FontManager interface

GetFontID method

UnderLine property

(ISch_FontManager2 interface)

Syntax

```
Property UnderLine [Id : Integer] : Boolean Read GetState_UnderLine;
```

Description

This UnderLine property determines whether the font is underlined or not. This property is supported by the GetState_UnderLine method.

DelphiScript **Example**

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,True,False,False,'Times New Roman');
// Strikeout set to false (third parameter)
```

See also

ISch_FontManager interface

GetFontID method

ISch_JunctionConvertSettings Interface

Overview

The ISch_JunctionConvertSettings interface hierarchy is as follows;

ISch_JunctionConvertSettings Methods and Properties Table

ISch_JunctionConvertSettings methods	ISch_JunctionConvertSettings properties
GetJunctionConversion	JunctionConversion
SetJunctionConversion	MiterSize
GetMiterSize	BatchMode
SetMiterSize	ShowDialog
GetBatchMode	
SetBatchMode	
GetShowDialog	
SetShowDialog	
Export_ToIniFile	
Import_FromIniFile	

ISch_JunctionConvertSettings Methods

SetShowDialog method

(ISch_JunctionConvertSettings interface)

Syntax

```
Procedure SetShowDialog (Value : Boolean);
```

Description

Example

See also

ISch_JunctionConvertSettings interface

SetMiterSize method

(ISch_JunctionConvertSettings interface)

Syntax

```
Procedure SetMiterSize (Value : TDistance);
```

Description

Example

See also

ISch_JunctionConvertSettings interface

SetJunctionConversion method

(ISch_JunctionConvertSettings interface)

Syntax

```
Procedure SetJunctionConversion(Value : TJunctionConversionKind);
```

Description

Example

See also

ISch_JunctionConvertSettings interface

SetBatchMode method

(ISch_JunctionConvertSettings interface)

Syntax

```
Procedure SetBatchMode (Value : Boolean);
```

Description

Example

See also

ISch_JunctionConvertSettings interface

Import_FromIniFile method

(ISch_JunctionConvertSettings interface)

Syntax

```
Procedure Import_FromIniFile(Const OptionsReader : IOptionsReader);
```

Description

Example

See also

ISch_JunctionConvertSettings interface

GetShowDialog method

(ISch_JunctionConvertSettings interface)

Syntax

```
Function GetShowDialog : Boolean;
```

Description

Example

See also

ISch_JunctionConvertSettings interface

GetMiterSize method

(ISch_JunctionConvertSettings interface)

Syntax

```
Function GetMiterSize : TDistance;
```

Description

Example

See also

ISch_JunctionConvertSettings interface

GetJunctionConversion method

(ISch_JunctionConvertSettings interface)

Syntax

```
Function GetJunctionConversion : TJunctionConversionKind;
```

Description

Example

See also

ISch_JunctionConvertSettings interface

GetBatchMode method

(ISch_JunctionConvertSettings interface)

Syntax

```
Function GetBatchMode : Boolean;
```

Description

Example

See also

ISch_JunctionConvertSettings interface

Export_ToIniFile method

(ISch_JunctionConvertSettings interface)

Syntax

```
Procedure Export_ToIniFile (Const OptionsWriter : IOptionsWriter);
```

Description

Example

See also

ISch_JunctionConvertSettings interface

ISch_JunctionConvertSettings Properties

MiterSize property

(ISch_JunctionConvertSettings interface)

Syntax

```
Property MiterSize : TDistance Read GetMiterSize Write SetMiterSize;
```

Description

Example

See also

ISch_JunctionConvertSettings interface

JunctionConversion property

(ISch_JunctionConvertSettings interface)

Syntax

```
Property JunctionConversion : TJunctionConversionKind Read GetJunctionConversion Write
SetJunctionConversion;
```

Description

Example

See also

ISch_JunctionConvertSettings interface

BatchMode property

(ISch_JunctionConvertSettings interface)

Syntax

```
Property BatchMode : Boolean Read GetBatchMode Write SetBatchMode;
```

Description

Example

See also

ISch_JunctionConvertSettings interface

ShowDialog property

(ISch_JunctionConvertSettings interface)

Syntax

```
Property ShowDialog : Boolean Read GetShowDialog Write SetShowDialog;
```

Description

Example

See also

ISch_JunctionConvertSettings interface

ISch_LibraryRuleChecker Interface

Overview

The ISch_LibraryRuleChecker interface represents the internal library rule checker facility that checks the validity of symbols in schematic libraries.

ISch_LIbraryRuleChecker Methods and Properties Table

ISch_LibraryRuleChecker methods	ISch_LibraryRuleChecker properties
GetState_Duplicate_Pins	Duplicate_Pins
GetState_Duplicate_Component	Duplicate_Component
GetState_Missing_Pin_Number	Missing_Pin_Number
GetState_Missing_Default_Designator	Missing_Default_Designator
GetState_Missing_Footprint	Missing_Footprint
GetState_Missing_Description	Missing_Description
GetState_Missing_Pin_Name	Missing_Pin_Name
GetState_Missing_Pins_In_Sequence	Missing_Pins_In_Sequence
GetState_ShowReport	ShowReport
SetState_Duplicate_Pins	
SetState_Duplicate_Component	
SetState_Missing_Pin_Number	
SetState_Missing_Default_Designator	
SetState_Missing_Footprint	
SetState_Missing_Description	
SetState_Missing_Pin_Name	
SetState_Missing_Pins_In_Sequence	
SetState_ShowReport	
SetState_FromParameters	
Import_FromUser	
Run	
I_ObjectAddress	

ISch_LibraryRuleChecker Methods

GetState_Duplicate_Component method

(ISch_LibraryRuleChecker interface)

Syntax

```
Function GetState_Duplicate_Component : Boolean;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

GetState_Duplicate_Pins method

(ISch_LibraryRuleChecker interface)

Syntax

```
Function GetState_Duplicate_Pins : Boolean;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

GetState_Missing_Default_Designator method

(ISch_LibraryRuleChecker interface)

Syntax

```
Function GetState_Missing_Default_Designator : Boolean;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

GetState_Missing_Description method

(ISch_LibraryRuleChecker interface)

Syntax

```
Function GetState_Missing_Description : Boolean;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

GetState_Missing_Footprint method

(ISch_LibraryRuleChecker interface)

Syntax

```
Function GetState_Missing_Footprint : Boolean;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

GetState_Missing_Pin_Name method

(ISch_LibraryRuleChecker interface)

Syntax

```
Function GetState_Missing_Pin_Name : Boolean;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

GetState_Missing_Pin_Number method

(ISch_LibraryRuleChecker interface)

Syntax

```
Function GetState_Missing_Pin_Number : Boolean;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

GetState_Missing_Pins_In_Sequence method

(ISch_LibraryRuleChecker interface)

Syntax

```
Function GetState_Missing_Pins_In_Sequence : Boolean;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

GetState_ShowReport method

(ISch_LibraryRuleChecker interface)

Syntax

```
Function GetState_ShowReport : Boolean;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

SetState_Duplicate_Component method

(ISch_LibraryRuleChecker interface)

Syntax

```
Procedure SetState_Duplicate_Component (AValue : Boolean);
```

Description

Example

See also

ISch_LibraryRuleChecker interface

SetState_Duplicate_Pins method

(ISch_LibraryRuleChecker interface)

Syntax

```
Procedure SetState_Duplicate_Pins (AValue : Boolean);
```

Description

Example

See also

ISch_LibraryRuleChecker interface

SetState_FromParameters method

(ISch_LibraryRuleChecker interface)

Syntax

```
Function SetState_FromParameters(Parameters : PChar) : Boolean;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

SetState_Missing_Default_Designator method

(ISch_LibraryRuleChecker interface)

Syntax

```
Procedure SetState_Missing_Default_Designator(AValue : Boolean);
```

Description

Example

See also

ISch_LibraryRuleChecker interface

SetState_Missing_Description method

(ISch_LibraryRuleChecker interface)

Syntax

```
Procedure SetState_Missing_Description (AValue : Boolean);
```

Description

Example

See also

ISch_LibraryRuleChecker interface

SetState_Missing_Footprint method

(ISch_LibraryRuleChecker interface)

Syntax

```
Procedure SetState_Missing_Footprint (AValue : Boolean);
```

Description

Example

See also

ISch_LibraryRuleChecker interface

SetState_Missing_Pin_Name method

(ISch_LibraryRuleChecker interface)

Syntax

```
Procedure SetState_Missing_Pin_Name (AValue : Boolean);
```

Description

Example

See also

ISch_LibraryRuleChecker interface

SetState_Missing_Pin_Number method

(ISch_LibraryRuleChecker interface)

Syntax

```
Procedure SetState_Missing_Pin_Number (AValue : Boolean);
```

Description

Example

See also

ISch_LibraryRuleChecker interface

SetState_Missing_Pins_In_Sequence method

(ISch_LibraryRuleChecker interface)

Syntax

```
Procedure SetState_Missing_Pins_In_Sequence (AValue : Boolean);
```

Description

Example

See also

ISch_LibraryRuleChecker interface

SetState_ShowReport method

(ISch_LibraryRuleChecker interface)

Syntax

```
Procedure SetState_ShowReport (AValue : Boolean);
```

Description

Example

See also

ISch_LibraryRuleChecker interface

Import_FromUser method

(ISch_LibraryRuleChecker interface)

Syntax

```
Function Import_FromUser : Boolean;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

I_ObjectAddress method

(ISch_LibraryRuleChecker interface)

Syntax

```
Function I_ObjectAddress : TSCHObjectHandle;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

Run method

(ISch_LibraryRuleChecker interface)

Syntax

```
Function Run : Boolean;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

ISch_LibraryRuleChecker Properties

Duplicate_Component property

(ISch_LibraryRuleChecker interface)

Syntax

```
Property Duplicate_Component : Boolean Read GetState_Duplicate_Component Write
SetState_Duplicate_Component ;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

Duplicate_Pins property

(ISch_LibraryRuleChecker interface)

Syntax

```
Property Duplicate_Pins : Boolean Read GetState_Duplicate_Pins Write SetState_Duplicate_Pins ;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

Missing_Default_Designator property

(ISch_LibraryRuleChecker interface)

Syntax

```
Property Missing_Default_Designator : Boolean Read GetState_Missing_Default_Designator Write
SetState_Missing_Default_Designator;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

Missing_Description property

(ISch_LibraryRuleChecker interface)

Syntax

```
Property Missing_Description : Boolean Read GetState_Missing_Description Write
SetState_Missing_Description ;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

Missing_Footprint property

(ISch_LibraryRuleChecker interface)

Syntax

```
Property Missing_Footprint : Boolean Read GetState_Missing_Footprint Write
SetState_Missing_Footprint ;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

Missing_Pins_In_Sequence property

(ISch_LibraryRuleChecker interface)

Syntax

```
Property Missing_Pins_In_Sequence : Boolean Read GetState_Missing_Pins_In_Sequence Write
SetState_Missing_Pins_In_Sequence ;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

Missing_Pin_Name property

(ISch_LibraryRuleChecker interface)

Syntax

```
Property Missing_Pin_Name : Boolean Read GetState_Missing_Pin_Name Write
SetState_Missing_Pin_Name ;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

Missing_Pin_Number property

(ISch_LibraryRuleChecker interface)

Syntax

```
Property Missing_Pin_Number : Boolean Read GetState_Missing_Pin_Number Write
SetState_Missing_Pin_Number ;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

ShowReport property

(ISch_LibraryRuleChecker interface)

Syntax

```
Property ShowReport : Boolean Read GetState_ShowReport Write SetState_ShowReport ;
```

Description

Example

See also

ISch_LibraryRuleChecker interface

ISch_HitTest Interface

Overview

This ISch_HitTest interface returns you the number of objects and object type at a particular point on the schematic document.

Notes

To specify the location where the objects can be checked on the schematic document, pass in the location (of TLocation type) and invoke the CreateHitTest method from the ISchDocument interface. This location parameter can be set either programmatically or by the ChooseLocationInteractively method form the ISch_Document interface.

ISch_HitTest methods	ISch_HitTest properties
GetState_HitTestCount	HitTestCount
GetState_HitObject	HitObject

See also

ISch_Document interface

CreateHitTest method

ChooseLocationInteractively method

ChooseRectangleInteractively method

TLocation type

ISch_HitTest Methods

GetState_HitObject method

(ISch_HitTest interface)

Syntax

```
Function GetState_HitObject (i : Integer) : ISch_GraphicalObject;
```

Description

This function returns you the indexed object at the particular point on the schematic document. This method is used in the HitObject property.

Example

See also

ISch_HitTest interface

GetState_HitTestCount method

(ISch_HitTest interface)

Syntax

```
Function GetState_HitTestCount : Integer;
```

Description

This function returns you the number of objects at the particular point on the schematic document. This method is used in the HitTestCount property.

Example

See also

ISch_HitTest interface

ISch_HitTest Properties

HitObject property

(ISch_HitTest interface)

Syntax

```
Property HitObject[i : Integer] : ISch_GraphicalObject Read GetState_HitObject;
```

Description

This property returns you the indexed object at the particular point on the schematic document. This property is supported by the GetState_HitObject method.

Example

See also

ISch_HitTest interface

HitTestCount property

HitTestCount property

(ISch_HitTest interface)

Syntax

```
Property HitTestCount : Integer Read GetState_HitTestCount;
```

Description

This property returns you the number of objects at the particular point on the schematic document. This property is supported by the GetState_HitTestCount method.

Example

See also

ISch_HitTest interface

ISch_Iterator Interface

Overview

An iterator object interface represents an existing iterator object which iterates through a design database to fetch specified objects within a specified region if necessary.

Important Notes

Delphi Script does not support sets. Therefore, to specify the object set or the layer set, you need to use the MkSet function to create a set of objects, for example Iterator.AddFilter_ObjectSet(MkSet(ePort));

The TIterationDepth type denotes how deep the iterator can look - look for first level objects (for example standalone system parameters of the document only, or all levels for example all parameters on the document including system parameters, objects' parameters such as component's parameters. By default, eIterateAllLevels value is used.

SetState_FilterAll denotes that all objects and the whole schematic document is to be searched within. Otherwise, use the following AddFilter_ObjectSet, AddFilter_Area etc methods to set up a restricted search.

The ISch_Iterator interface hierarchy is as follows;

ISch_Iterator Methods and Properties Table

ISch_Iterator methods	ISch_Iterator properties
I_ObjectAddress	
SetState_FilterAll	
AddFilter_ObjectSet	
AddFilter_CurrentPartPrimitives	
AddFilter_CurrentDisplayModePrimitives	
AddFilter_PartPrimitives	
AddFilter_Area	
SetState_IterationDepth	
FirstSchObject	
NextSchObject	

See also

ISch_BasicContainer interface

ISch_Lib interface

ISch_Iterator Methods

AddFilter_Area method

(ISch_Iterator interface)

Syntax

```
Procedure AddFilter_Area(X1, Y1, X2, Y2 : TCoord);
```

Description

The AddFilter_Area procedure defines the rectangular bounds (X1,Y1 and X2,Y2) of the schematic/library document that the iterator will search within.

Example

See also

ISch_Iterator interface

TCoord type

AddFilter_CurrentDisplayModePrimitives method

(ISch_Iterator interface)

Syntax

```
Procedure AddFilter_CurrentDisplayModePrimitives;
```

Description

This procedure sets the iterator to look for current display mode primitives only. A component can be represented by different modes - ie there can be different graphical representations of the same component type.

Example

See also

ISch_Iterator interface

AddFilter_CurrentPartPrimitives method

(ISch_Iterator interface)

Syntax

```
Procedure AddFilter_CurrentPartPrimitives;
```

Description

This procedure sets up the filter of the iterator to look for the current primitives of a part only. A component can be composed of multiple parts and each part is identified by its PartID value.

Example

See also

ISch_Iterator interface

AddFilter_ObjectSet method

(ISch_Iterator interface)

Syntax

```
Procedure AddFilter_ObjectSet(Const AObjectSet : TObjectSet);
```

Description

This procedure defines which objects the iterator will look for on a schematid document or a library document.

Example

See also

ISch_Iterator interface

TObjectSet type

AddFilter_PartPrimitives method

(ISch_Iterator interface)

Syntax

```
Procedure AddFilter_PartPrimitives(APartId : Integer; ADisplayMode : TDisplayMode);
```

Description

This procedure sets up the filter of the iterator to look for primitives of a part (of a component). A component can be a multi-part component, for example a 74LS04 can have four parts and they are identified by the PartID value.

Example

See also

ISch_Iterator interface

TDisplayMode type in Workspace Manager API

FirstSchObject method

(ISch_Iterator interface)

Syntax

```
Function FirstSchObject : ISch_BasicContainer;
```

Description

The FirstSchObject function fetches the first object found by the iterator. The FirstSchObject method is to be invoked first and then in a While Nil loop, the NextSchObject is called repeatedly until it returns a nil value where the loop is terminated.

DelphiScript **Example**

```
Iterator   := CurrentSheet.SchIterator_Create;
Iterator.AddFilter_ObjectSet(MkSet(ePort));
If Iterator = Nil Then Exit;
Try
    Port := Iterator.FirstSchObject;
    While Port <> Nil Do
    Begin
        PortNumber := PortNumber + 1;
        Port := Iterator.NextSchObject;
    End;
Finally
    CurrentSheet.SchIterator_Detroy(Iterator);
End;
```

See also

ISch_Iterator interface

NextSchObject interface

I_ObjectAddress method

(ISch_Iterator interface)

Syntax

```
Function I_ObjectAddress : TSCHObjectHandle;
```

Description

This function obtains the pointer to the iterator object.

Example

See also

ISch_Iterator interface

TSchObjectHandle type

NextSchObject method

(ISch_Iterator interface)

Syntax

```
Function NextSchObject : ISch_BasicContainer;
```

Description

The NextSchObject function fetches the next object found by the iterator. The FirstSchObject method is to be invoked first and then in a While Nil loop, the NextSchObject is called repeatedly until it returns a nil value where the loop is terminated.

DelphiScript **Example**

```
Iterator   := CurrentSheet.SchIterator_Create;
Iterator.AddFilter_ObjectSet(MkSet(ePort));
If Iterator = Nil Then Exit;
Try
    Port := Iterator.FirstSchObject;
    While Port <> Nil Do
    Begin
        PortNumber := PortNumber + 1;
        Port := Iterator.NextSchObject;
    End;
Finally
    CurrentSheet.SchIterator_Detroy(Iterator);
End;
```

See also

ISch_Iterator interface

FirstSchObject method

SetState_FilterAll method

(ISch_Iterator interface)

Syntax

```
Procedure SetState_FilterAll;
```

Description

This procedure sets the iterator to look for everything on a document.

Example

See also

ISch_Iterator interface

SetState_IterationDepth method

(ISch_Iterator interface)

Syntax

```
Procedure SetState_IterationDepth(AIterationDepth : TIterationDepth);
```

Description

The TIterationDepth type denotes how deep the iterator can look on a document.

Look for first level objects, for example standalone system parameters of the document only, or all levels for example all parameters on the document including system parameters, objects' parameters such as component's parameters.

By default, eIterateAllLevels value is used.

Example

See also

ISch_Iterator interface

TIterationDepth type

ILibCompInfoReader Interface

Overview

The ILibCompInfoReader interface represents the object which has the list of library components (symbols) of a loaded schematic library.

A Schematic library file with a SchLib extension can be loaded in the object represented by the ILibCompInfoReader interface and to obtain each component (Symbol), invoke the indexed ComponentInfos method. This method fetches the object which is represented by the IComponentInfo interface.

The steps required to load a schematic library and its components.

1. Create an object and pass in the filename of a schematic library file. This object is represented by the ILibCompInfoReader interface. This object is created by the SchServer.CreateLibCompInfoReader(LibraryFileName);

2. Invoke the ReadAllComponentInfo method to load the components specified by the library name.

3. Invoke the NumComponentInfos method to obtain the number of components for this library

4. Obtain the indexed ComponentInfos method. This ComponentInfos method returns the indexed IComponentInfo interface.

ILibCompInfoReader methods

GetState_ComponentInfo

GetState_FileName

ReadAllComponentInfo

NumComponentInfos

I_ObjectAddress

ILibCompInfoReader properties

ComponentInfos

FileName

ILibCompInfoReader Methods

GetState_ComponentInfo method

(ILibCompInfoReader interface)

Syntax

```
Function GetState_ComponentInfo (i : Integer) : IComponentInfo;
```

Description

This GetState_ComponentInfo function retrieves the indexed IComponentInfo interface representing the component information datastructure. The ComponentInfo interface contains information such as component name, alias name, part count and offset for the indexed schematic symbol (component) in the library.

Example

```
Var
    ALibCompReader : ICompInfoReader;
    CompInfo       : IComponentInfo;
    CompNum, J     : Integer;
Begin
    ALibCompReader := SchServer.CreateLibCompInfoReader(FileName);
    ALibCompReader.ReadAllComponentInfo;
    CompNum := ALIbCompReader.NumComponentInfos;
    For J := 0 To CompNum -1 Do
    Begin
        ReportInfo.Add(FileName);
```

```
        CompInfo := ALibCompReader.ComponentInfos[J];
        ReportInfo.Add(' Name : '         + CompInfo.CompName);
        ReportInfo.Add('  Alias Name : '  + CompInfo.AliasName);
        ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.PartCount));
        ReportInfo.Add('  Description : ' + CompInfo.Description);
        ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.Offset));
        ReportInfo.Add('  FileName : '    + CompInfo.FileName);
        ReportInfo.Add('');
    End;
```

See also

ILibCompInfoReader interface

IComponentInfo interface

GetState_FileName method

(ILibCompInfoReader interface)

Syntax

```
Function GetState_FileName : WideString;
```

Description

This GetState_FileName function gets the temporary filename of the datastructure.

Example

```
Var
    ALibCompReader : ICompInfoReader;
    CompInfo       : IComponentInfo;
    CompNum, J     : Integer;
Begin
    ALibCompReader := SchServer.CreateLibCompInfoReader(FileName);
    ALibCompReader.ReadAllComponentInfo;
    ShowMessage(ALibCompReader.GetState_FileName);
    CompNum := ALIbCompReader.NumComponentInfos;
    For J := 0 To CompNum -1 Do
    Begin
        ReportInfo.Add(FileName);
        CompInfo := ALibCompReader.ComponentInfos[J];
        ReportInfo.Add(' Name : '         + CompInfo.CompName);
        ReportInfo.Add('  Alias Name : '  + CompInfo.AliasName);
        ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.PartCount));
        ReportInfo.Add('  Description : ' + CompInfo.Description);
        ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.Offset));
        ReportInfo.Add('  FileName : '    + CompInfo.FileName);
        ReportInfo.Add('');
    End;
```

See also

ILibCompInfoReader interface

IComponentInfo interface

I_ObjectAddress method

(ILibCompInfoReader interface)

Syntax

```
Function I_ObjectAddress : TSCHObjectHandle;
```

Description

This function obtains the pointer to the ILibCompInfoReader object.

Example

See also

ILibCompInfoReader interface

NumComponentInfos method

(ILibCompInfoReader interface)

Syntax

```
Function NumComponentInfos : Integer;
```

Description

This NumComponentInfos function retrieves the number of component information data structures. This method is also used by the ComponentInfos property. The ComponentInfo interface contains information such as component name, alias name, part count and offset for the indexed schematic symbol (component) in the library.

Example

```
Var
    ALibCompReader : ICompInfoReader;
    CompInfo       : IComponentInfo;
    CompNum, J     : Integer;
Begin
    ALibCompReader := SchServer.CreateLibCompInfoReader(FileName);
    ALibCompReader.ReadAllComponentInfo;
    ShowMessage(ALibCompReader.GetState_FileName);
    CompNum := ALIbCompReader.NumComponentInfos;
    For J := 0 To CompNum -1 Do
    Begin
        ReportInfo.Add(FileName);
        CompInfo := ALibCompReader.ComponentInfos[J];
        ReportInfo.Add(' Name : '         + CompInfo.CompName);
        ReportInfo.Add('  Alias Name : '  + CompInfo.AliasName);
        ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.PartCount));
        ReportInfo.Add('  Description : ' + CompInfo.Description);
        ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.Offset));
        ReportInfo.Add('  FileName : '    + CompInfo.FileName);
        ReportInfo.Add('');
    End;
```

See also

ILibCompInfoReader interface

ReadAllComponentInfo method

(ILibCompInfoReader interface)

Syntax

```
Procedure ReadAllComponentInfo;
```

Description

The ReadAllComponentInfo retrieves all the IComponentInfo data structures for the ILibCompInfoReader interface. The ComponentInfo interface contains information such as component name, alias name, part count and offset for the indexed schematic symbol (component) in the library.

Example

```
Var
    ALibCompReader : ICompInfoReader;
    CompInfo       : IComponentInfo;
    CompNum, J     : Integer;
Begin
    ALibCompReader := SchServer.CreateLibCompInfoReader(FileName);
    ALibCompReader.ReadAllComponentInfo;
    ShowMessage(ALibCompReader.GetState_FileName);
    CompNum := ALIbCompReader.NumComponentInfos;
    For J := 0 To CompNum -1 Do
    Begin
        ReportInfo.Add(FileName);
        CompInfo := ALibCompReader.ComponentInfos[J];
        ReportInfo.Add(' Name : '         + CompInfo.CompName);
        ReportInfo.Add('  Alias Name : '  + CompInfo.AliasName);
        ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.PartCount));
        ReportInfo.Add('  Description : ' + CompInfo.Description);
        ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.Offset));
        ReportInfo.Add('  FileName : '    + CompInfo.FileName);
        ReportInfo.Add('');
    End;
```

See also

ILibCompInfoReader interface

ILibCompInfoReader Properties

ComponentInfos property

(ILibCompInfoReader interface)

Syntax

```
Property ComponentInfos[i : Integer] : IComponentInfo Read GetState_ComponentInfo;
```

Description

This ComponentInfos property retrieves the indexed IComponentInfo data structure. This property is supported by the GetState_ComponentInfo method. The ComponentInfo interface contains information such as component name, alias name, part count and offset for the indexed schematic symbol (component) in the library.

Example

```
Var
    ALibCompReader : ICompInfoReader;
    CompInfo       : IComponentInfo;
    CompNum, J     : Integer;
Begin
    ALibCompReader := SchServer.CreateLibCompInfoReader(FileName);
    ALibCompReader.ReadAllComponentInfo;
    ShowMessage(ALibCompReader.GetState_FileName);
```

```
CompNum := ALIbCompReader.NumComponentInfos;
For J := 0 To CompNum -1 Do
Begin
    ReportInfo.Add(FileName);
    CompInfo := ALibCompReader.ComponentInfos[J];
    ReportInfo.Add(' Name : '         + CompInfo.CompName);
    ReportInfo.Add('  Alias Name : '  + CompInfo.AliasName);
    ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.PartCount));
    ReportInfo.Add('  Description : ' + CompInfo.Description);
    ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.Offset));
    ReportInfo.Add('  FileName : '    + CompInfo.FileName);
    ReportInfo.Add('');
End;
```

See also

ILibCompInfoReader interface

FileName property

(ILibCompInfoReader interface)

Syntax

```
Property FileName : WideString Read GetState_FileName;
```

Description

This FileName property gets the temporary filename of the datastructure. The FileName property is supported by the GetState_FileName function.

Example

```
ShowMessage(ALibCompReader.Filename)
```

See also

ILibCompInfoReader interface

IComponentInfo Interface

Overview

The IComponentInfo interface is an item within the ILibCompInfoReader interface. This IComponentInfo interface represents a schematic symbol in a specified schematic library file with a SchLib extension.

The steps required to load a schematic library and its components.

1. Create an object and pass in the filename of a schematic library file. This object is represented by the ILibCompInfoReader interface by the SchServer.CreateLibCompInfoReader(FileName);

2. Invoke the ReadAllComponentInfo method to load the library and its components.

3. Invoke the NumComponentInfos method to obtain the number of components for this library

4. Obtain the indexed ComponentInfos method. This ComponentInfos method returns the indexed IComponentInfo interface.

Notes

The IComponentInfo interface is extracted from the ILibCompInfoReader.ComponentInfos[Index] method.

IComponentInfo methods	IComponentInfo properties
GetState_Offset	Offset
GetState_AliasName	AliasName
GetState_CompName	CompName
GetState_PartCount	PartCount

GetState_Description	Description

See also

ILibCompInfoReader interface

IComponentInfo Methods

GetState_AliasName method

(IComponentInfo interface)

Syntax

```
Function GetState_AliasName : WideString;
```

Description

This function returns the alias name for this component. Ie a component can be referred to by one of its multiple names.

Example

```
// Obtain the number of components in the specified sch library.
CompNum := ALibCompReader.NumComponentInfos;

// Go thru each component obtained by the LibCompReader interface.
For J := 0 To CompNum - 1 Do
Begin
    ReportInfo.Add(FileName);
    CompInfo := ALibCompReader.ComponentInfos[J];
    ReportInfo.Add(' Name : '         + CompInfo.CompName);
    ReportInfo.Add('  Alias Name : '  + CompInfo.GetState_AliasName);
    ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.PartCount));
    ReportInfo.Add('  Description : ' + CompInfo.Description);
    ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.Offset));
    ReportInfo.Add('');
End;
```

See also

IComponentInfo interface

GetState_CompName method

(IComponentInfo interface)

Syntax

```
Function GetState_CompName : WideString;
```

Description

This function returns the name string for this component from the IComponentInfo object interface.

Example

```
// Obtain the number of components in the specified sch library.
CompNum := ALibCompReader.NumComponentInfos;

// Go thru each component obtained by the LibCompReader interface.
For J := 0 To CompNum - 1 Do
Begin
    ReportInfo.Add(FileName);
    CompInfo := ALibCompReader.ComponentInfos[J];
    ReportInfo.Add(' Name : '         + CompInfo.GetState_CompName);
```

```
    ReportInfo.Add('  Alias Name : '  + CompInfo.GetState_AliasName);
    ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.GetState_PartCount));
    ReportInfo.Add('  Description : ' + CompInfo.Getstate_Description);
    ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.GetState_Offset));
    ReportInfo.Add('');
End;
```

See also

IComponentInfo interface

GetState_Description method

(IComponentInfo interface)

Syntax

```
Function GetState_Description : WideString;
```

Description

This function returns the description string for this component from the IComponentInfo object interface.

Example

```
// Obtain the number of components in the specified sch library.
CompNum := ALibCompReader.NumComponentInfos;

// Go thru each component obtained by the LibCompReader interface.
For J := 0 To CompNum - 1 Do
Begin
    ReportInfo.Add(FileName);
    CompInfo := ALibCompReader.ComponentInfos[J];
    ReportInfo.Add(' Name : '         + CompInfo.GetState_CompName);
    ReportInfo.Add('  Alias Name : '  + CompInfo.GetState_AliasName);
    ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.GetStatePartCount));
    ReportInfo.Add('  Description : ' + CompInfo.GetState_Description);
    ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.GetState_Offset));
    ReportInfo.Add('');
End;
```

See also

IComponentInfo interface

GetState_Offset method

(IComponentInfo interface)

Syntax

```
Function GetState_Offset : Integer;
```

Description

This function returns the offset as a number - each part of a component whole has an offset to denote its place within the component.

Example

```
// Obtain the number of components in the specified sch library.
CompNum := ALibCompReader.NumComponentInfos;

// Go thru each component obtained by the LibCompReader interface.
For J := 0 To CompNum - 1 Do
Begin
```

```
    ReportInfo.Add(FileName);
    CompInfo := ALibCompReader.ComponentInfos[J];
    ReportInfo.Add(' Name : '         + CompInfo.GetState_CompName);
    ReportInfo.Add('  Alias Name : '  + CompInfo.GetState_AliasName);
    ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.GetState_PartCount));
    ReportInfo.Add('  Description : ' + CompInfo.GetState_Description);
    ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.GetState_Offset));
    ReportInfo.Add('');
End;
```

See also

IComponentInfo interface

GetState_PartCount method

(IComponentInfo interface)

Syntax

```
Function GetState_PartCount : Integer;
```

Description

This function obtains the number of parts (multiple types of the same component type as an example). For example an Integrated circuit may have multiple smaller modules, such as a 74LS00 has multiple OR gates.

Example

```
// Obtain the number of components in the specified sch library.
CompNum := ALibCompReader.NumComponentInfos;

// Go thru each component obtained by the LibCompReader interface.
For J := 0 To CompNum - 1 Do
Begin
    ReportInfo.Add(FileName);
    CompInfo := ALibCompReader.ComponentInfos[J];
    ReportInfo.Add(' Name : '         + CompInfo.GetState_CompName);
    ReportInfo.Add('  Alias Name : '  + CompInfo.GetState_AliasName);
    ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.GetState_PartCount));
    ReportInfo.Add('  Description : ' + CompInfo.GetState_Description);
    ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.GetState_Offset));
    ReportInfo.Add('');
End;
```

See also

IComponentInfo interface

IComponentInfo Properties

AliasName property

(IComponentInfo interface)

Syntax

```
Property AliasName : WideString Read GetState_AliasName;
```

Description

This property returns the alias name for this component. Ie a component can be referred to by one of its multiple names. This property is supported by the GetState_AliasName method.

Example

```
// Obtain the number of components in the specified sch library.
CompNum := ALibCompReader.NumComponentInfos;

// Go thru each component obtained by the LibCompReader interface.
For J := 0 To CompNum - 1 Do
Begin
    ReportInfo.Add(FileName);
    CompInfo := ALibCompReader.GetState_ComponentInfos[J];
    ReportInfo.Add(' Name : '         + CompInfo.CompName);
    ReportInfo.Add('  Alias Name : '  + CompInfo.AliasName);
    ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.PartCount));
    ReportInfo.Add('  Description : ' + CompInfo.Description);
    ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.Offset));
    ReportInfo.Add('');
End;
```

See also

IComponentInfo interface

CompName property

(IComponentInfo interface)

Syntax

```
Property CompName : WideString Read GetState_CompName;
```

Description

This property returns the name string for this component from the IComponentInfo object interface. This property is supported by the GetState_CompName function.

Example

```
// Obtain the number of components in the specified sch library.
CompNum := ALibCompReader.NumComponentInfos;

// Go thru each component obtained by the LibCompReader interface.
For J := 0 To CompNum - 1 Do
Begin
    ReportInfo.Add(FileName);
    CompInfo := ALibCompReader.GetState_ComponentInfos[J];
    ReportInfo.Add(' Name : '         + CompInfo.CompName);
    ReportInfo.Add('  Alias Name : '  + CompInfo.AliasName);
    ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.PartCount));
    ReportInfo.Add('  Description : ' + CompInfo.Description);
    ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.Offset));
    ReportInfo.Add('');
End;
```

See also

IComponentInfo interface

Description property

(IComponentInfo interface)

Syntax

```
Property Description : WideString Read GetState_Description;
```

Description

This property returns the description string for this component from the IComponentInfo object interface. This property is supported by the GetState_Description method.

Example

```
// Obtain the number of components in the specified sch library.
CompNum := ALibCompReader.NumComponentInfos;

// Go thru each component obtained by the LibCompReader interface.
For J := 0 To CompNum - 1 Do
Begin
    ReportInfo.Add(FileName);
    CompInfo := ALibCompReader.GetState_ComponentInfos[J];
    ReportInfo.Add(' Name : '         + CompInfo.CompName);
    ReportInfo.Add('  Alias Name : '  + CompInfo.AliasName);
    ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.PartCount));
    ReportInfo.Add('  Description : ' + CompInfo.Description);
    ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.Offset));
    ReportInfo.Add('');
End;
```

See also

IComponentInfo interface

Offset property

(IComponentInfo interface)

Syntax

```
Property Offset : Integer Read GetState_Offset;
```

Description

This property returns the offset as a number - each part of a component whole has an offset to denote its place within the component. This property is supported by the GetState_Offset function.

Example

```
// Obtain the number of components in the specified sch library.
CompNum := ALibCompReader.NumComponentInfos;

// Go thru each component obtained by the LibCompReader interface.
For J := 0 To CompNum - 1 Do
Begin
    ReportInfo.Add(FileName);
    CompInfo := ALibCompReader.GetState_ComponentInfos[J];
    ReportInfo.Add(' Name : '         + CompInfo.CompName);
    ReportInfo.Add('  Alias Name : '  + CompInfo.AliasName);
    ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.PartCount));
    ReportInfo.Add('  Description : ' + CompInfo.Description);
    ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.Offset));
    ReportInfo.Add('');
End;
```

See also

IComponentInfo interface

PartCount property

(IComponentInfo interface)

Syntax

```
Property PartCount : Integer Read GetState_PartCount;
```

Description

Example

```
// Obtain the number of components in the specified sch library.
CompNum := ALibCompReader.NumComponentInfos;

// Go thru each component obtained by the LibCompReader interface.
For J := 0 To CompNum - 1 Do
Begin
    ReportInfo.Add(FileName);
    CompInfo := ALibCompReader.ComponentInfos[J];
    ReportInfo.Add(' Name : '         + CompInfo.CompName);
    ReportInfo.Add('  Alias Name : '  + CompInfo.AliasName);
    ReportInfo.Add('  Part Count : '  + IntToStr(CompInfo.PartCount));
    ReportInfo.Add('  Description : ' + CompInfo.Description);
    ReportInfo.Add('  Offset : '      + IntToStr(CompInfo.Offset));
    ReportInfo.Add('  Filename : '      + CompInfo.Filename);
    ReportInfo.Add('');
End;
```

See also

IComponentInfo interface

IComponentPainterView Interface

Overview

IComponentPainterView Methods and Properties Table

IComponentPainterView methods	IComponentPainterView properties
HideComponentTextualDescriptions;	
HighLightComponentPins	
RegisterListener	
RenameSpecifiedPins	
SetComponent	
SetComponentByHandle	
ShowAllPins	
ShowPinsAsSelected	
ShowSpecifiedPinsOnly	

See also

ISch_ServerInterface interface

IComponentMetafilePainter interface

IDocumentPainterView interface

IComponentPainterView Methods

SetComponent method

(IComponentPainterView interface)

Syntax

```
Procedure SetComponent(LibReference, LibraryPath : WideString; APartIndex: Integer);
```

Description

The SetComponent procedure sets the ComponentPainter object to display the specific part of a component from the library with the specified library path. Note a component can be a multi-part component and the first part is numbered 1 and so on.

A component painter object can also be set with the component's handle of ISch_Component type.

Example

```
// display Schematic model on the 3d panel
// cLibraryPath_Sch = 'C:\Program Files\Altium Designer\Developer Kit\Examples\Sch\View Models\Xilinx CoolRunner II.SchLib';
// cLibraryReference_Sch = 'XC2C32-3CP56C';

FExternalFormComponent_Sch.Visible := True;
ComponentPainter := FExternalForm_Sch As IComponentPainterView;
ComponentPainter.SetComponent(cLibraryReference_Sch, cLibraryPath_Sch, 1);
```

See also

IComponentPainterView interface

ViewModel server example in `\Developer Kit\Examples\Sch\ViewModel` folder of SDK installation.

SetComponentByHandle method

(IComponentPainterView interface)

Syntax

```
Procedure SetComponentByHandle(AHandle : ISch_Component; APartIndex : Integer);
```

Description

The SetComponentByHandle procedure sets the ComponentPainter object to display the specific part of a component. Note a component can be a multi-part component and the first part is numbered 1 and so on.

A component painter object can also be set with the full path to a library and its component.

Example

```
FExternalFormComponent_Sch.Visible := True;
ComponentPainter := FExternalForm_Sch As IComponentPainterView;
ComponentPainter.SetComponent(ACompHandle, 1);
```

See also

IComponentPainterView interface

CreateComponentPainter method

SetComponent method

IExternalForm interface in RT_ClientServerInterface unit.

TExternalFormComponent in ExternalForms unit.

HighLightComponentPins method

(IComponentPainterView interface)

Syntax

```
Procedure HighLightComponentPins(APinNameList : WideString; AHighlightColor : TColor; ANonHighlightColor : TColor);
```

Description

Example

See also

IComponentPainterView interface

ShowSpecifiedPinsOnly method

(IComponentPainterView interface)

Syntax

```
Procedure ShowSpecifiedPinsOnly(APinNameList : WideString);
```

Description

Example

See also

IComponentPainterView interface

ShowAllPins method

(IComponentPainterView interface)

Syntax

```
Procedure ShowAllPins;
```

Description

Example

See also

IComponentPainterView interface

RenameSpecifiedPins method

(IComponentPainterView interface)

Syntax

```
Procedure RenameSpecifiedPins(APinNamesParam : WideString);
```

Description

Example

See also

IComponentPainterView interface

HideComponentTextualDescriptions method

(IComponentPainterView interface)

Syntax

```
Procedure HideComponentTextualDescriptions;
```

Description

Example

See also

IComponentPainterView interface

ShowPinsAsSelected method

(IComponentPainterView interface)

Syntax

```
Procedure ShowPinsAsSelected(APinNameList : WideString);
```

Description

Example

See also

IComponentPainterView interface

RegisterListener method

(IComponentPainterView interface)

Syntax

```
Procedure RegisterListener  (APinSelectionListener : IComponentPinSelectionListener);
```

Description

Example

See also

IComponentPainterView interface

IComponentPinSelectionListener Interface

Overview

This is for internal use.

IComponentPinSelectionListener methods **IComponentPinSelectionListener** properties

ComponentPinSelectionChanged

See also

ISch_ServerInterface interface

IComopnentPainterView interface

Methods

ComponentPinSelectionChanged method

(IComponentPinSelectionListener interface)

Syntax

```
Procedure (NewPinSelectionList : WideString);
```

Description

This is for internal use.

Example

See also

IComponentPinSelectionListener interface

IComponentMetafilePainter

Overview

The IComponentMetaFilePainter interface is an internal interface that provides a mechanism to generate images into library reports within the Schematic Library Editor.

The IComponentMetafilePainter interface hierarchy is as follows;

IComponentMetafilePainter methods	IComponentMetafilePainter properties
SetComponent	
DrawToMetafile	

See also

ISch_ServerInterface interface

IComponentPainterView interface

IComponentMetafilePainter interface

Methods

DrawToMetafile method

(IComponentMetafilePainter interface)

Syntax

```
Procedure DrawToMetafile(APartIndex : Integer; APaintColorMode : TPaintColorMode;AScaleMode : TPaintScaleMode; Const AFileName : WideString);
```

Description

This is for internal use.

Example

See also

IComponentMetafilePainter interface

TPaintColorMode type

TPaintScaleMode type

SetComponent method

(IComponentMetafilePainter interface)

Syntax

```
Procedure SetComponent (Const ALibReference, ALibraryPath : WideString);
```

Description

This is for internal use.

Example

See also

IComponentMetafilePainter interface

IDocumentPainterView Interface

Overview

The IDocumentPainterView interface is an internal interface for the Schematic Editor and it represents the Mini Viewer facility. This is for internal use.

IDocumentPainterView methods	IDocumentPainterView properties

DrawCurrentZoomRectangle_Invert
PaintSingleObject
Redraw
Refresh
RefreshCurrentZoomWindow

SetState_ClickHandler
SetState_DbleClickHandler
SetState_DocumentToPaint
SetState_MouseMoveOverLocationHandler

See also
ISch_ServerInterface interface
IComponentPainterView interface
IComponentMetafilePainter interface

Methods

SetState_MouseMoveOverLocationHandler method

(IDocumentPainterView interface)

Syntax

```
Procedure SetState_MouseMoveOverLocationHandler(ALocationProcedure : TLocationProcedure);
```

Description

This is for internal use.

Example

See also
IDocumentPainterView interface

SetState_DocumentToPaint method

(IDocumentPainterView interface)

Syntax

```
Procedure SetState_DocumentToPaint(Const ADocument : ISch_Document);
```

Description

This is for internal use.

Example

See also
IDocumentPainterView interface

SetState_DbleClickHandler method

(IDocumentPainterView interface)

Syntax

```
Procedure SetState_DbleClickHandler (ALocationProcedure : TLocationProcedure);
```

Description

This is for internal use.

Example

See also

IDocumentPainterView interface

SetState_ClickHandler method

(IDocumentPainterView interface)

Syntax

```
Procedure SetState_ClickHandler (ALocationProcedure : TLocationProcedure);
```

Description

This is for internal use.

Example

See also

IDocumentPainterView interface

RefreshCurrentZoomWindow method

(IDocumentPainterView interface)

Syntax

```
Procedure RefreshCurrentZoomWindow;
```

Description

This is for internal use.

Example

See also

IDocumentPainterView interface

Refresh method

(IDocumentPainterView interface)

Syntax

```
Procedure Refresh;
```

Description

This is for internal use.

Example

See also

IDocumentPainterView interface

Redraw method

(IDocumentPainterView interface)

Syntax

```
Procedure Redraw (Const AGraphicalObject : ISch_GraphicalObject);
```

Description

This is for internal use.

Example

See also

IDocumentPainterView interface

PaintSingleObject method

(IDocumentPainterView interface)

Syntax

```
Procedure PaintSingleObject (Const AGraphicalObject : ISch_GraphicalObject);
```

Description

This is for internal use.

Example

See also

IDocumentPainterView interface

DrawCurrentZoomRectangle_Invert method

(IDocumentPainterView interface)

Syntax

```
Procedure DrawCurrentZoomRectangle_Invert;
```

Description

This is for internal use.

Example

See also

IDocumentPainterView interface

Component Mapping Interfaces

ISch_MapDefiner

Overview

The `ISch_MapDefiner` interface represents the object that is used to define a mapping between schematic pins of a schematic component and its model for example the associated PCB pad objects of the PCB component in the same PCB project.

This interface is part of the `ISch_Implementation` interface. Each component can have a number of implementations (models of the same type and/or different types as well).

The `ISch_Implementation.DefinerByInterfaceDesignator` returns you `theISch_MapDefiner` interface with the Designator string representing the component's designator text string.

Notes

A model represents all the information needed for a component in a given domain, while a datafile entity (or link) is the only information which is in an external file.

A model can be represented by external data sources called data file links. For example, pins of a component can have links to different data files, as for signal integrity models. We will consider each model type in respect to the data file links for the main editor servers supported in Altium Designer.

For the PCB footprints, the model and the data file are both the same.

With the simulation models, you can have a simulation model which is a 4ohm resistor for example, there is a simulation model but there is no information is coming from an external file, therefore, a no external file is needed for this as the resistor model is built from spice. This is the case where you have a model with no data file entity. Thus the parameters are used for these types of simulation models that don't have data file links.

With signal integrity models, it can have information required for each pin. If we used IBIS datafiles, not the Altium Designer's central database, then each signal integrity model would then have multiple data files, each one for each type of pin.

The **ISch_MapDefiner** interface hierarchy is as follows;

ISch_MapDefiner methods

GetState_Designator_Implementation
GetState_Designator_ImplementationCount
GetState_Designator_Interface
GetState_Designators_Implementation_AsString
GetState_IsTrivial

SetState_AllFromString
SetState_Designator_ImplementationAdd
SetState_Designator_ImplementationClear
SetState_Designator_Interface

ISch_MapDefiner properties

Designator_Interface
Designator_ImplementationCount
Designator_Implementation
Designator_Implementations_AsString
IsTrivial

See also

ISch_BasicContainer interface
ISch_Component interface
ISch_Implementation interface

Methods

GetState_Designator_Implementation method

(ISch_MapDefiner interface)

Syntax

```
Function GetState_Designator_Implementation(Index : Integer) : WideString;
```

Description

Example

See also

ISch_MapDefiner interface

GetState_Designator_ImplementationCount method

(ISch_MapDefiner interface)

Syntax

```
Function GetState_Designator_ImplementationCount : Integer;
```

Description

Example

See also

ISch_MapDefiner interface

GetState_Designator_Interface method

(ISch_MapDefiner interface)

Syntax

```
Function GetState_Designator_Interface : WideString;
```

Description

Example

See also

ISch_MapDefiner interface

SetState_AllFromString method

(ISch_MapDefiner interface)

Syntax

```
Procedure SetState_AllFromString (AValue : WideString);
```

Description

Example

See also

ISch_MapDefiner interface

SetState_Designator_ImplementationAdd method

(ISch_MapDefiner interface)

Syntax

```
Procedure SetState_Designator_ImplementationAdd(AValue : WideString);
```

Description

Example

See also

ISch_MapDefiner interface

SetState_Designator_Interface method

(ISch_MapDefiner interface)

Syntax

```
Procedure SetState_Designator_Interface(AValue : WideString);
```

Description

Example

See also

ISch_MapDefiner interface

SetState_Designator_ImplementationClear method

(ISch_MapDefiner interface)

Syntax

```
Procedure SetState_Designator_ImplementationClear;
```

Description

Example

See also

ISch_MapDefiner interface

GetState_IsTrivial method

(ISch_MapDefiner interface)

Syntax

```
Function GetState_IsTrivial : Boolean;
```

Description

This function determines whether the mapping is trivial or not. Basically the mapping is trivial if there is no other possible mappings. For example if there is only 1 schematic pin and one PCB pad then the map is trivial.

This function is used by the IsTrivial property.

Example

See also

ISch_MapDefiner interface

GetState_Designators_Implementation_AsString method

(ISch_MapDefiner interface)

Syntax

```
Function GetState_Designators_Implementation_AsString : WideString;
```

Description

Example

See also

ISch_MapDefiner interface

Properties

Designator_Implementations_AsString property

(ISch_MapDefiner interface)

Syntax

```
Property Designator_Implementations_AsString : WideString Read
GetState_Designators_Implementation_AsString;
```

Description

Example

See also

ISch_MapDefiner interface

IsTrivial property

(ISch_MapDefiner interface)

Syntax

```
Property IsTrivial : Boolean Read GetState_IsTrivial;
```

Description

This property determines whether the mapping is trivial or not. Basically the mapping is trivial if there is no other possible mappings. For example if there is only 1 schematic pin and one PCB pad then the map is trivial.

This property implements the GetState_IsTrivial method.

Example

See also

ISch_MapDefiner interface

Designator_Interface property

(ISch_MapDefiner interface)

Syntax

```
Property Designator_Interface : WideString Read GetState_Designator_Interface Write
SetState_Designator_Interface;
```

Description

Example

See also

ISch_MapDefiner interface

Designator_ImplementationCount property

(ISch_MapDefiner interface)

Syntax

```
Property Designator_ImplementationCount : Integer Read
GetState_Designator_ImplementationCount;
```

Description

Example

See also

ISch_MapDefiner interface

Designator_Implementation property

(ISch_MapDefiner interface)

Syntax

```
Property Designator_Implementation[i : Integer] : WideString Read
GetState_Designator_Implementation;
```

Description

Example

See also

ISch_MapDefiner interface

ISch_ModelDatafileLink Interface

Overview

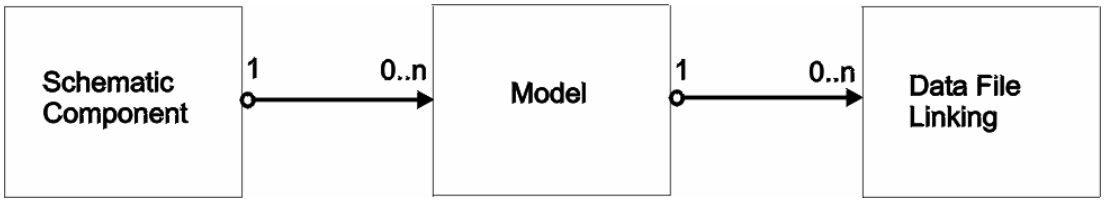
A model represents all the information needed for a component in a given domain, while a datafile entity (or link) is the only information which is in an external file. A model can be represented by external data sources called data file links. For example, pins of a component can have links to different data files, as for signal integrity models. We will consider each model type in respect to the data file links for the editor servers.

For the PCB footprints, the model and the data file are both the same.

With the simulation models, you can have a simulation model which is a 4ohm resistor for example, there is a simulation model here, but there is no information is coming from an external file, therefore, a no external file is needed for this as the resistor model is built from spice. This is the case where you have a model with no data file entity. Thus the parameters are used for these types of simulation models that don't have data file links.

With signal integrity models, it can have information required for each pin. If we used IBIS datafiles, not the Altium Designer's central database, then each signal integrity model would then have multiple data files, each one for each type of pin.

A diagram of the relationship between a component and its models



ISch_ModelDatafileLink methods

GetState_EntityName
GetState_FileKind
GetState_Location

SetState_EntityName
SetState_FileKind
SetState_Location

ISch_ModelDatafileLink properties

EntityName
FileKind
Location

See also

ISch_Component interface
ISch_Implementation interface

Methods

GetState_EntityName method

(ISch_ModelDatafileLink interface)

Syntax

```
Function GetState_EntityName : WideString;
```

Description

Example

See also

ISch_ModelDatafileLink interface

GetState_FileKind method

(ISch_ModelDatafileLink interface)

Syntax

```
Function GetState_FileKind : WideString;
```

Description

Example

See also

ISch_ModelDatafileLink interface

GetState_Location method

(ISch_ModelDatafileLink interface)

Syntax

```
Function GetState_Location : WideString;
```

Description

Example

See also

ISch_ModelDatafileLink interface

SetState_EntityName method

(ISch_ModelDatafileLink interface)

Syntax

```
Procedure SetState_EntityName(AValue : WideString);
```

Description

Example

See also

ISch_ModelDatafileLink interface

SetState_FileKind method

(ISch_ModelDatafileLink interface)

Syntax

```
Procedure SetState_FileKind (AValue : WideString);
```

Description

Example

See also

ISch_ModelDatafileLink interface

SetState_Location method

(ISch_ModelDatafileLink interface)

Syntax

```
Procedure SetState_Location (AValue : WideString);
```

Description

Example

See also

ISch_ModelDatafileLink interface

Properties

EntityName property

(ISch_ModelDatafileLink interface)

Syntax

```
Property EntityName : WideString Read GetState_EntityName Write SetState_EntityName;
```

Description

Example

See also

ISch_ModelDatafileLink interface

FileKind property

(ISch_ModelDatafileLink interface)

Syntax

```
Property FileKind : WideString Read GetState_FileKind Write SetState_FileKind ;
```

Description

Example

See also

ISch_ModelDatafileLink interface

Location property

(ISch_ModelDatafileLink interface)

Syntax

```
Property Location : WideString Read GetState_Location Write SetState_Location ;
```

Description

Example

See also

ISch_ModelDatafileLink interface

ISch_Implementation Interface

Overview

Each schematic component can have models from one or more domains. A schematic component can also have multiple models per domain, one of which will be the current model for that domain.

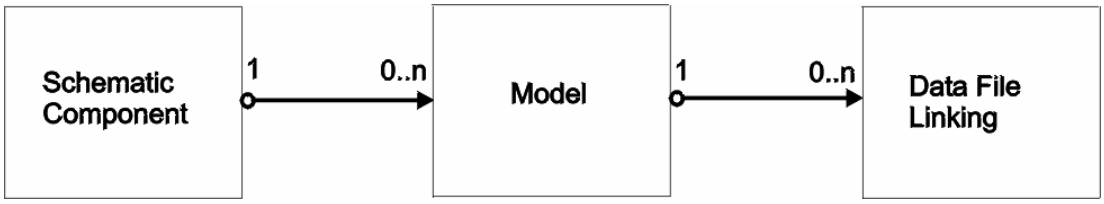
A model represents all the information needed for a component in a given domain, while a datafile entity (or link) is the only information which is in an external file.

The models of a component are represented by the **ISch_Implementation** interface.

The mapping of pins of a component and the nodes/pads of a model are represented by the **ISch_MapDefiner** interfaces.

The link between a model and its external data file links are represented by the **ISch_DataFileLink** interfaces.

A diagram of the relationship between a component and its models



Notes

A model can be represented by external data sources called data file links. For example, pins of a component can have links to different data files, as for signal integrity models. We will consider each model type in respect to the data file links for the main editor servers supported in Altium Designer.

For the PCB footprints, the model and the data file are both the same.

With the simulation models, you can have a simulation model which is a 4ohm resistor for example, there is a simulation model here, but there is no information is coming from an external file, therefore, a no external file is needed for this as the resistor model is built from spice. This is the case where you have a model with no data file entity. Thus the parameters are used for these types of simulation models that don't have data file links.

With signal integrity models, it can have information required for each pin. If we used IBIS datafiles, not the Altium Designer's central database, then each signal integrity model would then have multiple data files, each one for each type of pin.

A model can also be called an implementation. Each implementation linked to a component can have parameters and data file links.

ISch_Implementation methods

AddDataFileLink
ClearAllDatafileLinks
LockImplementation
Map_Import_FromUser

GetState_DatabaseDatalinksLocked
GetState_DatabaseModel
GetState_DatafileLinkCount
GetState_DatalinksLocked
GetState_Description

ISch_Implementation properties

DatabaseDatalinksLocked
DatabaseModel
DatafileLink
DatafileLinkCount
DatalinksLocked
DefinerByInterfaceDesignator
Description
IntegratedModel
IsCurrent
MapAsString
ModelName

GetState_IntegratedModel ModelType
GetState_IsCurrent
GetState_MapAsString
GetState_ModelName
GetState_ModelType
GetState_SchDatafileLink
GetState_SchDefinerByInterfaceDesignator

SetState_DatabaseDatalinksLocked
SetState_DatalinksLocked
SetState_DatabaseModel
SetState_Description
SetState_IntegratedModel
SetState_IsCurrent
SetState_MapAsString
SetState_ModelName
SetState_ModelType

See also
ISch_MapDefiner interface
ISch_ModelDatafileLink interface

Methods

AddDataFileLink method

(ISch_Implementation interface)
Syntax
```
Procedure AddDataFileLink(anEntityName, aLocation, aFileKind : WideString);
```
Description

Example

See also
ISch_Implementation interface

ClearAllDatafileLinks method

(ISch_Implementation interface)
Syntax
```
Procedure ClearAllDatafileLinks;
```
Description
This procedure removes all the data file links of the implementation (model) for the current component.
Example

See also
ISch_Implementation interface

LockImplementation method

(ISch_Implementation interface)

Syntax

```
Procedure LockImplementation;
```

Description

Example

See also

ISch_Implementation interface

Map_Import_FromUser method

(ISch_Implementation interface)

Syntax

```
Function Map_Import_FromUser (AlowOneToMany : Boolean): Boolean;
```

Description

Example

See also

ISch_Implementation interface

Properties

DatafileLinkCount property

(ISch_Implementation interface)

Syntax

```
Property DatafileLinkCount : Integer Read GetState_DatafileLinkCount;
```

Description

This property fetches the number of data file links for the current implementation of the schematic component.

This property is supported by the GetState_DatafileLinkCount function.

Example

```
For j := 0 To SchImplementation.DatafileLinkCount - 1 Do
Begin
    ModelDataFile := SchImplementation.DatafileLink[j];
    If ModelDataFile <> Nil Then
    Begin
        ModelsList.Add('   Implementation Data File Link Details:');
        ModelsList.Add('   Data File Location: ' + ModelDataFile.Location +
                       ', Entity Name: '         + ModelDataFile.EntityName +
                       ', FileKind: '            + ModelDataFile.FileKind);
        ModelsList.Add('');
    End;
End;
```

See also

ISch_Implementation interface

DataFileLink property

DatabaseModel property

(ISch_Implementation interface)

Syntax

```
Property DatabaseModel : Boolean Read GetState_DatabaseModel Write SetState_DatabaseModel;
```

Description

This property is implemented by the GetState_DatabaseModel and SetState_DatabaseModel methods.

Example

See also

ISch_Implementation interface

IntegratedModel property

DatafileLink property

(ISch_Implementation interface)

Syntax

```
Property DatafileLink [i : Integer] : ISch_ModelDatafileLink Read GetState_SchDatafileLink;
```

Description

The `DatafileLink` property determines the indexed datafilelink of the model type linked to the component. A component can have multiple linked models and each model can have multiple external data file links.

This property is implemented with the `GetState_SchDatafileLink(i : Integer) : ISch_ModelDatafileLink` method.

Example

```
For j := 0 To SchImplementation.DatafileLinkCount - 1 Do
Begin
    ModelDataFile := SchImplementation.DatafileLink[j];
    If ModelDataFile <> Nil Then
    Begin
        ModelsList.Add('   Implementation Data File Link Details:');
        ModelsList.Add('   Data File Location: ' + ModelDataFile.Location +
                       ', Entity Name: '         + ModelDataFile.EntityName +
                       ', FileKind: '            + ModelDataFile.FileKind);
        ModelsList.Add('');
    End;
End;
```

See also

ISch_Implementation interface

DatalinksLocked property

(ISch_Implementation interface)

Syntax

```
Property DatalinksLocked : Boolean Read GetState_DatalinksLocked Write
SetState_DatalinksLocked;
```

Description

Example

See also

ISch_Implementation interface

DefinerByInterfaceDesignator property

(ISch_Implementation interface)

Syntax

```
Property DefinerByInterfaceDesignator[S : WideString] : ISch_MapDefiner Read
GetState_SchDefinerByInterfaceDesignator;
```

Description

Example

See also

ISch_Implementation interface

Description property

(ISch_Implementation interface)

Syntax

```
Property Description : WideString Read GetState_Description Write SetState_Description ;
```

Description

The `Description` property fetches or sets the `Description` string for the model. This is optional and is for reference purposes and do not have any impact on simulation processes. This property is implemented by the `GetState_Description : WideString` and `SetState_Description(AValue : WideString)` methods.

Example

```
SchImplementation := ImplIterator.FirstSchObject;
While SchImplementation <> Nil Do
Begin
    ShowMessage   ('   ModelName: ' + SchImplementation.ModelName +
                   ' ModelType: '   + SchImplementation.ModelType +
                   ' Description: ' + SchImplementation.Description);
End;
```

See also

ISch_Implementation interface

IntegratedModel property

(ISch_Implementation interface)

Syntax

```
Property IntegratedModel : Boolean Read GetState_IntegratedModel Write
SetState_IntegratedModel;
```

Description

The property determines whether the implementation is an integrated model type or not.

Example

See also

ISch_Implementation interface

DatabaseModel property

IsCurrent property

(ISch_Implementation interface)

Syntax

```
Property IsCurrent : Boolean Read GetState_IsCurrent Write SetState_IsCurrent ;
```

Description

Example

See also

ISch_Implementation interface

MapAsString property

(ISch_Implementation interface)

Syntax

```
Property MapAsString : WideString Read GetState_MapAsString Write SetState_MapAsString ;
```

Description

This `MapAsString` property returns or sets the map of the component pins to a model pins (simulation ports for example) as a string of the following format: (SchematicPinNumber:ModelPinNumber) for example `(1:1),(2:2), … ,(X:X)`

Example

See also

ISch_Implementation interface

ModelName property

(ISch_Implementation interface)

Syntax

```
Property ModelName : WideString Read GetState_ModelName Write SetState_ModelName ;
```

Description

The `ModelName` property fetches or sets the name of the indexed model name.This property is implemented with `GetState_ModelName : WideString` and `SetState_ModelName(AValue : WideString)` methods.

Example

```
Result := IntegratedLibraryManager.ModelName(Component.LibReference,PathToLibrary,'SIM',0);
```

See also

ISch_Implementation interface

ModelType property

(ISch_Implementation interface)

Syntax

```
Property ModelType : WideString Read GetState_ModelType Write SetState_ModelType ;
```

Description

Example

See also

ISch_Implementation interface

UseComponentLibrary

(ISch_Implementation interface)

Syntax

```
Property UseComponentLibrary     : Boolean    Read GetState_UseComponentLibrary     Write
SetState_UseComponentLibrary;
```

Description

This `UseComponentLibrary` property determines whether the component is from an integrated library or not (either as an installed library or part of the Project Libraries. This is accessed from the *Available Libraries* dialog in Altium Designer). A

Boolean value is returned. This property is implemented with `GetState_UseComponentLibrary : Boolean` and `SetState_UseComponentLibrary(AValue : Boolean)` methods.

Example

See also

ISch_Implementation interface

Schematic Design Objects

A schematic design object on a schematic document is represented by its interface. An interface represents an existing object in memory and its properties and methods can be invoked.

Since many design objects are descended from ancestor interfaces and thus the ancestor methods and properties are also available to use. For example the ISch_Image interface is inherited from an immediate ISch_Rectangle interface and in turn inherited from the ISch_GraphicalObject interface. If you check the ISCh_Image entry in this online help you will see the following information;

The ISch_Image interface hierachy is as follows;

ISch_GraphicalObject

ISch_Rectangle

ISch_Image

ISch_Rectangle properties

```
Corner    : TLocation
LineWidth : TSize
IsSolid   : Boolean
```

ISch_Image Properties

```
EmbedImage : Boolean
FileName   : WideString
KeepAspect : Boolean
```

Therefore you have the Image object properties, along with ISch_Rectangle methods and properties AND ISch_GraphicalObject methods and properties as well to use in your scripts.

ISch_Arc Interface

Overview

An arc object is a circular curve used to place on the schematic sheet.

Notes

The ISch_Arc interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Arc

ISch_Arc methods

GetState_Radius
GetState_StartAngle
GetState_EndAngle
GetState_LineWidth
SetState_Radius
SetState_StartAngle
SetState_EndAngle
SetState_LineWidth

ISch_Arc properties

Radius
StartAngle
EndAngle
LineWidth

See also

Methods

All methods are implemented by the ISch_Arc properties. More information for each property of the ISch_Arc interface is presented in the Properties section.

Properties

StartAngle property

(ISch_Arc interface)

Syntax

```
Property StartAngle : TAngle Read GetState_StartAngle Write SetState_StartAngle;
```

Description

This property defines the start angle of the arc in degrees from the horizontal. The arc is drawn in an anti-clockwise direction from the start angle to the end angle. The value can be between -360 to 360 to define the start angle directly.

Example

See also

ISch_Arc interface

TAngle type

Radius property

(ISch_Arc interface)

Syntax

```
Property Radius : TDistance Read GetState_Radius Write SetState_Radius ;
```

Description

The Radius property defines the radius of the arc. This property is supported by the GetState_Radius and SetState_Radius methods.

Example

See also

ISch_Arc interface

TDistance type

LineWidth property

(ISch_Arc interface)

Syntax

```
Property LineWidth : TSize Read GetState_LineWidth Write SetState_LineWidth ;
```

Description

The LineWidth property defines the border width of the arc with one of the following values from the TSize enumerated type. This property is supported by the GetState_LineWidth and SetState_LineWidth methods.

Example

```
Arc.LineWidth := eMedium;
```

See also

TSize Type

ISch_Arc interface

EndAngle property

(ISch_Arc interface)

Syntax

```
Property EndAngle : TAngle Read GetState_EndAngle Write SetState_EndAngle ;
```

Description

This property defines the end angle of the arc in degrees from the horizontal. The arc is drawn in an anti-clockwise direction from the start angle to the end angle. The value can be between -360 to 360 to define the end angle directly.

Example

See also

ISch_Arc interface

TAngle type

ISch_Bezier Interface

Overview

A bezier curve is used to create curved line shapes (For example a section of a sine wave or a pulse). At least four points are required to define a bezier curve. More than four points used will define another bezier curve and so on.

The ISch_Bezier interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Polygon

ISch_BasicPolyline

ISch_Bezier

ISch_Bezier methods **ISch_Bezier properties**

See also

ISch_Bus Interface

Overview

Buses are special graphical elements that represent a common pathway for multiple signals on a schematic document. Buses have no electrical properties, and they must be correctly identified by net labels and ports.

Notes

The ISch_Bus interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Polygon

ISch_Polyline

ISch_Wire

ISch_Bus

Note that the ISch_Wire interface has no extra properties and methods but has inherited properties and methods only.

ISch_Bus methods **ISch_Bus properties**

See also

ISch_Wire

ISch_Polyline

ISCh_Polygon

ISch_GraphicalObject

ISch_BusEntry Interface

Overview

A bus entry is a special wire at an angle of 45 degrees which is used to connect a wire to the bus line.

The ISch_BusEntry interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Line

ISch_BusEntry

ISch_BusEntry methods

ISch_BusEntry properties

See also

ISch_Line interface

ISch_Circle Interface

Overview

A circle is a closed arc object.

The ISch_Circle interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Circle

ISch_Circle methods

SetState_LineWidth
SetState_IsSolid
SetState_Radius
SetState_Transparent
GetState_LineWidth
GetState_IsSolid
GetState_Radius
GetState_Transparent

ISch_Circle properties

LineWidth
IsSolid
Radius
Transparent

See also

ISch_GraphicalObject interface

TSize type

TDistance type

Methods

All methods are implemented by the ISch_Circle properties. More information for each property of the ISch_Circle interface is presented in the Properties section.

Properties

LineWidth property

(ISch_Circle interface)

Syntax

```
Property LineWidth : TSize Read GetState_LineWidth Write SetState_LineWidth;
```

Description

The LineWidth property defines the border width of the circle with one of the following values from the TSize enumerated type. This property is supported by the GetState_LineWidth and SetState_LineWidth methods.

Example

```
Circle.LineWidth := eLarge;
```

See also

TSize type.

ISch_Circle interface

IsSolid property

(ISch_Circle interface)

Syntax

```
Property IsSolid : Boolean Read GetState_IsSolid Write SetState_IsSolid;
```

Description

This property defines whether the circle is to be filled inside or not. If it is true, the circle is filled with the color set by the AreaColor property (from its ancestor ISch_GraphicalObject interface).

This property is supported by the GetState_IsSolid and SetState_IsSolid methods.

Example

```
If Circle.IsSolid Then
    Circle.AreaColor := 0; // black fill.
```

See also

ISch_Circle interface

Radius property

(ISch_Circle interface)

Syntax

```
Property Radius : TDistance Read GetState_Radius Write SetState_Radius;
```

Description

The Radius property defines the radius of the circle (pie chart). This property is supported by the GetState_Radius and SetState_Radius methods.

Example

See also

ISch_Circle interface

TDistance type

Transparent property

(ISch_Circle interface)

Syntax

```
Property Transparent : Boolean Read GetState_Transparent Write SetState_Transparent;
```

Description

This transparent property toggles the transparency of this circle object. This property is supported by the GetState_Transparent and SetState_Transparent methods.

Example

See also

ISch_Circle interface

ISch_CompileMask Interface

Overview

A compile mask is used to effectively hide the area of the design within the PCB project it contains from the Compiler, allowing you to manually prevent error checking for circuitry that may not yet be complete and you know will generated compile errors.

This can prove very useful if you need to compile the active document or project to check the integrity of the design in other specific areas, but do not want the clutter of compiler-generated messages associated with unfinished portions of the design.

The CompileMask object hold multiple lines of free text that can be collapsed or not.

The ISch_CompileMask interface hierarchy is as follows;

ISch_TextFrame interface hierarchy is as follows;

ISch_GraphicalObject

    ISch_Rectangle

        ISch_CompileMask

ISch_CompileMask methods

SetState_Collapsed

GetState_Collapsed

ISch_CompileMask properties

Collapsed

See also

ISch_Rectangle interface

Methods

All methods are implemented by the ISch_CompileMask properties. More information for each property of the ISch_CompileMask interface is presented in the Properties section.

Properties

Collapsed property

(ISch_CompileMask interface)

Syntax

```
Property Collapsed : Boolean Read GetState_Collapsed Write SetState_Collapsed;
```

Description

When the property is false, the compile mask is collapsed and disabled. When this property is true, the compile mask is fully expanded and enabled meaning the portion of the schematic covered by the Compile Mask object is not affected by the Compiler.

This property is supported by the GetState_Collapsed and SetState_Collapsed methods.

Example

See also

ISch_CompileMask interface

ISch_ComplexText Interface

Overview

An immediate ancestor interface for ISch_SheetFilename and ISch_SheetName interfaces.

The ISch_ComplexText interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Label

ISch_ComplexText

ISch_ComplexText methods

SetState_Autoposition

SetState_IsHidden

SetState_TextHorzAnchor

SetState_TextVertAnchor

GetState_Autoposition

GetState_IsHidden

GetState_TextHorzAnchor

GetState_TextVertAnchor

ISch_ComplexText properties

Autoposition

IsHidden

TextHorzAnchor

TextVertAnchor

See also

Methods

GetState_Autoposition method

(ISch_ComplexText interface)

Syntax

```
Function GetState_Autoposition : Boolean;
```

Description

The property defines whether the parameter can be positioned automatically every time the associated component is rotated or moved. If this property is false, the parameter will have a dot appear below it on the schematic to denote that this parameter will not be auto positioned everytime the component is rotated/moved.

The function reads the autoposition value and is used for the Autoposition property.

To prevent dots form being displayed, disable the MarkManualParameters property from the ISch_Preferences interface.

Example

See also

ISch_ComplexText interface

GetState_IsHidden method

(ISch_ComplexText interface)

Syntax

```
Function GetState_IsHidden : Boolean;
```

Description

The property determines whether the text object is hidden or not. This method obtains the boolean value whether the complex text (a parameter object) is hidden or not and is used in the IsHidden property.

Example

See also

ISch_ComplexText interface

GetState_TextVertAnchor method

(ISch_ComplexText interface)

Syntax

```
Function GetState_TextVertAnchor : TTextVertAnchor;
```

Description

The TextVertAnchor property defines the vertical justification style of the parameter object.

The method obtains the vertical justification style of the object represented by the ISch_ComplexText interface and is used for the TextVertAnchor property.

Example

See also

ISch_ComplexText interface

TTextVertAnchor type

GetState_TextHorzAnchor method

(ISch_ComplexText interface)

Syntax

```
Function GetState_TextHorzAnchor : TTextHorzAnchor;
```

Description

The TextHorzAnchor property defines the horizontal justification style of the parameter object.

The method obtains the horizontal justification style of the object represented by the ISch_ComplexText interface and is used for the TextHorzAnchor property.

Example

See also

ISch_ComplexText interface

SetState_TextVertAnchor method

(ISch_ComplexText interface)

Syntax

```
Procedure SetState_TextVertAnchor (A : TTextVertAnchor);
```

Description

The TextVertAnchor property defines the vertical justification style of the parameter object. The function sets the vertical justification of the parameter object and is used for the TextVertAnchor property.

Example

See also

ISch_ComplexText interface

SetState_TextHorzAnchor method

(ISch_ComplexText interface)

Syntax

```
Procedure SetState_TextHorzAnchor (A : TTextHorzAnchor);
```

Description

The TextHorzAnchor property defines the horizontal justification style of the parameter object.

The method obtains the horizontal justification style of the object represented by the ISch_ComplexText interface and is used for the TextHorzAnchor property.

Example

See also

ISch_ComplexText interface

SetState_IsHidden method

(ISch_ComplexText interface)

Syntax

```
Procedure SetState_IsHidden (B : Boolean);
```

Description

The property determines whether the text object is hidden or not. This method sets the boolean value whether the complex text (a parameter object) is hidden or not and is used in the IsHidden property.

Example

See also

ISch_ComplexText interface

SetState_Autoposition method

(ISch_ComplexText interface)

Syntax

```
Procedure SetState_Autoposition (B : Boolean);
```

Description

The property defines whether the parameter can be positioned automatically every time the associated component is rotated or moved. If this property is false, the parameter will have a dot appear below it on the schematic to denote that this parameter will not be auto positioned everytime the component is rotated/moved.

The procedure sets the value for autoposition of parameters and is used for the Autoposition property.

To prevent dots form being displayed, disable the MarkManualParameters property from the ISch_Preferences interface.

Example

See also

ISch_ComplexText interface

Properties

Autoposition property

(ISch_ComplexText interface)

Syntax

```
Property Autoposition : Boolean Read GetState_Autoposition Write SetState_Autoposition;
```

Description

The property defines whether the parameter can be positioned automatically every time the associated component is rotated or moved. If this property is false, the parameter will have a dot appear below it on the schematic to denote that this parameter will not be auto positioned everytime the component is rotated/moved.

To prevent dots form being displayed, disable the MarkManualParameters property from the ISch_Preferences interface.

Example

See also

ISch_ComplexText interface

IsHidden property

(ISch_ComplexText interface)

Syntax

```
Property IsHidden : Boolean Read GetState_IsHidden Write SetState_IsHidden;
```

Description

The property determines whether the text object is hidden or not. This property is supported by the GetState_IsHidden and SetState_IsHidden methods.

Example

See also

ISch_ComplexText interface

TextVertAnchor property

(ISch_ComplexText interface)

Syntax

```
Property TextVertAnchor : TTextVertAnchor Read GetState_TextVertAnchor Write
SetState_TextVertAnchor;
```

Description

This property defines the vertical justification style of the parameter object. This property is supported by the GetState_TextVertAnchor and SetState_TextVertAnchor methods.

Example

See also

ISch_ComplexText interface

TTextVertAnchor type

TextHorzAnchor property

(ISch_ComplexText interface)

Syntax

```
Property TextHorzAnchor : TTextHorzAnchor Read GetState_TextHorzAnchor Write
SetState_TextHorzAnchor;
```

Description

This property defines the horizontal justification style of the parameter object. This property is supported by the GetState_TextHorzAnchor and SetState_TextHorzAnchor methods.

Example

See also

ISch_ComplexText interface

TTextHorzAnchor type

ISch_Component Interface

Overview

The ISch_Component references the logical symbol as a component that can contain links to different model implementations such as PCB, Signal Integrity and Simulation models. Only one model of a particular model type (PCB footprint, SIM, SI, EDIF Macro and VHDL) can be enabled as the currently linked model, at any one time.

Each schematic component has two system parameters – the Designator parameter and the Comment parameter. Custom parameters can be added anytime. The Comment parameter can be assigned an indirect name parameter. Once a name parameter (with a equal sign character as a prefix to the name parameter) is assigned to the Comment field of the Component properties dialog, the value for this parameter appears on the document, ensure that the Convert Special Strings option in the *Schematic Preferences* dialog is enabled.

The Unique ID (UID) is an system generated value that uniquely identifies this current component. It is used for linking to an associated PCB component on a PCB document. Enter a new UID value or click the Reset button to generate a new UID if you wish to force the Schematic component to be linked to a different PCB component. You will need to run the Component Links... dialog to update the linkage on the corresponding PCB document.

This `SourceLibraryName` property denotes the source library where the symbol and its associated model links are from. The * character in this field denotes the current library of the current project. Note a schematic component is a symbol with a defined designator placed on a schematic document.

The `LibraryRef` property is the name of the symbol. The symbol is from the library specified in the Library field below.

The `SheetPartyFilename` property, enter a sub design project file name to be linked to the current schematic component. An example of a sub design project is a programmable logic device project or a schematic sub-sheet.

Notes

The ISch_Component interface hierarchy is as follows;

ISch_GraphicalObject

ISch_ParametrizedGroup

ISch_Component

ISch_Component methods

GetState_AliasAsText

GetState_AliasAt

GetState_AliasCount

GetState_ComponentDescription

GetState_ComponentKind

GetState_ConfiguratorName

ISch_Component properties

Alias

AliasAsText

AliasCount

Comment

ComponentDescription

ComponentKind

ConfiguratorName

GetState_CurrentPartID
GetState_DatabaseLibraryKeys
GetState_DatabaseLibraryName
GetState_DatabaseTableName
GetState_DesignatorLocked
GetState_DisplayFieldNames
GetState_DisplayMode
GetState_DisplayModeCount
GetState_IsMirrored
GetState_LibraryPath
GetState_LibReference
GetState_Orientation
GetState_OverideColors
GetState_PartCountNoPart0
GetState_PartIdLocked
GetState_PinColor
GetState_PinsMoveable
GetState_SchComment
GetState_SchDesignator
GetState_SheetPartFileName
GetState_ShowHiddenFields
GetState_ShowHiddenPins
GetState_SourceLibraryName
GetState_TargetFileName
GetState_UniqueId

CurrentPartID
DatabaseLibraryName
DatabaseTableName
Designator
DesignatorLocked
DisplayFieldNames
DisplayMode
DisplayModeCount
IsMirrored
LibraryPath
LibReference
Orientation
OverideColors
PartCount
PartIdLocked
PinColor
PinsMoveable
SheetPartFileName
ShowHiddenFields
ShowHiddenPins
SourceLibraryName
TargetFileName
UniqueId

SetState_AliasAsText
SetState_AliasAt
SetState_ComponentDescription
SetState_ComponentKind
SetState_CurrentPartID
SetState_DesignatorLocked
SetState_DisplayFieldNames
SetState_DisplayMode
SetState_DisplayModeCount_Check
SetState_FilePosition
SetState_IsMirrored
SetState_LibraryPath
SetState_LibReference
SetState_Orientation
SetState_OverideColors
SetState_PartCountNoPart0
SetState_PartIdLocked
SetState_PinColor
SetState_PinsMoveable
SetState_SheetPartFileName

SetState_ShowHiddenFields
SetState_ShowHiddenPins
SetState_SourceLibraryName
SetState_TargetFileName
SetState_UniqueId

AddDisplayMode
AddPart
AddSchImplementation
Alias_Add
Alias_Clear
Alias_Delete
Alias_Remove
DeleteDisplayMode
DeletePart
FullPartDesignator
InLibrary
InSheet
IsIntegratedComponent
IsMultiPartComponent
RemoveSchImplementation
UpdatePrimitivesAccessibility

See also

Methods

AddSchImplementation method

(ISch_Component interface)

Syntax

```
Function AddSchImplementation : ISch_Implementation;
```

Description

Each schematic component can have models from one or more domains. A schematic component can also have multiple models per domain, one of which will be the current model for that domain.

A model represents all the information needed for a component in a given domain, while a datafile entity (or link) is the only information which is in an external file.

The models of a component are represented by the `ISch_Implementation` interface.

The mapping of pins of a component and the nodes/ports/pads of a model are represented by the `ISch_MapDefiner` interfaces.

The link between a model and its external data file links are represented by the `ISch_DataFileLink` interfaces.

Example

```
Implementation := Comp.AddSchImplementation;
```

See also

ISch_Component interface
ISch_Implementation interface
`ISch_DataFileLink` interface

`ISch_MapDefiner` interface

AddDisplayMode method

(ISch_Component interface)

Syntax

```
Procedure AddDisplayMode;
```

Description

The AddDisplayMode procedure adds a graphical representation (mode) for the current component. Up to 255 alternative modes can be created.

Example

```
Comp.AddDisplayMode;
```

See also

ISch_Component interface

AddPart method

(ISch_Component interface)

Syntax

```
Procedure AddPart;
```

Description

Example

See also

ISch_Component interface

Alias_Add method

(ISch_Component interface)

Syntax

```
Procedure Alias_Add (S : WideString);
```

Description

Example

See also

ISch_Component interface

Alias_Clear method

(ISch_Component interface)

Syntax

```
Procedure Alias_Clear;
```

Description

Example

See also

ISch_Component interface

Alias_Delete method

(ISch_Component interface)

Syntax

```
Procedure Alias_Delete(i : Integer);
```

Description

Example

See also

ISch_Component interface

Alias_Remove method

(ISch_Component interface)

Syntax

```
Procedure Alias_Remove(S : WideString);
```

Description

Example

See also

ISch_Component interface

DeleteDisplayMode method

(ISch_Component interface)

Syntax

```
Procedure DeleteDisplayMode(AMode : TDisplayMode);
```

Description

This DeleteDisplayMode removes a display mode (graphical representation) from the component.

Example

```
Component.DeleteDisplayMode(3);
```

See also

TDisplayMode type from RT_Workspace unit. Byte type.

ISch_Component interface

DeletePart method

(ISch_Component interface)

Syntax

```
Procedure DeletePart (APartId : Integer);
```

Description

Example

See also

ISch_Component interface

FullPartDesignator method

(ISch_Component interface)

Syntax

```
Function FullPartDesignator(APartId : Integer) : WideString;
```

Description

Example

See also

ISch_Component interface

GetState_AliasAsText method

(ISch_Component interface)

Syntax

```
Function GetState_AliasAsText : WideString;
```

Description

Example

See also

ISch_Component interface

GetState_AliasAt method

(ISch_Component interface)

Syntax

```
Function GetState_AliasAt(i : Integer) : WideString;
```

Description

Example

See also

ISch_Component interface

GetState_AliasCount method

(ISch_Component interface)

Syntax

```
Function GetState_AliasCount : Integer;
```

Description

Example

See also

ISch_Component interface

GetState_ComponentDescription method

(ISch_Component interface)

Syntax

```
Function GetState_ComponentDescription : WideString;
```

Description

The GetState_ComponentDescription function returns the description string for this component. This string is normally used to describe what this component is for.

Example

```
Desc := Component.GetState_ComponentDescription;
```

See also

ISch_Component interface

GetState_ComponentKind method

(ISch_Component interface)

Syntax

```
Function GetState_ComponentKind : TComponentKind;
```

Description

The GetState_ComponentKind function returns a value of TComponentKind for the component.

eComponentKind_Standard: These components possess standard electrical properties, are always synchronized and are the type most commonly used on a schematic sheet.

eComponentKind_Mechanical: These components do not have electrical properties and will appear in the BOM. They are synchronized if the same components exist on both the Schematic and PCB documents. An example is a heatsink.

eComponentKind_Graphical: These components are not used during synchronization or checked for electrical errors. These components are used, for example, when adding company logos to documents.

eComponentKind_NetTie_BOM: These components short two or more different nets and these components will appear in the BOM and are maintained during synchronization.

eComponentKind_NetTie_NoBOM: These components short two or more different nets and these components will NOT appear in the BOM and are maintained during synchronization.

eComponentKind_Standard_NoBOM: These components possess standard electrical properties, and are synchronized BUT are not included in any BOM file produced from the file.

Example

```
Component.GetState_ComponentKind;
```

See also

TComponentKind from RT_Workspace unit.

ISch_Component interface

GetState_CurrentPartID method

(ISch_Component interface)

Syntax

```
Function GetState_CurrentPartID : Integer;
```

Description

Example

See also

ISch_Component interface

GetState_DesignatorLocked method

(ISch_Component interface)

Syntax

```
Function GetState_DesignatorLocked : Boolean;
```

Description

Example

See also

ISch_Component interface

GetState_DisplayFieldNames method

(ISch_Component interface)

Syntax

```
Function GetState_DisplayFieldNames : Boolean;
```

Description

Example

See also

ISch_Component interface

GetState_DisplayMode method

(ISch_Component interface)

Syntax

```
Function GetState_DisplayMode : TDisplayMode;
```

Description

The GetState_DisplayMode function returns the TDisplayMode value for this component. This TDisplayMode is a byte type from RT_Workspace unit.

Example

```
Mode := Comp.GetState_DisplayMode;
```

See also

ISch_Component interface

GetState_DisplayModeCount method

(ISch_Component interface)

Syntax

```
Function GetState_DisplayModeCount : Integer;
```

Description

This GetState_DisplayModeCount procedure returns the number of display modes or graphical representations for this component. There can be up to 255 modes.

Example

```
Count := Comp.GetState_DisplayModeCount;
```

See also

ISch_Component interface

GetState_IsMirrored method

(ISch_Component interface)

Syntax

```
Function GetState_IsMirrored : Boolean;
```

Description

The GetState_IsMirrored function determines whether the component is mirrored along the x-axis or not.

Example

```
Mirrored := Comp.GetState_IsMirrored;
```

See also

ISch_Component interface

GetState_LibraryPath method

(ISch_Component interface)

Syntax

```
Function GetState_LibraryPath : WideString;
```

Description

Example

See also

ISch_Component interface

GetState_LibReference method

(ISch_Component interface)

Syntax

```
Function GetState_LibReference : WideString;
```

Description

Example

See also

ISch_Component interface

GetState_Orientation method

(ISch_Component interface)

Syntax

```
Function GetState_Orientation : TRotationBy90;
```

Description

The Orientation property determines the orientation of the component on the schematic sheet in increments of 0,90,180 and 270 degrees only.

This method obtains the orientation value of the component and is used in the Orientation property.

Example

See also

ISch_Component interface

TRotationBy90 type

GetState_OverideColors method

(ISch_Component interface)

Syntax

```
Function GetState_OverideColors : Boolean;
```

Description

Example

See also

ISch_Component interface

GetState_PartCountNoPart0 method

(ISch_Component interface)

Syntax

```
Function GetState_PartCountNoPart0 : Integer;
```

Description

A component can consist of more than one part, for example a 74LS00 contains four parts. This property returns the number of parts for the component.

The function returns you the number of parts for a component and is used in the PartCountNoPart0 property.

Note

Each component also includes a non-graphical part, Part Zero. Part Zero is used for pins that are to be included in all parts of a multi-part component, for example power pins.

Example

See also

ISch_Component interface

GetState_PartIdLocked method

(ISch_Component interface)

Syntax

```
Function GetState_PartIdLocked : Boolean;
```

Description

Example

See also

ISch_Component interface

GetState_PinColor method

(ISch_Component interface)

Syntax

```
Function GetState_PinColor : TColor;
```

Description

Example

See also

ISch_Component interface

GetState_PinsMoveable method

(ISch_Component interface)

Syntax

```
Function GetState_PinsMoveable : Boolean;
```

Description

Example

See also

ISch_Component interface

GetState_SchComment method

(ISch_Component interface)

Syntax

```
Function GetState_SchComment : ISch_Parameter;
```

Description

The Comment property determines the comment object associated with the component object. The Component Properties dialog for this component has a Comment field. The Parameter object has a Name and Value fields and this Name field will normally have ‘Comment’ string and a Value string.

Example

```
Comp.GetState_SchComment := 'LM833M';
```

See also

ISch_Parameter interface

ISch_Component interface

GetState_SchDesignator method

(ISch_Component interface)

Syntax

```
Function GetState_SchDesignator : ISch_Designator;
```

Description

Example

See also

ISch_Component interface

GetState_SheetPartFileName method

(ISch_Component interface)

Syntax

```
Function GetState_SheetPartFileName : WideString;
```

Description

Example

See also

ISch_Component interface

GetState_ShowHiddenFields method

(ISch_Component interface)

Syntax

```
Function GetState_ShowHiddenFields : Boolean;
```

Description

Example

See also

ISch_Component interface

GetState_ShowHiddenPins method

(ISch_Component interface)

Syntax

```
Function GetState_ShowHiddenPins : Boolean;
```

Description

This property determines whether the hidden pins of a component can be hidden or not. Power pins are often defined as hidden. This method gets the boolean value whether the hidden pins are displayed or not and is used in the ShowHiddenPins property.

Example

See also

ISch_Component interface

GetState_SourceLibraryName method

(ISch_Component interface)

Syntax

```
Function GetState_SourceLibraryName : WideString;
```

Description

Example

See also

ISch_Component interface

GetState_TargetFileName method

(ISch_Component interface)

Syntax

```
Function GetState_TargetFileName : WideString;
```

Description

Example

See also

ISch_Component interface

GetState_UniqueId method

(ISch_Component interface)

Syntax

```
Function GetState_UniqueId : WideString;
```

Description

Example

See also

ISch_Component interface

InLibrary method

(ISch_Component interface)

Syntax

```
Function InLibrary : Boolean;
```

Description

Example

See also

ISch_Component interface

InSheet method

(ISch_Component interface)

Syntax

```
Function InSheet : Boolean;
```

Description

Example

See also

ISch_Component interface

IsIntegratedComponent method

(ISch_Component interface)

Syntax

```
Function IsIntegratedComponent : Boolean;
```

Description

Example

See also

ISch_Component interface

IsMultiPartComponent method

(ISch_Component interface)

Syntax

```
Function IsMultiPartComponent : Boolean;
```

Description

Example

See also

ISch_Component interface

RemoveSchImplementation method

(ISch_Component interface)

Syntax

```
Procedure RemoveSchImplementation(AnImplementation : ISch_Implementation);
```

Description

Example

See also

ISch_Component interface

SetState_AliasAsText method

(ISch_Component interface)

Syntax

```
Procedure SetState_AliasAsText (AValue : WideString);
```

Description

Example

See also

ISch_Component interface

SetState_AliasAt method

(ISch_Component interface)

Syntax

```
Procedure SetState_AliasAt (i : Integer; AValue : WideString);
```

Description

Example

See also

ISch_Component interface

SetState_ComponentDescription method

(ISch_Component interface)

Syntax

```
Procedure SetState_ComponentDescription (AValue : WideString);
```

Description

Example

See also

ISch_Component interface

SetState_ComponentKind method

(ISch_Component interface)

Syntax

```
Procedure SetState_ComponentKind (AValue : TComponentKind);
```

Description

The SetState_ComponentKind function sets the component of a TComponentKind value.

eComponentKind_Standard: These components possess standard electrical properties, are always synchronized and are the type most commonly used on a schematic sheet.

eComponentKind_Mechanical: These components do not have electrical properties and will appear in the BOM. They are synchronized if the same components exist on both the Schematic and PCB documents. An example is a heatsink.

eComponentKind_Graphical: These components are not used during synchronization or checked for electrical errors. These components are used, for example, when adding company logos to documents.

eComponentKind_NetTie_BOM: These components short two or more different nets and these components will appear in the BOM and are maintained during synchronization.

eComponentKind_NetTie_NoBOM: These components short two or more different nets and these components will NOT appear in the BOM and are maintained during synchronization.

eComponentKind_Standard_NoBOM: These components possess standard electrical properties, and are synchronized BUT are not included in any BOM file produced from the file.

Example

Component.SetState_ComponentKind(eComponentKind_Standard);

See also

ISch_Component interface

SetState_CurrentPartID method

(ISch_Component interface)

Syntax

```
Procedure SetState_CurrentPartID (AValue : Integer);
```

Description

Example

See also

ISch_Component interface

SetState_DesignatorLocked method

(ISch_Component interface)

Syntax

```
Procedure SetState_DesignatorLocked (AValue : Boolean);
```

Description

Example

See also

ISch_Component interface

SetState_DisplayFieldNames method

(ISch_Component interface)

Syntax

```
Procedure SetState_DisplayFieldNames (AValue : Boolean);
```

Description

Example

See also

ISch_Component interface

SetState_DisplayMode method

(ISch_Component interface)

Syntax

```
Procedure SetState_DisplayMode (AValue : TDisplayMode);
```

Description

Example

See also

ISch_Component interface

SetState_DisplayModeCount_Check method

(ISch_Component interface)

Syntax

```
Procedure SetState_DisplayModeCount_Check (AValue : Integer);
```

Description

Example

See also

ISch_Component interface

SetState_FilePosition method

(ISch_Component interface)

Syntax

```
Procedure SetState_FilePosition (AValue : Integer);
```

Description

Example

See also

ISch_Component interface

SetState_IsMirrored method

(ISch_Component interface)

Syntax

```
Procedure SetState_IsMirrored (AValue : Boolean);
```

Description

The SetState_IsMirrored function sets the component's mirror property along the x-axis.

Example

```
Comp.SetState_IsMirrored(True);
```

See also

ISch_Component interface

SetState_LibraryPath method

(ISch_Component interface)

Syntax

```
Procedure SetState_LibraryPath (AValue : WideString);
```

Description

Example

See also

ISch_Component interface

SetState_LibReference method

(ISch_Component interface)

Syntax

```
Procedure SetState_LibReference (AValue : WideString);
```

Description

Example

See also

ISch_Component interface

SetState_Orientation method

(ISch_Component interface)

Syntax

```
Procedure SetState_Orientation (AValue : TRotationBy90);
```

Description

The Orientation property determines the orientation of the component on the schematic sheet in increments of 0,90,180 and 270 degrees only. This method sets the orientation value of the component and is used in the Orientation property.

Example

```
Component.SetState_Orientation(eRotate180);
```

See also

TRotationBy90 type

ISch_Component interface

SetState_OverideColors method

(ISch_Component interface)

Syntax

```
Procedure SetState_OverideColors (AValue : Boolean);
```

Description

The SetState_OverrideColors procedure sets the local colors for the component. This component's fill, line and pin colors are overridden with the colors from the Fill, Lines and Pins color boxes respectively.

Example

```
Comp.SetState_OverrideColors(True);
```

See also

ISch_Component interface

SetState_PartCountNoPart0 method

(ISch_Component interface)

Syntax

```
Procedure SetState_PartCountNoPart0 (AValue : Integer);
```

Description

A component can consist of more than one part, for example a 74LS00 contains four parts. This property returns the number of parts for the component.

The function sets the number of parts for a component and is used in the PartCountNoPart0 property.

Note

Each component also includes a non-graphical part, Part Zero. Part Zero is used for pins that are to be included in all parts of a multi-part component, for example power pins.

Example

See also

ISch_Component interface

SetState_PartIdLocked method

(ISch_Component interface)

Syntax

```
Procedure SetState_PartIdLocked (AValue : Boolean);
```

Description

Example

See also

ISch_Component interface

SetState_PinColor method

(ISch_Component interface)

Syntax

```
Procedure SetState_PinColor (AValue : TColor);
```

Description

Example

See also

ISch_Component interface

SetState_PinsMoveable method

(ISch_Component interface)

Syntax

```
Procedure SetState_PinsMoveable (AValue : Boolean);
```

Description

Example

See also

ISch_Component interface

SetState_SheetPartFileName method

(ISch_Component interface)

Syntax

```
Procedure SetState_SheetPartFileName (AValue : WideString);
```

Description

Example

See also

ISch_Component interface

SetState_ShowHiddenFields method

(ISch_Component interface)

Syntax

```
Procedure SetState_ShowHiddenFields (AValue : Boolean);
```

Description

The SetState_ShowHiddenFields procedure determines the visibility of the text fields associated with the component, such as its name and filename. If the Value is true, the hidden fields of the comonent will be displayed on the schematic sheet. If the value is False, the hidden text fields are not shown on the schematic.

Example

```
Comp.SetState_ShowHiddenFields(True); // display the hidden text fields.
```

See also

ISch_Component interface

SetState_ShowHiddenPins method

(ISch_Component interface)

Syntax

```
Procedure SetState_ShowHiddenPins (AValue : Boolean);
```

Description

This property determines whether the hidden pins of a component can be hidden or not. Power pins are often defined as hidden. This method sets the boolean value whether the hidden pins are displayed or not and is used in the ShowHiddenPins property.

Example

```
Comp.SetState_ShowHiddenPins(True); // show hidden pins of this component.
```

See also

ISch_Component interface

SetState_SourceLibraryName method

(ISch_Component interface)

Syntax

```
Procedure SetState_SourceLibraryName (AValue : WideString);
```

Description

Example

See also

ISch_Component interface

SetState_TargetFileName method

(ISch_Component interface)

Syntax

```
Procedure SetState_TargetFileName (AValue : WideString);
```

Description

Example

See also

ISch_Component interface

SetState_UniqueId method

(ISch_Component interface)

Syntax

```
Procedure SetState_UniqueId (AValue : WideString);
```

Description

The SetState_UniqueID procedure sets the new ID for the component. All parameters, sheet symbols, ports, pins, components, openbus links, openbus ports and openbus components have Unique IDs. Unique IDs are used to maintain design synchronization in design projects.

The Unique ID (UID) is an system generated value that uniquely identifies this current component. It is used for linking to a PCB document and for project management. Enter a new UID value or click the **Reset** button to generate a new UID for this design object from the Change Properties dialog. You can also globally reset UIDs of components and sheet symbols from the Schematic Editor's **Tools » Convert » Reset Component Unique IDs** menu.

Example

```
UID := WSM.DM_GenerateUniqueID; // interface and method from Workspace Manager API.
Component.SetState_UniqueID(UID);
```

See also

ISch_Component interface

Properties

Alias property

(ISch_Component interface)

Syntax

```
Property Alias[i : Integer] : WideString Read GetState_AliasAt Write SetState_AliasAt;
```

Description

The indexed property returns an alias string. A component can have multiple aliases because a component name can be referred to by multiple names. For example a SN7432 is also SN74LS32 or SN74S32.

Notes

Use the AliasCount property to obtain the number of aliases before going through one by one.

Example

See also

ISch_Component interface

AliasAsText property

(ISch_Component interface)

Syntax

```
Property AliasAsText : WideString Read GetState_AliasAsText Write SetState_AliasAsText;
```

Description

Example

See also

ISch_Component interface

AliasCount property

(ISch_Component interface)

Syntax

```
Property AliasCount : Integer Read GetState_AliasCount;
```

Description

Notes

Use the AliasCount to obtain the count before going through each indexed Alias property one by one.

Example

See also

ISch_Component interface

Comment property

(ISch_Component interface)

Syntax

```
Property Comment : ISch_Parameter Read GetState_SchComment;
```

Description

The Comment property determines the comment object associated with the component object. The Component Properties dialog for this component has a Comment field. The Parameter object has a Name and Value fields and this Name field will normally have ‘Comment’ string and a Value string.

Example

Comp.Comment.Name := ‘LM833M’;

See also

ISch_Parameter interface;

ISch_Component interface

ComponentDescription property

(ISch_Component interface)

Syntax

```
Property ComponentDescription : WideString Read GetState_ComponentDescription Write
SetState_ComponentDescription;
```

Description

The ComponentDescription property determines the description string for this component. Normally this string contains text on what this component is. This property is supported by the GetState_ComponentDescription and SetState_ComponentDescription methods.

Example

```
Comp.ComponentDescription := ‘Fast Settling Dual Operational Amplifier’;
```

See also

ISch_Component interface

ComponentKind property

(ISch_Component interface)

Syntax

```
Property ComponentKind : TComponentKind Read GetState_ComponentKind Write
SetState_ComponentKind;
```

Description

The ComponentKind property deteremines the component’s type of TComponentKind type. This property is supported by the GetState_ComponentKind and Setstate_Component kind methods.

eComponentKind_Standard: These components possess standard electrical properties, are always synchronized and are the type most commonly used on a schematic sheet.

eComponentKind_Mechanical: These components do not have electrical properties and will appear in the BOM. They are synchronized if the same components exist on both the Schematic and PCB documents. An example is a heatsink.

eComponentKind_Graphical: These components are not used during synchronization or checked for electrical errors. These components are used, for example, when adding company logos to documents.

eComponentKind_NetTie_BOM: These components short two or more different nets and these components will appear in the BOM and are maintained during synchronization.

eComponentKind_NetTie_NoBOM: These components short two or more different nets and these components will NOT appear in the BOM and are maintained during synchronization.

eComponentKind_Standard_NoBOM: These components possess standard electrical properties, and are synchronized BUT are not included in any BOM file produced from the file.

Example

```
Component.ComponentKind := eComponentKind_NetTie_BOM;
```

See also

TComponentKind from RT_Workspace unit.

ISch_Component interface

CurrentPartID property

(ISch_Component interface)

Syntax

```
Property CurrentPartID : Integer Read GetState_CurrentPartID Write SetState_CurrentPartID;
```

Description

Example

See also

ISch_Component interface

Designator property

(ISch_Component interface)

Syntax

```
Property Designator : ISch_Designator Read GetState_SchDesignator;
```

Description

Example

See also

ISch_Designator interface.

ISch_Component interface

DisplayFieldNames property

(ISch_Component interface)

Syntax

```
Property DisplayFieldNames : Boolean Read GetState_DisplayFieldNames Write
SetState_DisplayFieldNames;
```

Description

Example

See also

ISch_Component interface

DesignatorLocked property

(ISch_Component interface)

Syntax

```
Property DesignatorLocked : Boolean Read GetState_DesignatorLocked Write
SetState_DesignatorLocked;
```

Description

Example

See also

ISch_Component interface

DisplayMode property

(ISch_Component interface)

Syntax

```
Property DisplayMode : TDisplayMode Read GetState_DisplayMode Write SetState_DisplayMode;
```

Description

Example

See also

ISch_Component interface

DisplayModeCount property

(ISch_Component interface)

Syntax

```
Property DisplayModeCount : Integer Read GetState_DisplayModeCount Write
SetState_DisplayModeCount_Check;
```

Description

The property can return up to 255 display modes for the same component. Modes are added or edited in the Schematic Library Editor.

This property is supported by the GetState_DisplayModeCount and SetState_DisplayModeCount_Check methods.

Example

See also

ISch_Component interface

IsMirrored property

(ISch_Component interface)

Syntax

```
Property IsMirrored : Boolean Read GetState_IsMirrored Write SetState_IsMirrored;
```

Description

The IsMirrored property determines whether the component is mirrored along the x-axis. This property is supported by the GetState_IsMirrored and SetState_IsMirrored methods.

Example

Component.IsMirrored := False;

See also

ISch_Component interface

LibraryPath property

(ISch_Component interface)

Syntax

```
Property LibraryPath : WideString Read GetState_LibraryPath Write SetState_LibraryPath;
```

Description

Example

See also

ISch_Component interface

LibReference property

(ISch_Component interface)

Syntax

```
Property LibReference : WideString Read GetState_LibReference Write SetState_LibReference;
```

Description

Example

See also

ISch_Component interface

Orientation property

(ISch_Component interface)

Syntax

```
Property Orientation : TRotationBy90 Read GetState_Orientation Write SetState_Orientation;
```

Description

This property determines the orientation of the component on the schematic sheet in increments of 0,90,180 and 270 degrees only. This property is supported by the GetState_Orientation and SetState_Orientation methods.

Example

```
Component.Orientation := eRotate180;
```

See also

ISch_Component interface

TRotationBy90 type

OverideColors property

(ISch_Component interface)

Syntax

```
Property OverideColors : Boolean Read GetState_OverideColors Write SetState_OverideColors;
```

Description

Example

See also

ISch_Component interface

PartCount property

(ISch_Component interface)

Syntax

```
Property PartCount : Integer Read GetState_PartCountNoPart0 Write SetState_PartCountNoPart0;
```

Description

A component can consist of more than one part, for example a 74LS00 contains four parts. This property returns the number of parts for the component and is supported by the GetState_PartCountNoPart0 and SetState_PartCountNoPart0 methods.

Note

Each component also includes a non-graphical part, Part Zero. Part Zero is used for pins that are to be included in all parts of a multi-part component, for example power pins.

Example

See also

ISch_Component interface

PinsMoveable property

(ISch_Component interface)

Syntax

```
Property PinsMoveable : Boolean Read GetState_PinsMoveable Write SetState_PinsMoveable;
```

Description

Example

See also

ISch_Component interface

PinColor property

(ISch_Component interface)

Syntax

```
Property PinColor : TColor Read GetState_PinColor Write SetState_PinColor;
```

Description

Example

See also

ISch_Component interface

PartIdLocked property

(ISch_Component interface)

Syntax

```
Property PartIdLocked : Boolean Read GetState_PartIdLocked Write SetState_PartIdLocked;
```

Description

Example

See also

ISch_Component interface

SheetPartFileName property

(ISch_Component interface)

Syntax

```
Property SheetPartFileName : WideString Read GetState_SheetPartFileName Write
SetState_SheetPartFileName;
```

Description

Example

See also

ISch_Component interface

ShowHiddenFields property

(ISch_Component interface)

Syntax

```
Property ShowHiddenFields : Boolean Read GetState_ShowHiddenFields Write
SetState_ShowHiddenFields;
```

Description

The ShowHiddenFields property determines the visibility of the text fields associated with the component, such as its name. If the Value is true, the hidden fields of the component will be displayed on the schematic sheet. If the value is False, the hidden text fields are not shown on the schematic.

Example

```
Comp.ShowHiddenFields := True;
```

See also

ISch_Component interface

ShowHiddenPins property

(ISch_Component interface)

Syntax

```
Property ShowHiddenPins : Boolean Read GetState_ShowHiddenPins Write SetState_ShowHiddenPins;
```

Description

This property determines whether the hidden pins of a component can be hidden or not. Power pins are often defined as hidden. This property is supported by the GetState_ShowHiddenPins and SetState_ShowHiddenPins methods.

Example

```
Comp.ShowHiddenPins := True;
```

See also

ISch_Component interface

SourceLibraryName property

(ISch_Component interface)

Syntax

```
Property SourceLibraryName : WideString Read GetState_SourceLibraryName Write
SetState_SourceLibraryName;
```

Description

Example

See also

ISch_Component interface

TargetFileName property

(ISch_Component interface)

Syntax

```
Property TargetFileName : WideString Read GetState_TargetFileName Write
SetState_TargetFileName;
```

Description

Example

See also

ISch_Component interface

UniqueId property

(ISch_Component interface)

Syntax

```
Property UniqueId : WideString Read GetState_UniqueId Write SetState_UniqueId;
```

Description

The UniqueID property sets the new ID for the component. All parameters, sheet symbols, ports, pins, components, openbus links, openbus ports and openbus components have Unique IDs. Unique IDs are used to maintain design synchronization in design projects.

The Unique ID (UID) is an system generated value that uniquely identifies this current component. It is used for linking to a PCB document and for project management. Enter a new UID value or click the **Reset** button to generate a new UID for this design object from the Change Properties dialog. You can also globally reset UIDs of components and sheet symbols from the Schematic Editor's **Tools » Convert » Reset Component Unique IDs** menu.

Example

```
UID := WSM.DM_GenerateUniqueID; // interface and method from Workspace Manager API.
Component.UniqueID(UID);
```

See also

ISch_Component interface

ISch_ConnectionLine Interface

Overview

A connection line represents a line that has corner properties as well as width and style properties between two nodes on a schematic document.

Notes

The ISch_ConnectionLine interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Line

ISch_BusEntry

ISch_ConnectionLine

ISch_ConnectionLine methods

GetState_IsInferred

SetState_IsInferred

ISch_ConnectionLine properties

IsInferred

See also

Methods

UpdatePrimitivesAccessibility method

(ISch_Component interface)

Syntax

```
Procedure UpdatePrimitivesAccessibility;
```

Description

When the connection lines have been modified, invoke the UpdatePrimitivesAccessibility to ensure the primitives associated with the connection lines have been refreshed.

Example

See also

ISch_Component interface

GetState_IsInferred method

(ISch_ConnectionLine interface)

Syntax

```
Function GetState_IsInferred : Boolean;
```

Description

An inferred property indicates that a connection between documents has been detected by the Schematic Navigation system after the project has been compiled.

An inferred property denotes whether the object is an inferred object with respect to connective objects. Bus and Sheet Symbols can be defined in ranges using the NetLabel [] and Repeat statements respectively and once the project has been compiled, inferred objects created in memory for navigation/connective purposes. For example, a Bus with a range of A[0..4] ends up with five wires with A0...A5 net labels (only in memory). This property is useful for multi – channel projects and for sheets that have Bus objects.

This method gets the IsInferred state and is used in the IsInferred property.

Example

See also

ISch_ConnectionLine interface

SetState_IsInferred method

(ISch_ConnectionLine interface)

Syntax

```
Procedure SetState_IsInferred(B : Boolean);
```

Description

An inferred property indicates that a connection between documents has been detected by the Schematic Navigation system after the project has been compiled.

An inferred property denotes whether the object is an inferred object with respect to connective objects. Bus and Sheet Symbols can be defined in ranges using the NetLabel [] and Repeat statements respectively and once the project has been compiled, inferred objects created in memory for navigation/connective purposes. For example, a Bus with a range of A[0..4] ends up with five wires with A0...A5 net labels (only in memory). This property is useful for multi – channel projects and for sheets that have Bus objects.

This method sets the IsInferred state and is used in the IsInferred property.

Example

See also

ISch_ConnectionLine interface

Properties

IsInferred property

(ISch_ConnectionLine interface)

Syntax

```
Property IsInferred : Boolean Read GetState_IsInferred Write SetState_IsInferred;
```

Description

An inferred property indicates that a connection between documents has been detected by the Schematic Navigation system after the project has been compiled.

An inferred property denotes whether the object is an inferred object with respect to connective objects. Bus and Sheet Symbols can be defined in ranges using the NetLabel [] and Repeat statements respectively and once the project has been compiled, inferred objects created in memory for navigation/connective purposes. For example, a Bus with a range of A[0..4] ends up with five wires with A0...A5 net labels (only in memory). This property is useful for multi – channel projects and for sheets that have Bus objects.

This property is supported by the GetState_IsInferred and SetState_IsInferred methods.

Example

See also

ISch_ConnectionLine interface

ISch_CrossSheetConnector Interface

Overview

Cross sheet connector objects can be used to link a net from a sheet to other sheets within a project. This method defines global connections between sheets within a project.

Notes

The ISch_CrossSheetConnector interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Label

ISch_PowerObject

ISch_CrossSheetConnector

ISch_CrossSheetConnector methods

GetCrossSheetConnectorStyle

SetCrossSheetConnectorStyle

ISch_CrossSheetConnector properties

CrossSheetStyle

See also

ISch_GraphicalObject interface

ISch_Label interface

ISch_PowerObject interface

ISch_CrossSheetConnector interface

Methods

GetCrossSheetConnectorStyle method

(ISch_CrossSheetConnector interface)

Syntax

```
Function GetCrossSheetConnectorStyle : TCrossSheetConnectorStyle;
```

Description

The `GetCrossSheetConnectorStyle` function determines the style or the alignment of the Off Sheet Connector object.

Example

```
// Port alignment is determined by the CrossConnector's Style.
If CrossConn.GetCrossSheetStyle = eCrossSheetRight Then
    Port.Alignment := eRightAlign
Else
    Port.Alignment := eLeftAlign;
```

See also

TCrossSheetConnectorStyle type

ISch_CrossSheetConnector interface

SetCrossSheetConnectorStyle method

(ISch_CrossSheetConnector interface)

Syntax

```
Procedure SetCrossSheetConnectorStyle (Const Value : TCrossSheetConnectorStyle);
```

Description

The SetCrossSheetConnectorStyle function sets the style or the alignment of the off sheet connector object.

Example

```
// Port alignment is determined by the CrossConnector's Style.
If Port.Alignment := eRightAlign Then
    CrossConn.CrossSheetStyle := eCrossSheetRight
Else
```

```
    CrossConn.CrossSheetStyle := eCrossSheetLeft
```

See also

TCrossSheetConnectorStyle type

ISch_CrossSheetConnector interface

Properties

CrossSheetStyle property

(ISch_CrossSheetConnector interface)

Syntax

```
Property CrossSheetStyle : TCrossSheetConnectorStyle Read GetCrossSheetConnectorStyle Write
SetCrossSheetConnectorStyle;
```

Description

The CrossSheetStyle property represents the style or the alignment of the cross sheet object. This property is supported by the GetCrossSheetConnectorStyle and SetCrossSheetConnectorStyle methods.

Example

```
// Port alignment is determined by the CrossConnector's Style.
If CrossConn.CrossSheetStyle = eCrossSheetRight Then
    Port.Alignment := eRightAlign
Else
    Port.Alignment := eLeftAlign;
```

See also

TCrossSheetConnectorStyle type

ISch_CrossSheetConnector interface

ISch_Designator Interface

Overview

The ISch_Designator interface represents a designator object which is part of the component object that identifies it as part of a net. Refer to the `ISch_Parameter` interface for details.

Notes

The ISch_Designator interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Label

ISch_ComplexText

ISch_Parameter

ISch_Designator

ISch_Designator methods

ISch_Designator properties

See also

ISch_GraphicalObject interface

ISch_Label interface

ISch_ComplexText interface

ISch_Parameter interface

ISch_Designator interface

ISch_Directive Interface

Overview

An ISch_Directive interface represents an object that stores a text string. It is an ancestor interface for the ISch_ErrorMarker interface. Design constraints (rules) can be defined prior to PCB layout, by adding parameters that are configured as design rule directives to the schematic source document(s).

Notes

The ISch_Directive interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Directive

ISch_Directive methods

ISch_Directive properties

Text

See also

ISchGraphicalObject interface

Properties

Text property

(ISch_Directive interface)

Syntax

```
Property Text : WideString Read GetState_Text Write SetState_Text;
```

Description

The Text property represents the text information for the directive objects and the error marker objects.

Example

```
Directive.Text := 'Schematic Directive';
```

See also

ISch_Directive interface

ISch_ErrorMarker interface

ISch_Ellipse

Overview

An ellipse is a drawing object which is filled or unfilled graphic elements on a schematic sheet. Refer to the `ISch_Circle` interface for details.

Notes

The ISch_Ellipse interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Circle

ISch_Ellipse

ISch_Ellipse methods

GetState_SecondaryRadius

SetState_SecondaryRadius

ISch_Ellipse properties

SecondaryRadius

Methods

GetState_SecondaryRadius method

(ISch_Ellipse interface)

Syntax

```
Function GetState_SecondaryRadius : TDistance;
```

Description

This function retrieves the secondary radius or the Y coordinate of the elliptical arc with a TDistance value.

Example

```
XRadius := Ellipse.Radius;
YRadius := Ellipse.SecondaryRadius;
```

See also

TDistance type

ISch_Circle interface

SetState_SecondaryRadius method

(ISch_Ellipse interface)

Syntax

```
Procedure SetState_SecondaryRadius(ARadius : TDistance);
```

Description

This function sets the secondary radius or the Y coordinate of the ellipse with a TDistance value.

Example

```
Ellipse.Radius          := 4000000
Ellipse.SecondaryRadius := 7000000;
```

See also

ISch_EllipticalArc interface

Properties

SecondaryRadius property

(ISch_Ellipse interface)

Syntax

```
Property SecondaryRadius : TDistance Read GetState_SecondaryRadius Write
SetState_SecondaryRadius;
```

Description

The secondary radius property defines the second set of arcs the define the elliptical arc. The elliptical arc has two sets of arcs (four all together). The Radius property defines the first set of arcs that define the elliptical arc (inherited from the ISch_Arc interface). This property is supported by the GetState_SecondaryRadius and SetState_SecondaryRadius methods.

Example

```
XRadius := Ellipse.Radius;
YRadius := Ellipse.SecondaryRadius;
```

See also

TDistance type

ISch_Circle interface

ISch_EllipticalArc Interface

Overview

Elliptical arc objects are drawing objects which represent open circular or elliptical curves on a schematic sheet. Refer to the ISch_Arc interface for extra details.

Notes

The ISch_EllipticalArc interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Arc

ISch_EllipticalArc

ISch_EllipticalArc methods

GetState_SecondaryRadius

ISch_EllipticalArc properties

SecondaryRadius

SetState_SecondaryRadius

See also

ISch_GraphicalObject interface

ISch_Arc interface

Methods

GetState_SecondaryRadius method

(ISch_EllipticalArc interface)

Syntax

```
Function GetState_SecondaryRadius : TDistance;
```

Description

This function retrieves the secondary radius or the Y coordinate of the elliptical arc with a TDistance value.

Example

```
XRadius := EllipticalArc.Radius;
YRadius := EllipticalArc.SecondaryRadius;
```

See also

TDistance type

ISch_EllipticalArc interface

SetState_SecondaryRadius method

(ISch_EllipticalArc interface)

Syntax

```
Procedure SetState_SecondaryRadius(ARadius : TDistance);
```

Description

This function sets the secondary radius or the Y coordinate of the elliptical arc with a TDistance value.

Example

```
EllipticalArc.Radius          := 4000000
EllipticalArc.SecondaryRadius := 7000000;
```

See also

TDistance type

ISch_EllipticalArc interface

Properties

SecondaryRadius property

(ISch_EllipticalArc interface)

Syntax

```
Property SecondaryRadius : TDistance Read GetState_SecondaryRadius Write
SetState_SecondaryRadius;
```

Description

The secondary radius property defines the second set of arcs the define the elliptical arc. The elliptical arc has two sets of arcs (four all together). The Radius property defines the first set of arcs that define the elliptical arc (inherited from the ISch_Arc interface). This property is supported by the GetState_SecondaryRadius and SetState_SecondaryRadius methods.

Example

```
XRadius := EllipticalArc.Radius;
YRadius := EllipticalArc.SecondaryRadius;
```

See also

TDistance type

ISch_Arc interface

ISch_EllipticalArc interface

ISch_ErrorMarker Interface

Overview

Error Markers are placed on a schematic sheet at the site of each ERC violation by the Schematic Editor. Refer to the ISch_Directive and ISch_GraphicalObject interfaces for details.

Notes

The ISch_ErrorMarker interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Directive

ISch_ErrorMarker

See also

ISch_GraphicalObject interface

ISch_Directive interface

ISch_HarnessConnector Interface

Overview

The ISch_HarnessConnector interface is used to represent a harness connector design obejct which is a member of the harness system.

Notes

The ISch_HarnessEntry interface hierarchy is as follows;

ISch_GraphicalObject

ISch_RectangularGroup

ISch_HarnessConnector

ISch_HarnessConnector Methods

SetState_LineWidth

GetState_LineWidth

GetState_SchHarnessConnectorType

GetState_MasterEntryLocation

ISch_HarnessConnector Properties

LineWidth

HarnessConnectorType

MasterEntryLocation

Methods

SetState_LineWidth method

(ISch_HarnessConnector interface)

Syntax

```
Procedure SetState_LineWidth(Value : TSize);
```

Description

The SetState_LineWidth sets the line width of the harness connector which is based on one of the the TSize values.

Example

```
HarnessConn.SetState_LineWidth(eLarge);
```

See also

TSize type

ISch_HarnessConnector interface

ISch_HarnessEntry interface

GetState_LineWidth method

(ISch_HarnessConnector interface)

Syntax

```
Function GetState_LineWidth : TSize;
```

Description

The GetState_LineWidth gets the line width of the harness connector which is based on one of the the TSize values.

Example

```
LineWidth := HarnessConn.GetState_LineWidth;
```

See also

TSize type

ISch_HarnessConnector interface

ISch_HarnessEntry interface

GetState_SchHarnessConnectorType method

(ISch_HarnessConnector interface)

Syntax

```
Function GetState_SchHarnessConnectorType : ISch_HarnessConnectorType;
```

Description

The GetState_SchHarnessConnectorType function retrieves the harness connector type of the harness connector. The default type is ‘Harness’. This type value can be modified.

Example

```
Var
    HarnessConn : ISch_HarnessConnector;
    ConnType    : ISch_HarnessConnectorType;
    S           : String;
Begin
    // HarnessConn is a ISch_harnessConnector interface representing
    // a harness connector design object.
    ConnType := HarnessConn. GetState_SchHarnessConnectorType;

    // Display the Text string for this harness connector.
    S := ConnType.Text;
```

See also

ISch_HarnessConnectorType interface

ISch_HarnessConnector interface

ISch_HarnessEntry interface

GetState_MasterEntryLocation method

(ISch_HarnessConnector interface)

Syntax

```
Function  GetState_MasterEntryLocation : TLocation;
```

Description

The GetState_MasterEntryLocation function returns the location of the master entry of the harness connector. The master entry represents the tip of the harness connector and the position of the tip is determined from the top side of the connector.

Example

```
Location := HarnessConn.GetState_MasterEntryLocation;
```

See also

TLocation type

ISch_HarnessConnectorType interface

ISch_HarnessConnector interface

ISch_HarnessEntry interface

Properties

LineWidth property

(ISch_HarnessConnector interface)

Syntax

```
Property LineWidth : TSize Read GetState_LineWidth Write SetState_LineWidth;
```

Description

The LineWidth property defines the line width of the harness connector which is based on one of the TSize values. . This property is supported by the GetState_LineWidth and SetState_LineWidth methods.

Example

```
HarnessConn.LineWidth := eLarge;
```

See also

TSize type

ISch_HarnessConnector interface

HarnessConnectorType property

(ISch_HarnessConnector interface)

Syntax

```
Property HarnessConnectorType: ISch_HarnessConnectorType Read
GetState_SchHarnessConnectorType;
```

Description

The HarnessConnectorType property defines the harness connector type of the harness connector and returns the ISch_HarnessConnectorType interface. The default connector type is ‘Harness’. This property is supported by the GetState_HarnessConnectorType method.

Example

```
Var
    HarnessConn : ISch_HarnessConnector;
    ConnType    : ISch_HarnessConnectorType;
    S           : String;
Begin
    // HarnessConn is a ISch_HarnessConnector interface representing
    // a harness connector design object.
    ConnType := HarnessConn.HarnessConnectorType;

    // Display the Text string for this harness connector.
    S := ConnType.Text;
```

See also

TSize type

ISch_HarnessConnectorType interface

ISch_HarnessConnector interface

MasterEntryLocation property

(ISch_HarnessConnector interface)

Syntax

```
Property MasterEntryLocation : TLocation Read GetState_MasterEntryLocation;
```

Description

The MasterEntryLocation property defines the location of the master entry of the harness connector. The master entry represents the tip of the harness connector and the position of the tip is determined from the top side of the connector.. This property is supported by the GetState_LineWidth method.

Example

See also

TSize type

ISch_HarnessConnector interface

ISch_HarnessConnectorType Interface

Overview

The ISchHarnessConnectorType interface represents the text object of the harness connector and defines the harness connector type. By Default the Type string is Harness.

Notes

The ISch_HarnessConnectorType interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Label

ISch_ComplexT0065t

ISch_HarnessConnectorType

ISch_HarnessConnector Methods **ISch_HarnessConnector Properties**

See also

ISch_HarnessConnector interface

ISch_HarnessEntry interface.

ISch_HarnessEntry Interface

Overview

The ISch_HarnessEntry interface is used to represent a harness entry which is a member of the harness system. Harness Entries are the graphical definition of a Signal Harness member. They are placed within a Harness Connector and they are the connection point through which actual nets, buses and Signal Harnesses are combined to form a higher level Signal Harness. Harness Entries along with Harness Connectors, Signal Harnesses and Harness Definition Files make up a complete Signal Harness.

Notes

The ISch_HarnessEntry interface hierarchy is as follows;

ISch_GraphicalObject

ISch_HarnessEntry

ISch_HarnessEntry methods

SetState_Name

SetState_Side

SetState_DistanceFromTop

SetState_TextColor

SetState_OverrideDisplayString

GetState_Name

GetState_Side

GetState_DistanceFromTop

GetState_TextColor

ISch_HarnessEntry properties

IsVertical

Name

Side

DistanceFromTop

TextColor

OverrideDisplayString

OwnerHarnessConnector

GetState_OverrideDisplayString

GetState_SchOwnerHarnessConnector

Methods

GetState_Name method

(ISch_HarnessEntry interface)

Syntax

```
Function GetState_Name : WideString;
```

Description

The GetState_Name function returns the name of the harness entry. Normally the name is a number but can be alphanumeric.

Example

```
EntryName := HarnessEntry.GetStateName
```

See also

Name property.

ISch_HarnessEntry interface

GetState_Side method

(ISch_HarnessEntry interface)

Syntax

```
Function GetState_Side : TLeftRightSide;
```

Description

The GetState_Side function returns the orientation of the harness entry in respect to the associated harness connector as a `TLeftRightSide` type.

Example

```
Side := HarnessEntry.GetState_Side;
```

See also

TLeftRightSide type

ISch_HarnessEntry interface

GetState_DistanceFromTop method

(ISch_HarnessEntry interface)

Syntax

```
Function  GetState_DistanceFromTop : TCoord;
```

Description

The GetState_DistanceFromTop function returns the distance from this harness entry to the top edge of the harness connector in a value that's dependent on the grid units. For example if the grid was in DXP Defaults (10 DXP units = 100 mils for example) and the Entry is 10 Units away from the Top part of the Harness Connector.

Example

```
Distance := HarnessEntry.GetState_DistanceFromTop;
```

See also

ISch_HarnessEntry interface

GetState_TextColor method

(ISch_HarnessEntry interface)

Syntax

```
Function  GetState_TextColor : TColor;
```

Description

The GetState_TextColor function returns the color of the text used for the Name of the Harness Entry.

Example

```
Color := HarnessEntry.GetState_TextColor;
```

See also

TColor type

ISch_HarnessEntry

GetState_OverrideDisplayString method

(ISch_HarnessEntry interface)

Syntax

```
Function  GetState_OverrideDisplayString : WideString;
```

Description

The GetState_OverrrideDisplayString function returns the override display string which overrides the Name string.

Example

```
DisplayString := HarnessEntry.GetState_OverrideDisplayString;
```

See also

ISch_HarnessEntry interface

GetState_SchOwnerHarnessConnector method

(ISch_HarnessEntry interface)

Syntax

```
Function  GetState_SchOwnerHarnessConnector : ISch_HarnessConnector;
```

Description

The GetState_SchOwnerHarnessConnector function returns the harness connector (ISch_HarnessConnector) that this harness entry is associated with.

Example

```
OwnerHarnessConnector := HarnessEntry.GetState_SchOwnerHarnessConnector;
```

See also

ISch_HarnessEntry interface

SetState_Name method

(ISch_HarnessEntry interface)

Syntax

```
Procedure SetState_Name(Value : WideString);
```

Description

The SetState_Name procedure sets the new name for the Harness Entry.

Example

```
HarnessEntry.SetState_Name('HarnessType2');
```

See also

ISch_HarnessEntry interface

SetState_Side method

(ISch_HarnessEntry interface)

Syntax

```
Procedure SetState_Side(Value : TLeftRightSide);
```

Description

The SetState_Side procedure sets the orientation of the harness entry in respect to the associated harness connector.

Example

```
HarnessEntry.SetState_Side(eLeftSide);
```

See also

TLeftRightSide type.

ISch_HarnessEntry interface.

SetState_DistanceFromTop method

(ISch_HarnessEntry interface)

Syntax

```
Procedure SetState_DistanceFromTop(Value : TCoord);
```

Description

The SetState_DistanceFromTop function sets the distance from this harness entry to the top edge of the harness connector in a value that's dependent on the grid units. For example if the grid was in DXP Defaults (10 DXP units = 100 mils for example) and the Entry is 10 Units away from the Top part of the Harness Connector then you would use the DxpToCoords function to translate the 10 grid units into a coordinate value.

Example

```
HarnessEntry.SetState_DistanceFromTop(DxpsToCoord(10));
```

See also

DXPsToCoord function

Measurement Conversion functions

ISch_HarnessEntry interface

SetState_TextColor method

(ISch_HarnessEntry interface)

Syntax

```
Procedure SetState_TextColor(Value : TColor);
```

Description

The SetState_TextColor procedure sets the color (a value of TColor type) for the Harness Entry's Name string.

Notes

The TColor value specifies a 6 digit hexadecimal number of the $FFFFFF format. For example the color blue would be RGB:0,0,255 and Hex:FF0000 therefore the converted decimal value would be 16711680. The following formula may be used to calculate the required value, R+256*(G+(256*B)).

Example

```
HarnessEntry.SetState_TextColor(0); // sets the text color to black.
```

See also

TColor type

ISch_HarnessEntry interface

SetState_OverrideDisplayString method

(ISch_HarnessEntry interface)

Syntax

```
Procedure SetState_OverrideDisplayString(Value : WideString );
```

Description

The SetState_OverrideDisplayString procedure sets a new value consisting of alph-numeric characters for the Override Display string.

Example

```
HarnessEntry.SetState_OverrideDisplayString('New Override String');
```

See also

ISch_HarnessEntry interface

Properties

IsVertical

(ISch_HarnessEntry interface)

Syntax

```
Function  IsVertical : Boolean;
```

Description

The IsVertical property defines the orientation of the harness entry in respect to the harness connector.

Example

```
If HarnessEntry.IsVertical Then ShowMessage('The hentry is vertical.');
```

See also

ISch_HarnessEntry interface

Name

(ISch_HarnessEntry interface)

Syntax

```
Property Name : WideString Read GetState_Name Write SetState_Name;
```

Description

The Name property defines the name of the harness entry. Normally the name property is a number but can be alphanumeric... This property is supported by the GetState_Name and SetState_Name methods.

Example

```
HarnessEntry.Name := 'HarnessType_2';
```

See also

ISch_HarnessEntry interface

Side

(ISch_HarnessEntry interface)

Syntax

```
Property Side : TLeftRightSide Read GetState_Side Write SetState_Side;
```

Description

The Side property defines the orientation of the harness entry in respect to the associated harness connector. This property is supported by the GetState_Side and SetState_Side methods.

Example

```
HarnessEntry.Side := eLeftSide;
```

See also

ISch_HarnessEntry interface

DistanceFromTop

(ISch_HarnessEntry interface)

Syntax

```
Property DistanceFromTop : TCoord Read GetState_DistanceFromTop Write
SetState_DistanceFromTop;
```

Description

The DistanceFromTop property defines the location of the harness entry in respect to the associated harness connector. This property is supported by the GetState_DistanceFromTop and SetState_DistanceFromTop methods.

Example

```
HarnessEntry.DistanceFromTop := DxpsToCoord(10);
```

See also

ISch_HarnessEntry interface

TextColor

(ISch_HarnessEntry interface)

Syntax

```
Property TextColor : TColor Read GetState_TextColor Write SetState_TextColor;
```

Description

The TextColor property defines the color (a value of TColor type) for the Harness Entry's Name string. This property is supported by the GetState_TextColor and SetState_TextColor methods.

Notes

The TColor value specifies a 6 digit hexadecimal number of the $FFFFFF format. For example the color blue would be RGB:0,0,255 and Hex:FF0000 therefore the converted decimal value would be 16711680. The following formula may be used to calculate the required value, R+256*(G+(256*B)).

Example

```
HarnessEntry.TextColor := 0; // sets the name color to black.
```

See also

TColor type

ISch_HarnessEntry interface

OverrideDisplayString

(ISch_HarnessEntry interface)

Syntax

```
Property OverrideDisplayString : WideString Read GetState_OverrideDisplayString Write
SetState_OverrideDisplayString;
```

Description

The OverrideDisplayString property defines the OverRideDisplayString property. This property is supported by the GetState_OverrirdeDisplayString and SetState_OverrirdeDisplayString methods.

Example

```
HarnessEntry.OverrideDisplayString('Display String overridden.');
```

See also

ISch_HarnessEntry interface

OwnerHarnessConnector

(ISch_HarnessEntry interface)

Syntax

```
Property OwnerHarnessConnector : ISch_HarnessConnector Read GetState_SchOwnerHarnessConnector;
```

Description

The OwnerHarnessConnector property retrieves the HarnessConnector interface this harness entry is associated with. This property is supported by the GetState_OwnerHarnessConnector method.

Example

```
HarnessConnector := HarnessEntry.GetState_OwnerHarnessConnector;
```

See also

ISch_HarnessEntry interface

IHarnessTypeHolder Interface

Overview

The IHarnessTypeHolder

IHarnessTypeHolder methods

SetState_HarnessType

SetState_HarnessTypeInferred

SetState_IsHarnessObject

GetState_HarnessType

GetState_HarnessTypeInferred

GetState_IsHarnessObject

IHarnessTypeHolder properties

HarnessType

HarnessTypeInferred

IsHarnessObject

Methods

SetState_HarnessType
SetState_HarnessTypeInferred
SetState_IsHarnessObject
GetState_HarnessType
GetState_HarnessTypeInferred
GetState_IsHarnessObject

Properties

HarnessType
HarnessTypeInferred
IsHarnessObject

ISch_Image Interface

Overview

The ISch_Image interfaces are used to represent graphical images on a schematic document.

Notes

The ISch_Image interface hierarchy is as follows;

ISch_GraphicalObject
    ISch_Rectangle
        ISch_Image

ISch_Image methods

SetState_FileName
SetState_EmbedImage
SetState_KeepAspect
GetState_FileName
GetState_EmbedImage
GetState_KeepAspect

ISch_Image properties

EmbedImage
FileName
KeepAspect

See also

ISch_GraphicalObject interface
ISch_Rectangle interface

Methods

SetState_FileName method

(ISch_Image interface)

Syntax

```
Procedure SetState_FileName (Const Value : WideString);
```

Description

Example

See also

ISch_Image interface

SetState_EmbedImage method

(ISch_Image interface)

Syntax

```
Procedure SetState_EmbedImage (Const Value : Boolean);
```

Description

Example

See also

ISch_Image interface

GetState_KeepAspect method

(ISch_Image interface)

Syntax

```
Function GetState_KeepAspect : Boolean;
```

Description

Example

See also

ISch_Image interface

GetState_FileName method

(ISch_Image interface)

Syntax

```
Function GetState_FileName : WideString;
```

Description

Example

See also

ISch_Image interface

GetState_EmbedImage method

(ISch_Image interface)

Syntax

```
Function GetState_EmbedImage : Boolean;
```

Description

Example

See also

ISch_Image interface

SetState_KeepAspect method

(ISch_Image interface)

Syntax

```
Procedure SetState_KeepAspect (Const Value : Boolean);
```

Description

Example

See also

ISch_Image interface

Properties

KeepAspect property

(ISch_Image interface)

Syntax

```
Property KeepAspect : Boolean Read GetState_KeepAspect Write SetState_KeepAspect;
```

Description

Example

See also

ISch_Image interface

FileName property

(ISch_Image interface)

Syntax

```
Property FileName : WideString Read GetState_FileName Write SetState_FileName;
```

Description

Example

See also

ISch_Image interface

EmbedImage property

(ISch_Image interface)

Syntax

```
Property EmbedImage : Boolean Read GetState_EmbedImage Write SetState_EmbedImage;
```

Description

Example

See also

ISch_Image interface

ISch_Junction Interface

Overview

Junctions are small circular objects used to logically join intersecting wires on the schematic sheet. The `ISch_Junction` interfaces represent manually placed junctions NOT system generated junctions. You will use the `IConnection` interfaces to work with system generated junctions.

Notes

The `ISch_Junction` interface hierarchy is as follows;

```
ISch_GraphicalObject
       ISch_Junction
```

ISch_Junction Methods and Properties Table

ISch_Junction methods	ISch_Junction properties
SetState_Size	Size
SetState_Locked	Locked
GetState_Size	
GetState_Locked	

See also

ISch_GraphicalObject interface

ISch_Junction Methods

SetState_Size method

(ISch_Junction interface)

Syntax

```
Procedure SetState_Size (ASize : TSize);
```

Description

This procedure sets the size of the manual junction. The size is one of four values; Smallest, Small, Medium and Large. This method is also used by the Size property.

Example

ManualJunction.SetState_Size(eMedium);

See also

ISch_Junction interface

TSize type

SetState_Locked method

(ISch_Junction interface)

Syntax

```
Procedure SetState_Locked(ALocked : Boolean);
```

Description

This procedure sets the Locked state of the manual junction. This method is also used by the Locked property.

Example

ManualJunction.SetState_Locked(True);

See also

ISch_Junction interface

GetState_Size method

(ISch_Junction interface)

Syntax

```
Function GetState_Size : TSize;
```

Description

This function gets the size of the manual junction. The size is one of four values; Smallest, Small, Medium and Large. This method is also used by the Size property.

Example

Size := ManualJunction.GetState_Size;

See also

ISch_Junction interface

TSize type

GetState_Locked method

(ISch_Junction interface)

Syntax

```
Function GetState_Locked : Boolean;
```

Description

This function gets the Locked state of the manual junction. This method is also used by the Locked property.

Example

Locked := ManualJunction.GetState_Locked;

See also

ISch_Junction interface

Properties

Size property

(ISch_Junction interface)

Syntax

```
Property Size : TSize Read GetState_Size Write SetState_Size;
```

Description

This property represents the size of the manual junction. The GetState_Size and SetState_Size methods are used by this property.

Example

```
Junction.Size := eSmallest;
```

See also

ISch_Junction interface

TSize type.

Locked property

(ISch_Junction interface)

Syntax

```
Property Locked : Boolean Read GetState_Locked Write SetState_Locked;
```

Description

This property represents the Locked property of the manual junction. The GetState_Locked and SetState_Locked methods are used by this property.

Example

```
Junction.Locked := True;
```

See also

ISch_Junction interface

ISch_Label Interface

Overview

The ISch_Label interface represents an existing label object on a schematic document. This interface is the ancestor interface for the ISch_NetLabel interfaces.

Notes

The interface hierarchy for the ISch_Label interface is as follows;

ISch_GraphicalObject

ISch_Label

ISch_Label methods **ISch_Label properties**

SetState_FontId	FontId
SetState_Orientation	Orientation
SetState_Justification	Justification
SetState_OverrideDisplayString	Text
SetState_IsMirrored	OverrideDisplayString
GetState_FontId	DisplayString
GetState_Orientation	Formula
GetState_Justification	CalculatedValueString
GetState_DisplayString	IsMirrored
GetState_Formula	
GetState_CalculatedValueString	
GetState_OverrideDisplayString	
GetState_IsMirrored	

See also

ISch_GraphicalObject interface

Methods

SetState_OverrideDisplayString method

(ISch_Label interface)

Syntax

```
Procedure SetState_OverrideDisplayString(S : WideString );
```

Description

Example

See also

ISch_Label interface

SetState_Orientation method

(ISch_Label interface)

Syntax

```
Procedure SetState_Orientation (ARotation : TRotationBy90);
```

Description

This Orientation property determines the angle the ISch_Label is at on the Schematic document. The angle is in 90 degree increments - 0, 90, 180, 270. This property is supported by the GetState_Orientation and SetState_Orientation methods.

Example

```
SchLabel.Orientation := eRotate90;
```

Example

See also

ISch_Label interface

SetState_Justification method

(ISch_Label interface)

Syntax

```
Procedure SetState_Justification (AValue : TTextJustification);
```

Description

The Justification property determines the alignment of the text in respect to the Label object whether it is left justified, centered and so on. This property is supported by the GetState_Justification and SetState_Justification methods.

Example

See also

ISch_Label interface

SetState_IsMirrored method

(ISch_Label interface)

Syntax

```
Procedure SetState_IsMirrored (AValue : Boolean);
```

Description

Example

See also

ISch_Label interface

SetState_FontId method

(ISch_Label interface)

Syntax

```
Procedure SetState_FontId (AFontId : TFontID);
```

Description

Example

See also

ISch_Label interface

GetState_OverrideDisplayString method

(ISch_Label interface)

Syntax

```
Function GetState_OverrideDisplayString : WideString;
```

Description

The GetState_OverrrideDisplayString function returns the override display string which overrides the Name string.

Example

```
DisplayString := Label.GetState_OverrideDisplayString;
```

See also

ISch_Label interface

GetState_Orientation method

(ISch_Label interface)

Syntax

```
Function GetState_Orientation : TRotationBy90;
```

Description

This Orientation property determines the angle the ISch_Label is at on the Schematic document. The angle is in 90 degree increments - 0, 90, 180, 270. This property is supported by the GetState_Orientation and SetState_Orientation methods.

Example

```
SchLabel.Orientation := eRotate90;
```

See also

ISch_Label interface

GetState_Justification method

(ISch_Label interface)

Syntax

```
Function GetState_Justification : TTextJustification;
```

Description

The Justification property determines the alignment of the text in respect to the Label object whether it is left justified, centered and so on. This property is supported by the GetState_Justification and SetState_Justification methods.

Example

```
Justification := Label.GetState_Justification;
```

See also

ISch_Label interface

GetState_IsMirrored method

(ISch_Label interface)

Syntax

```
Function GetState_IsMirrored : Boolean;
```

Description

Example

See also

ISch_Label interface

GetState_Formula method

(ISch_Label interface)

Syntax

```
Function GetState_Formula : WideString;
```

Description

Example

See also

ISch_Label interface

GetState_FontId method

(ISch_Label interface)

Syntax

```
Function GetState_FontId : TFontID;
```

Description

Example

See also

ISch_Label interface

GetState_DisplayString method

(ISch_Label interface)

Syntax

```
Function GetState_DisplayString : WideString;
```

Description

Example

See also

ISch_Label interface

GetState_CalculatedValueString method

(ISch_Label interface)

Syntax

```
Function GetState_CalculatedValueString : WideString;
```

Description

Example

See also

ISch_Label interface

Properties

Text property

(ISch_Label interface)

Syntax

```
Property Text : WideString Read GetState_Text Write SetState_Text;
```

Description

The Text property of the ISch_Label represents the actual text string. This property is supported by the GetState_Text and SetState_Text methods.

Example

```
Location.X := MilsToCoord(1000);
Location.Y := MilsToCoord(1000);
SchLabel.SetState_Location(Location);
SchLabel.Color       := 12345;
Schlabel.Text        := 'A new name';
SchLabel.FontID      := SchServer.FontManager.GetFontID(14,90,False,False,False,False,'Times New Roman');
```

See also

ISch_Label interface

OverrideDisplayString property

(ISch_Label interface)

Syntax

```
Property OverrideDisplayString : WideString Read GetState_OverrideDisplayString Write SetState_OverrideDisplayString;
```

Description

The OverrrideDisplayString property determines the override display string which overrides the Name string. This property is supported by the GetState_OverrideDisplayString and SetState_OverrideDisplayString methods.

Example

```
DisplayString := SheetEntry.GetState_OverrideDisplayString;
```

See also

ISch_Label interface

Orientation property

(ISch_Label interface)

Syntax

```
Property Orientation : TRotationBy90 Read GetState_Orientation Write SetState_Orientation;
```

Description

This Orientation property determines the angle the ISch_Label is at on the Schematic document. The angle is in 90 degree increments - 0, 90, 180, 270. This property is supported by the GetState_Orientation and SetState_Orientation methods.

However if you are using the FontID property to be assigned by the FontManager (ISch_FontManger interface) then you will need to set the Orientation property as well as passing in the same rotation parameter for the GetFontID method of the ISch_FontManager interface.

Example

```
ALabel.Orientation := eRotate90;
ALabel.FontId      := SchServer.FontManager.GetFontID(14,90,False,False,False,False,'Times New Roman');
```

See also

ISch_Label interface

Justification property

(ISch_Label interface)

Syntax

```
Property Justification : TTextJustification Read GetState_Justification Write SetState_Justification;
```

Description

The Justification property determines the alignment of the text in respect to the Label object whether it is left justified, centered and so on. This property is supported by the GetState_Justification and SetState_Justification methods.

Example

See also

ISch_Label interface

TTextJustification type

IsMirrored property

(ISch_Label interface)

Syntax

```
Property IsMirrored : Boolean Read GetState_IsMirrored Write SetState_IsMirrored;
```

Description

Example

See also

ISch_Label interface

Formula property

(ISch_Label interface)

Syntax

```
Property Formula : WideString Read GetState_Formula;
```

Description

Example

See also

ISch_Label interface

FontId property

(ISch_Label interface)

Syntax

```
Property FontId : TFontID Read GetState_FontId Write SetState_FontId;
```

Description

The FontID property determines the style and type of font for the ISch_Label object on a Schematic document. This property is supported by the GetState_FontID and SetState_FontID methods.

Example

```
Location.X := MilsToCoord(1000);
Location.Y := MilsToCoord(1000);
SchLabel.SetState_Location(Location);
SchLabel.Color       := 12345;
Schlabel.Text        := 'A new name';
SchLabel.FontID      := SchServer.FontManager.GetFontID(14,90,False,False,False,False,'Times New Roman');
```

See also

ISch_Label interface

ISch_FontManager interface

DisplayString property

(ISch_Label interface)

Syntax

```
Property DisplayString : WideString Read GetState_DisplayString;
```

Description

Example

See also

ISch_Label interface

CalculatedValueString property

(ISch_Label interface)

Syntax

```
Property CalculatedValueString : WideString Read GetState_CalculatedValueString;
```

Description

Example

See also

ISch_Label interface

ISch_Line Interface

Overview

Lines are graphical drawing objects with any number of joined segments. A line object is represented by the ISch_Line interface.

Notes

The ISch_Line interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Line

ISch_Line methods

GetState_Corner
GetState_LineWidth
GetState_LineStyle
SetState_Corner
SetState_LineWidth
SetState_LineStyle

ISch_Line properties

Corner
LineWidth
LineStyle

Example

```
Procedure PlaceASchLine;
Var
    SchDoc    : ISch_Document;
    WorkSpace : IWorkSpace;
    SchLine   : ISch_Line;
Begin
    // Generate a blank Schematic document
    WorkSpace := GetWorkSpace;
    If WorkSpace = Nil Then Exit;
    Workspace.DM_CreateNewDocument('SCH');

    // Check if Schematic Editor is active
    If SchServer = Nil Then Exit;
    SchDoc := SchServer.GetCurrentSchDocument;
    If SchDoc = Nil Then Exit;

     // Create a new line and place it on the document.
     SchLine := SchServer.SchObjectFactory(eLine,eCreate_GlobalCopy);
     If SchLine = Nil Then Exit;
     SchLine.Location  := Point(180, 200);
     SchLine.Corner    := Point(180, 400);
     SchLine.LineWidth := eMedium;
     SchLine.LineStyle := eLineStyleSolid;
     SchLine.Color := $FF00FF;
     SchDoc.RegisterSchObjectInContainer(SchLine);
End;
```

See also

ISch_GraphicalObject interface

Methods

SetState_LineStyle method

(ISch_Line interface)

Syntax

```
Procedure SetState_LineStyle (AStyle : TLineStyle);
```

Description

Example

See also

ISch_Line interface

SetState_Corner method

(ISch_Line interface)

Syntax

```
Procedure SetState_Corner (ALocation : TLocation);
```

Description

Example

See also

ISch_Line interface

GetState_LineWidth method

(ISch_Line interface)

Syntax

```
Function GetState_LineWidth : TSize;
```

Description

This GetState_LineWidth function gets the width of the border around the line object. The width is determined by the TSize type.

Example

```
Width := Line.GetState_LineWidth; // Width is of TSize type.
```

See also

TSize type.

ISch_Line interface

GetState_LineStyle method

(ISch_Line interface)

Syntax

```
Function GetState_LineStyle : TLIneStyle;
```

Description

Example

See also

ISch_Line interface

GetState_Corner method

(ISch_Line interface)

Syntax

```
Function GetState_Corner : TLocation;
```

Description

Example

See also

ISch_Line interface

SetState_LineWidth method

(ISch_Line interface)

Syntax

```
Procedure SetState_LineWidth (ASize : TSize);
```

Description

This SetState_LineWidth procedure sets the width of the border line around the line. The width is determined by the TSize type.

Example

```
Line.SetState_LineWidth(eSmall);
```

See also

TSize type.

ISch_Line interface

Properties

LineWidth property

(ISch_Line interface)

Syntax

```
Property LineWidth : TSize Read GetState_LineWidth Write SetState_LineWidth;
```

Description

The LineWidth property defines the border width of the line with one of the following values from the TSize enumerated type. This property is supported by the GetState_LineWidth and SetState_LineWidth methods.

Example

```
Line.LineWidth(eSmall);
```

See also

TSize type.

ISch_Line interface

LineStyle property

(ISch_Line interface)

Syntax

```
Property LineStyle : TLineStyle Read GetState_LineStyle Write SetState_LineStyle;
```

Description

Example

See also

ISch_Line interface

Corner property

(ISch_Line interface)

Syntax

```
Property Corner : TLocation Read GetState_Corner Write SetState_Corner;
```

Description

Example

See also

ISch_Line interface

ISch_NetLabel Interface

Overview

A net describes a connection from one component pin, to a second pin, and then to a third pin and so on. A net label is a text string with the text property that holds the net name that attachs to a connection such as wires. A net label object is represented by the ISch_NetLabel interface.

The ISch_NetLabel interface hierachy is as follows;

ISch_GraphicalObject

ISch_Label

ISch_NetLabel

Text property is the net name of the net label.

ISch_NetLabel itself has no properties or methods but has inherited properties and methods.

ISch_NetLabel methods ISch_NetLabel properties

See also

ISch_GraphicalObject interface

ISch_NoERC Interface

Overview

The NoERC directive is a special symbol that identifies a pin as one that you want the Electrical Rules Checker to ignore.

The ISch_NoERC interface hierarchy is as follows;

ISch_GraphicalObject

ISch_NoERC

ISch_NoERC methods ISch_NoERC properties

See also

ISch_GraphicalObject interface

ISch_Note Interface

Overview

The ISch_Note interface represents the note object on the schematic sheet. This note object stores textual information and can be collapsed upon user's mouse click on the schematic sheet.

The interface hierarchy for the ISch_Note interface is as follows;

ISch_GraphicalObject

ISch_Rectangle

ISch_TextFrame

ISch_Note

ISch_Note methods	ISch_Note properties
SetState_Author	Author
SetState_Collapsed	Collapsed
GetState_Author	
GetState_Collapsed	

See also
ISch_GraphicalObject
ISch_Rectangle
ISch_TextFrame

Methods

SetState_Author method

(ISch_Note interface)

Syntax

```
Procedure SetState_Author (AValue : WideString);
```

Description

Example

See also
ISch_Note interface

GetState_Collapsed method

(ISch_Note interface)

Syntax

```
Function GetState_Collapsed : Boolean;
```

Description

Example

See also
ISch_Note interface

GetState_Author method

(ISch_Note interface)

Syntax

```
Function GetState_Author : WideString;
```

Description

Example

See also
ISch_Note interface

SetState_Collapsed method

(ISch_Note interface)

Syntax

```
Procedure SetState_Collapsed(AValue : Boolean);
```

Description

Example

See also

ISch_Note interface

Properties

Collapsed property

(ISch_Note interface)

Syntax

```
Property Collapsed : Boolean Read GetState_Collapsed Write SetState_Collapsed;
```

Description

Example

See also

ISch_Note interface

Author property

(ISch_Note interface)

Syntax

```
Property Author : WideString Read GetState_Author Write SetState_Author;
```

Description

Example

See also

ISch_Note interface

ISch_Parameter Interface

Overview

There are two types of parameters – system parameters which are owned by a schematic document and parameters owned by certain schematic design objects.

A parameter is a child object of a Parameter Set, Part, Pin, Port, or Sheet Symbol object. A Parameter object has a Name property and Value property which can be used to store information, thus the parameters are a way of defining and associating information and could include strings that identify component manufacturer, date added to the document and also a string for the component's value (e.g. 100K for a resistor or 10PF for a capacitor).

Each parameter has a Unique Id assigned to it. This is used for those parameters that have been added as design rule directives. When transferring the design to the PCB document, any defined rule parameters will be used to generate the relevant design rules in the PCB. These generated rules will be given the same Unique Ids, allowing you to change rule constraints in either schematic or PCB and push the change across when performing a synchronization.

To look for system wide parameters (not associated with a schematic design object), you would set up an iterator to look for parameters. WIth DelphiScript, you will have to define the iteration depth with the method SetState_IterationDepth(eIterateFirstLevel).

The interface hierarchy for the ISch_Parameter interface is as follows;
ISch_GraphicalObject
ISch_Label
ISch_ComplexText
ISch_Parameter

ISch_Parameter methods	ISch_Parameter properties
SetState_ReadOnlyState	Name
SetState_UniqueId	ShowName
SetState_Description	ParamType
SetState_AllowLibrarySynchronize	ReadOnlyState
SetState_AllowDatabaseSynchronize	UniqueId
SetState_Name	Description
SetState_ShowName	AllowLibrarySynchronize
SetState_ParamType	AllowDatabaseSynchronize
GetState_ReadOnlyState	NameIsReadOnly
GetState_UniqueId	ValueIsReadOnly
GetState_Description	IsRule
GetState_AllowLibrarySynchronize	IsSystemParameter
GetState_AllowDatabaseSynchronize	
GetState_Name	
GetState_ShowName	
GetState_ParamType	
GetState_NameIsReadOnly	
GetState_ValueIsReadOnly	
GetState_IsRule	
GetState_IsSystemParameter	

Fetching system (standalone) parameters

Example

```
Procedure FetchParameters;
Var
    CurrentSch : ISch_Sheet;
    Iterator   : ISch_Iterator;
    Parameter  : ISch_Parameter;
Begin
    // Check if schematic server exists or not.
    If SchServer = Nil Then Exit;
    // Obtain the current schematic document interface.
    CurrentSch := SchServer.GetCurrentSchDocument;
    If CurrentSch = Nil Then Exit;

    Iterator := CurrentSch.SchIterator_Create;
    // look for stand alone parameters
    Iterator.SetState_IterationDepth(eIterateFirstLevel);
```

```
    Iterator.AddFilter_ObjectSet(MkSet(eParameter));

    Try
       Parameter := Iterator.FirstSchObject;
       While Parameter <> Nil Do
       Begin
          // do what you want with the parameter
          Parameter := Iterator.NextSchObject;
       End;
    Finally
        CurrentSch.SchIterator_Destroy(Iterator);
    End;
End;
```

See also

ISch_GraphicalObject interface

ISch_Label interface

ISch_ComplexText interface

Methods

SetState_UniqueId method

(ISch_Parameter interface)

Syntax

```
Procedure SetState_UniqueId (S : WideString);
```

Description

The SetState_UniqueID procedure sets the new ID for the parameter. All parameters, sheet symbols, ports, pins, components, openbus links, openbus ports and openbus components have Unique IDs. Unique IDs are used to maintain design synchronization in design projects.

The Unique ID (UID) is an system generated value that uniquely identifies this current parameter. It is used for linking to a PCB document and for project management. Enter a new UID value or click the **Reset** button to generate a new UID for this design object from the Change Properties dialog. You can also globally reset UIDs of components and sheet symbols from the Schematic Editor's **Tools » Convert » Reset Component Unique IDs** menu.

Example

UID := WSM.DM_GenerateUniqueID; // interface and method from Workspace Manager API.

Parameter.SetState_UniqueID(UID);

See also

ISch_Parameter interface

SetState_ShowName method

(ISch_Parameter interface)

Syntax

```
Procedure SetState_ShowName (N : Boolean);
```

Description

Example

See also

ISch_Parameter interface

SetState_ReadOnlyState method

(ISch_Parameter interface)

Syntax

```
Procedure SetState_ReadOnlyState (R : TParameter_ReadOnlyState);
```

Description

Example

See also

ISch_Parameter interface

SetState_ParamType method

(ISch_Parameter interface)

Syntax

```
Procedure SetState_ParamType (N : TParameterType);
```

Description

Example

See also

ISch_Parameter interface

SetState_Name method

(ISch_Parameter interface)

Syntax

```
Procedure SetState_Name (S : WideString);
```

Description

The SetState_Name procedure sets the new name for the parameter object.

Example

```
Parameter.SetState_Name('Parameter Name');
```

See also

ISch_Parameter interface

SetState_Description method

(ISch_Parameter interface)

Syntax

```
Procedure SetState_Description (S : WideString);
```

Description

Example

See also

ISch_Parameter interface

SetState_AllowLibrarySynchronize method

(ISch_Parameter interface)

Syntax

```
Procedure SetState_AllowLibrarySynchronize (B : Boolean);
```

Description

Example

See also

ISch_Parameter interface

SetState_AllowDatabaseSynchronize method

(ISch_Parameter interface)

Syntax

```
Procedure SetState_AllowDatabaseSynchronize(B : Boolean);
```

Description

Example

See also

ISch_Parameter interface

GetState_UniqueId method

(ISch_Parameter interface)

Syntax

```
Function GetState_UniqueId : WideString;
```

Description

Example

See also

ISch_Parameter interface

GetState_ReadOnlyState method

(ISch_Parameter interface)

Syntax

```
Function GetState_ReadOnlyState : TParameter_ReadOnlyState;
```

Description

Example

See also

ISch_Parameter interface

GetState_Description method

(ISch_Parameter interface)

Syntax

```
Function GetState_Description : WideString;
```

Description

Example

See also

ISch_Parameter interface

GetState_AllowLibrarySynchronize method

(ISch_Parameter interface)

Syntax

```
Function GetState_AllowLibrarySynchronize : Boolean;
```

Description

Example

See also

ISch_Parameter interface

GetState_AllowDatabaseSynchronize method

(ISch_Parameter interface)

Syntax

```
Function GetState_AllowDatabaseSynchronize : Boolean;
```

Description

Example

See also

ISch_Parameter interface

GetState_ValueIsReadOnly method

(ISch_Parameter interface)

Syntax

```
Function GetState_ValueIsReadOnly : Boolean;
```

Description

Example

See also

ISch_Parameter interface

GetState_ShowName method

(ISch_Parameter interface)

Syntax

```
Function GetState_ShowName : Boolean;
```

Description

Example

See also

ISch_Parameter interface

GetState_ParamType method

(ISch_Parameter interface)

Syntax

```
Function GetState_ParamType : TParameterType;
```

Description

Example

See also

ISch_Parameter interface

GetState_NameIsReadOnly method

(ISch_Parameter interface)

Syntax

```
Function GetState_NameIsReadOnly : Boolean;
```

Description

Example

See also

ISch_Parameter interface

GetState_Name method

(ISch_Parameter interface)

Syntax

```
Function GetState_Name : WideString;
```

Description

The GetState_Name procedure gets the Parameter Object's name.

Example

```
ParamName := Parameter.GetState_Name;
```

See also

ISch_Parameter interface

GetState_IsSystemParameter method

(ISch_Parameter interface)

Syntax

```
Function GetState_IsSystemParameter : Boolean;
```

Description

Example

See also

ISch_Parameter interface

GetState_IsRule method

(ISch_Parameter interface)

Syntax

```
Function GetState_IsRule : Boolean;
```

Description

Example

See also

ISch_Parameter interface

Properties

ValueIsReadOnly property

(ISch_Parameter interface)

Syntax

```
Property ValueIsReadOnly : Boolean Read GetState_ValueIsReadOnly;
```

Description

Example

See also

ISch_Parameter interface

UniqueId property

(ISch_Parameter interface)

Syntax

```
Property UniqueId : WideString Read GetState_UniqueId Write SetState_UniqueId;
```

Description

The UniqueID property sets the new ID for the parameter. All parameters, sheet symbols, ports, pins, components, openbus links, openbus ports and openbus components have Unique IDs. Unique IDs are used to maintain design synchronization in design projects.

The Unique ID (UID) is an system generated value that uniquely identifies this current parameter. It is used for linking to a PCB document and for project management. Enter a new UID value or click the **Reset** button to generate a new UID for this design object from the Change Properties dialog. You can also globally reset UIDs of components and sheet symbols from the Schematic Editor's **Tools » Convert » Reset Component Unique IDs** menu.

Example

UID := WSM.DM_GenerateUniqueID; // interface and method from Workspace Manager API.

Parameter.UniqueID(UID);

See also

ISch_Parameter interface

ShowName property

(ISch_Parameter interface)

Syntax

```
Property ShowName : Boolean Read GetState_ShowName Write SetState_ShowName;
```

Description

Example

See also

ISch_Parameter interface

ReadOnlyState property

(ISch_Parameter interface)

Syntax

```
Property ReadOnlyState : TParameter_ReadOnlyState Read GetState_ReadOnlyState Write
SetState_ReadOnlyState;
```

Description

Example

See also

ISch_Parameter interface

ParamType property

(ISch_Parameter interface)

Syntax

```
Property ParamType : TParameterType Read GetState_ParamType Write SetState_ParamType;
```

Description

Example

See also

ISch_Parameter interface

NameIsReadOnly property

(ISch_Parameter interface)

Syntax

```
Property NameIsReadOnly : Boolean Read GetState_NameIsReadOnly;
```

Description

Example

See also

ISch_Parameter interface

Name property

(ISch_Parameter interface)

Syntax

```
Property Name : WideString Read GetState_Name Write SetState_Name;
```

Description

The Name property determines the name for the parameter object.

Example

```
ParamName := Parameter.Name;
```

See also

ISch_Parameter interface

Description property

(ISch_Parameter interface)

Syntax

```
Property Description : WideString Read GetState_Description Write SetState_Description;
```

Description

Example

See also

ISch_Parameter interface

AllowLibrarySynchronize property

(ISch_Parameter interface)

Syntax

```
Property AllowLibrarySynchronize : Boolean Read GetState_AllowLibrarySynchronize Write
SetState_AllowLibrarySynchronize;
```

Description

Example

See also

ISch_Parameter interface

AllowDatabaseSynchronize property

(ISch_Parameter interface)

Syntax

```
Property AllowDatabaseSynchronize : Boolean Read GetState_AllowDatabaseSynchronize Write
SetState_AllowDatabaseSynchronize;
```

Description

Example

See also

ISch_Parameter interface

IsSystemParameter property

(ISch_Parameter interface)

Syntax

```
Property IsSystemParameter : Boolean Read GetState_IsSystemParameter;
```

Description

Example

See also

ISch_Parameter interface

IsRule property

(ISch_Parameter interface)

Syntax

```
Property IsRule : Boolean Read GetState_IsRule;
```

Description

Example

See also

ISch_Parameter interface

ISch_ParameterSet Interface

Overview

The ISch_ParameterSet interface is a group of parameters as a design parameter set directive for a wire or a net on the schematic document that can be transferred to its corresponding PCB document.

Notes

The ISch_ParameterSet interface hierarchy is as follows

ISch_GraphicalObject

ISch_ParametrizedGroup

ISch_ParameterSet

ISch_ParameterSet methods

SetState_Orientation

SetState_Name

GetState_Orientation

GetState_Name

ISch_ParameterSet properties

Orientation

Name

See also

ISch_GraphicalObject interface

ISch_ParametrizedGroup interface

Methods

SetState_Name method

(ISch_ParameterSet interface)

Syntax

```
Procedure SetState_Name (AValue : WideString);
```

Description

The SetState_Name procedure sets the new name for the parameterset object.

Example

```
ParameterSet.SetState_Name('Specific Name');
```

See also

ISch_ParameterSet interface

GetState_Orientation method

(ISch_ParameterSet interface)

Syntax

```
Function GetState_Orientation : TRotationBy90;
```

Description

Example

See also

ISch_ParameterSet interface

GetState_Name method

(ISch_ParameterSet interface)

Syntax

```
Function GetState_Name : WideString;
```

Description

The GetState_Name function gets the new name for the parameter set object.

Example

```
Name := ParameterSet.GetState_Name;
```

See also

ISch_ParameterSet interface

SetState_Orientation method

(ISch_ParameterSet interface)

Syntax

```
Procedure SetState_Orientation(AValue : TRotationBy90);
```

Description

Example

See also

ISch_ParameterSet interface

Properties

Orientation property

(ISch_ParameterSet interface)

Syntax

```
Property Orientation : TRotationBy90 Read GetState_Orientation Write SetState_Orientation;
```

Description

Example

See also

ISch_ParameterSet interface

Name property

(ISch_ParameterSet interface)

Syntax

```
Property Name : WideString Read GetState_Name Write SetState_Name;
```

Description

The Name property determines the Parameter Set obejct's name. This property is supported by the GetState_Name and SetState_Name methods.

Example

```
ParamSetName := ParameterSet.Name;
```

See also

ISch_ParameterSet interface

ISch_ParametrizedGroup Interface

Overview

The ISch_ParametrizedGroup is an immediate ancestor interface for ParameterSet, Port, Pin, Component and SheetSymbol interfaces. This interface deals with positions of parameters of such objects..

Notes

The ISch_ParametrizedGroup interface hierarchy is as follows

ISch_GraphicalObject

    ISch_ParameterizedGroup

ISch_ParametrizedGroup methods

Import_FromUser_Parameters

ResetAllSchParametersPosition

ISch_ParametrizedGroup properties

See also

ISch_GraphicalObject ancestor interface

ISch_ParameterSet descendent interface

ISch_Port descendent interface

ISch_Pin descendent interface

ISch_Component descendent interface

ISch_RectangularGroup descendent interface

ISch_SheetSymbol descendent interface

Methods

Import_FromUser_Parameters method

(ISch_ParametrizedGroup interface)

Syntax

```
Function Import_FromUser_Parameters : Boolean;
```

Description

Example

See also

ISch_ParametrizedGroup interface

ResetAllSchParametersPosition method

(ISch_ParametrizedGroup interface)

Syntax

```
Procedure ResetAllSchParametersPosition;
```

Description

Example

See also

ISch_ParametrizedGroup interface

ISch_Pie Interface

Overview

Pie objects are unfilled or filled graphic elements.

Notes

The ISch_Pie interface hierarchy is as follows;

ISch_GraphicalObject

    ISch_Arc

        ISch_Pie

ISch_Pie methods

GetState_IsSolid

ISch_Pie properties

IsSolid

SetState_IsSolid

See also

ISch_Arc interface.

Methods

GetState_IsSolid method

(ISch_Pie interface)

Syntax

```
Function GetState_IsSolid : Boolean;
```

Description

The GetState_IsSolid function returns a Boolean value whether the pie object has a solid internal fill or not.

Example

```
If Pie.GetState_IsSolid Then
    Pie. AreaColor := 0; // black fill
```

See also

ISch_Pie interface

SetState_IsSolid method

(ISch_Pie interface)

Syntax

```
Procedure SetState_IsSolid(B : Boolean);
```

Description

The SetState_IsSolid procedure sets a Boolean value which denotes that the pie object has a solid internal fill or not.

Example

```
Pie.SetState_IsSolid(True);
Pie.AreaColor := 0;
```

See also

ISch_Pie interface

Properties

IsSolid property

(ISch_Pie interface)

Syntax

```
Property IsSolid : Boolean Read GetState_IsSolid Write SetState_IsSolid;
```

Description

The IsSolid property denotes whether the pie object has a solid fill or not. This property is supported by the GetState_IsSolid and SetState_IsSolid methods.

Example

```
Pie.IsSolid := True;
```

See also

ISch_Pie interface

ISch_Pin Interface

Overview

Pins are special objects that have electrical characteristics and are used to direct signals in and out of components. Pins connect directly to other pins, wires, net labels, sheet entries or ports.

Notes

The ISch_Pin interface hierarchy is as follows;

ISch_GraphicalObject

ISch_ParameterizedGroup

ISch_Pin

ISch_Pin methods

SetState_Name
SetState_Designator
SetState_Orientation
SetState_Width
SetState_FormalType
SetState_DefaultValue
SetState_**Description**
SetState_ShowName
SetState_ShowDesignator
SetState_Electrical
SetState_PinLength
SetState_IsHidden
SetState_HiddenNetName
SetState_Symbol_Inner
SetState_Symbol_Outer
SetState_Symbol_InnerEdge
SetState_Symbol_OuterEdge
SetState_SwapIdPart
SetState_SwapIdPin
SetState_SwapIdPartPin
SetState_UniqueId
GetState_Name
GetState_Designator
GetState_Orientation
GetState_Width
GetState_FormalType
GetState_DefaultValue
GetState_**Description**
GetState_ShowName
GetState_ShowDesignator
GetState_Electrical
GetState_PinLength
GetState_IsHidden
GetState_HiddenNetName
GetState_Symbol_Inner
GetState_Symbol_Outer
GetState_Symbol_InnerEdge
GetState_Symbol_OuterEdge
GetState_SwapIdPart
GetState_SwapIdPin

ISch_Pin properties

Name
Designator
Orientation
Width
FormalType
DefaultValue
Description
ShowName
ShowDesignator
Electrical
PinLength
IsHidden
HiddenNetName
Symbol_Inner
Symbol_Outer
Symbol_InnerEdge
Symbol_OuterEdge
SwapId_Part
SwapId_Pin
SwapId_PartPin
UniqueId

GetState_SwapIdPartPin
GetState_UniqueId
OwnerSchComponent
FullDesignator

See also
ISch_GraphicalObject interface
ISch_ParametrizedGroup interface

Methods

GetState_UniqueId method

(ISch_Pin interface)
Syntax
```
Function GetState_UniqueId : WideString;
```
Description

Example

See also
ISch_Pin interface

GetState_Symbol_OuterEdge method

(ISch_Pin interface)
Syntax
```
Function GetState_Symbol_OuterEdge : TIeeeSymbol;
```
Description

Example

See also
ISch_Pin interface

GetState_Symbol_Outer method

(ISch_Pin interface)
Syntax
```
Function GetState_Symbol_Outer : TIeeeSymbol;
```
Description

Example

See also
ISch_Pin interface

GetState_Symbol_InnerEdge method

(ISch_Pin interface)
Syntax
```
Function GetState_Symbol_InnerEdge : TIeeeSymbol;
```
Description

Example

See also

ISch_Pin interface

GetState_Symbol_Inner method

(ISch_Pin interface)

Syntax

```
Function GetState_Symbol_Inner : TIeeeSymbol;
```

Description

Example

See also

ISch_Pin interface

GetState_SwapIdPin method

(ISch_Pin interface)

Syntax

```
Function GetState_SwapIdPin : WideString;
```

Description

Example

See also

ISch_Pin interface

GetState_SwapIdPartPin method

(ISch_Pin interface)

Syntax

```
Function GetState_SwapIdPartPin : WideString;
```

Description

Example

See also

ISch_Pin interface

GetState_SwapIdPart method

(ISch_Pin interface)

Syntax

```
Function GetState_SwapIdPart : WideString;
```

Description

Example

See also

ISch_Pin interface

SetState_Name method

(ISch_Pin interface)

Syntax

```
Procedure SetState_Name (AValue : WideString);
```

Description

The SetState_Name procedure sets the new name for the Pin object.

Example

```
Pin.SetState_Name('40');
```

See also

ISch_Pin interface

SetState_Designator method

(ISch_Pin interface)

Syntax

```
Procedure SetState_Designator (AValue : WideString);
```

Description

Example

See also

ISch_Pin interface

SetState_Width method

(ISch_Pin interface)

Syntax

```
Procedure SetState_Width (AValue : Integer);
```

Description

Example

See also

ISch_Pin interface

SetState_Symbol_OuterEdge method

(ISch_Pin interface)

Syntax

```
Procedure SetState_Symbol_OuterEdge(AValue : TIeeeSymbol);
```

Description

Example

See also

ISch_Pin interface

SetState_Symbol_Outer method

(ISch_Pin interface)

Syntax

```
Procedure SetState_Symbol_Outer (AValue : TIeeeSymbol);
```

Description

Example

See also

ISch_Pin interface

SetState_Symbol_InnerEdge method

(ISch_Pin interface)

Syntax

```
Procedure SetState_Symbol_InnerEdge(AValue : TIeeeSymbol);
```

Description

Example

See also

ISch_Pin interface

SetState_Symbol_Inner method

(ISch_Pin interface)

Syntax

```
Procedure SetState_Symbol_Inner (AValue : TIeeeSymbol);
```

Description

Example

See also

ISch_Pin interface

SetState_SwapIdPart method

(ISch_Pin interface)

Syntax

```
Procedure SetState_SwapIdPart (AValue : WideString);
```

Description

Example

See also

ISch_Pin interface

SetState_ShowName method

(ISch_Pin interface)

Syntax

```
Procedure SetState_ShowName (AValue : Boolean);
```

Description

Example

See also

ISch_Pin interface

SetState_ShowDesignator method

(ISch_Pin interface)

Syntax

```
Procedure SetState_ShowDesignator (AValue : Boolean);
```

Description

Example

See also

ISch_Pin interface

SetState_PinLength method

(ISch_Pin interface)

Syntax

```
Procedure SetState_PinLength (AValue : TCoord);
```

Description

Example

See also

ISch_Pin interface

SetState_Orientation method

(ISch_Pin interface)

Syntax

```
Procedure SetState_Orientation (AValue : TRotationBy90);
```

Description

Example

See also

ISch_Pin interface

SetState_IsHidden method

(ISch_Pin interface)

Syntax

```
Procedure SetState_IsHidden (AValue : Boolean);
```

Description

Example

See also

ISch_Pin interface

SetState_HiddenNetName method

(ISch_Pin interface)

Syntax

```
Procedure SetState_HiddenNetName (AValue : WideString);
```

Description

Example

See also

ISch_Pin interface

SetState_FormalType method

(ISch_Pin interface)

Syntax

```
Procedure SetState_FormalType (AValue : TStdLogicState);
```

Description

Example

See also

ISch_Pin interface

SetState_Electrical method

(ISch_Pin interface)

Syntax

```
Procedure SetState_Electrical (AValue : TPinElectrical);
```

Description

Example

See also

ISch_Pin interface

SetState_Description method

(ISch_Pin interface)

Syntax

```
Procedure SetState_Description (AValue : WideString);
```

Description

Example

See also

ISch_Pin interface

SetState_DefaultValue method

(ISch_Pin interface)

Syntax

```
Procedure SetState_DefaultValue (AValue : WideString);
```

Description

Example

See also

ISch_Pin interface

SetState_UniqueId method

(ISch_Pin interface)

Syntax

```
Procedure SetState_UniqueId (AValue : WideString);
```

Description

The SetState_UniqueID procedure sets the new ID for the pin. All parameters, sheet symbols, ports, pins, components, openbus links, openbus ports and openbus components have Unique IDs. Unique IDs are used to maintain design synchronization in design projects.

The Unique ID (UID) is an system generated value that uniquely identifies this current pin. It is used for linking to a PCB document and for project management. Enter a new UID value or click the **Reset** button to generate a new UID for this design object from the Change Properties dialog. You can also globally reset UIDs of components and sheet symbols from the Schematic Editor's **Tools » Convert » Reset Component Unique IDs** menu.

Example

UID := WSM.DM_GenerateUniqueID; // interface and method from Workspace Manager API.

Pin.SetState_UniqueID(UID);

See also

ISch_Pin interface

SetState_SwapIdPin method

(ISch_Pin interface)

Syntax

```
Procedure SetState_SwapIdPin (AValue : WideString);
```

Description

Example

See also

ISch_Pin interface

SetState_SwapIdPartPin method

(ISch_Pin interface)

Syntax

```
Procedure SetState_SwapIdPartPin (AValue : WideString);
```

Description

Example

See also

ISch_Pin interface

GetState_Width method

(ISch_Pin interface)

Syntax

```
Function GetState_Width : Integer;
```

Description

Example

See also

ISch_Pin interface

GetState_ShowName method

(ISch_Pin interface)

Syntax

```
Function GetState_ShowName : Boolean;
```

Description

Example

See also

ISch_Pin interface

GetState_ShowDesignator method

(ISch_Pin interface)

Syntax

```
Function GetState_ShowDesignator : Boolean;
```

Description

Example

See also

ISch_Pin interface

GetState_PinLength method

(ISch_Pin interface)

Syntax

```
Function GetState_PinLength : TCoord;
```

Description

Example

See also

ISch_Pin interface

GetState_Orientation method

(ISch_Pin interface)

Syntax

```
Function GetState_Orientation : TRotationBy90;
```

Description

Example

See also

ISch_Pin interface

GetState_Name method

(ISch_Pin interface)

Syntax

```
Function GetState_Name : WideString;
```

Description

The GetState_Name function gets the name for the Pin object.

Example

```
PinName := Pin.GetState_Name;
```

See also

ISch_Pin interface

GetState_IsHidden method

(ISch_Pin interface)

Syntax

```
Function GetState_IsHidden : Boolean;
```

Description

Example

See also

ISch_Pin interface

GetState_HiddenNetName method

(ISch_Pin interface)

Syntax

```
Function GetState_HiddenNetName : WideString;
```

Description

Example

See also

ISch_Pin interface

GetState_FormalType method

(ISch_Pin interface)

Syntax

```
Function GetState_FormalType : TStdLogicState;
```

Description

Example

See also

ISch_Pin interface

GetState_Electrical method

(ISch_Pin interface)

Syntax

```
Function GetState_Electrical : TPinElectrical;
```

Description

Example

See also

ISch_Pin interface

GetState_Designator method

(ISch_Pin interface)

Syntax

```
Function GetState_Designator : WideString;
```

Description

Example

See also

ISch_Pin interface

GetState_Description method

(ISch_Pin interface)

Syntax

```
Function GetState_Description : WideString;
```

Description

Example

See also

ISch_Pin interface

GetState_DefaultValue method

(ISch_Pin interface)

Syntax

```
Function GetState_DefaultValue : WideString;
```

Description

Example

See also

ISch_Pin interface

Properties

Width property

(ISch_Pin interface)

Syntax

```
Property Width : Integer Read GetState_Width Write SetState_Width ;
```

Description

Example

See also

ISch_Pin interface

OwnerSchComponent method

(ISch_Pin interface)

Syntax

```
Function OwnerSchComponent : ISch_Component;
```

Description

Example

See also

ISch_Pin interface

Orientation property

(ISch_Pin interface)

Syntax

```
Property Orientation : TRotationBy90 Read GetState_Orientation Write SetState_Orientation ;
```

Description

Example

See also

ISch_Pin interface

Name property

(ISch_Pin interface)

Syntax

```
Property Name : WideString Read GetState_Name Write SetState_Name ;
```

Description

The Name property determines the name for the Pin object. This property is supported by the GetState_Name and SetState_Name methods.

Example

```
PinName := Pin.Name;
```

See also

ISch_Pin interface

FullDesignator method

(ISch_Pin interface)

Syntax

```
Function FullDesignator : WideString;
```

Description

Example

See also

ISch_Pin interface

FormalType property

(ISch_Pin interface)

Syntax

```
Property FormalType : TStdLogicState Read GetState_FormalType Write SetState_FormalType ;
```

Description

Example

See also

ISch_Pin interface

Designator property

(ISch_Pin interface)

Syntax

```
Property Designator : WideString Read GetState_Designator Write SetState_Designator ;
```

Description

Example

See also

ISch_Pin interface

Description property

(ISch_Pin interface)

Syntax

```
Property Description : WideString Read GetState_Description Write SetState_Description ;
```

Description

Example

See also

ISch_Pin interface

DefaultValue property

(ISch_Pin interface)

Syntax

```
Property DefaultValue : WideString Read GetState_DefaultValue Write SetState_DefaultValue ;
```

Description

Example

See also

ISch_Pin interface

UniqueId property

(ISch_Pin interface)

Syntax

```
Property UniqueId : WideString Read GetState_UniqueId Write SetState_UniqueId ;
```

Description

The UniqueID property sets the new ID for the pin. All parameters, sheet symbols, ports, pins, components, openbus links, openbus ports and openbus components have Unique IDs. Unique IDs are used to maintain design synchronization in design projects.

The Unique ID (UID) is an system generated value that uniquely identifies this current sheet symbol. It is used for linking to a PCB document and for project management. Enter a new UID value or click the **Reset** button to generate a new UID for this design object from the Change Properties dialog. You can also globally reset UIDs of components and sheet symbols from the Schematic Editor's **Tools » Convert » Reset Component Unique IDs** menu.

Example

```
UID := WSM.DM_GenerateUniqueID; // interface and method from Workspace Manager API.
Pin.UniqueID(UID);
```

See also

ISch_Pin interface

Symbol_OuterEdge property

(ISch_Pin interface)

Syntax

```
Property Symbol_OuterEdge : TIeeeSymbol Read GetState_Symbol_OuterEdge Write
SetState_Symbol_OuterEdge;
```

Description

Example

See also

ISch_Pin interface

Symbol_Outer property

(ISch_Pin interface)

Syntax

```
Property Symbol_Outer : TIeeeSymbol Read GetState_Symbol_Outer Write SetState_Symbol_Outer ;
```

Description

Example

See also

ISch_Pin interface

Symbol_InnerEdge property

(ISch_Pin interface)

Syntax

```
Property Symbol_InnerEdge : TIeeeSymbol Read GetState_Symbol_InnerEdge Write
SetState_Symbol_InnerEdge;
```

Description

Example

See also

ISch_Pin interface

Symbol_Inner property

(ISch_Pin interface)

Syntax

```
Property Symbol_Inner : TIeeeSymbol Read GetState_Symbol_Inner Write SetState_Symbol_Inner ;
```

Description

Example

See also

ISch_Pin interface

SwapId_Pin property

(ISch_Pin interface)

Syntax

```
Property SwapId_Pin : WideString Read GetState_SwapIdPin Write SetState_SwapIdPin ;
```

Description

Example

See also

ISch_Pin interface

SwapId_PartPin property

(ISch_Pin interface)

Syntax

```
Property SwapId_PartPin : WideString Read GetState_SwapIdPartPin Write SetState_SwapIdPartPin
;
```

Description

Example

See also

ISch_Pin interface

SwapId_Part property

(ISch_Pin interface)

Syntax

```
Property SwapId_Part : WideString Read GetState_SwapIdPart Write SetState_SwapIdPart ;
```

Description

Example

See also

ISch_Pin interface

ShowName property

(ISch_Pin interface)

Syntax

```
Property ShowName : Boolean Read GetState_ShowName Write SetState_ShowName ;
```

Description

Example

See also

ISch_Pin interface

ShowDesignator property

(ISch_Pin interface)

Syntax

```
Property ShowDesignator : Boolean Read GetState_ShowDesignator Write SetState_ShowDesignator ;
```

Description

Example

See also

ISch_Pin interface

PinLength property

(ISch_Pin interface)

Syntax

```
Property PinLength : TCoord Read GetState_PinLength Write SetState_PinLength ;
```

Description

Example

See also

ISch_Pin interface

IsHidden property

(ISch_Pin interface)

Syntax

```
Property IsHidden : Boolean Read GetState_IsHidden Write SetState_IsHidden ;
```

Description

Example

See also

ISch_Pin interface

HiddenNetName property

(ISch_Pin interface)

Syntax

```
Property HiddenNetName : WideString Read GetState_HiddenNetName Write SetState_HiddenNetName ;
```

Description

Example

See also

ISch_Pin interface

Electrical property

(ISch_Pin interface)

Syntax

```
Property Electrical : TPinElectrical Read GetState_Electrical Write SetState_Electrical ;
```

Description

Example

See also

ISch_Pin interface

ISch_Polygon Interface

Overview

Polygons are multi-sided graphical elements. The vertices of a polygon object denote the link of lines to describe its outline.

The ISch_Polygon interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Polygon interface

ISch_Polygon methods

SetState_LineWidth
SetState_IsSolid
SetState_Vertex
SetState_VerticesCount
SetState_Transparent
GetState_LineWidth
GetState_IsSolid
GetState_Vertex
GetState_VerticesCount
GetState_Transparent
InsertVertex
RemoveVertex
ClearAllVertices

ISch_Polygon properties

IsSolid
LineWidth
Vertex
VerticesCount
Transparent

See also

ISch_GraphicalObject interface
ISch_Polyline interface
ISch_Wire interface
ISch_Bus interface
TLocation values
TSize enumerated values

Methods

SetState_LineWidth method

(ISch_Polygon interface)

Syntax

```
Procedure SetState_LineWidth (AValue : TSize);
```

Description

This SetState_LineWidth procedure sets the width of the border line around the polygon. The width is determined by the TSize type.

Example

```
Polygon.SetState_LineWidth(eSmall);
```

See also

TSize type.

ISch_Polygon interface

SetState_VerticesCount method

(ISch_Polygon interface)

Syntax

```
Procedure SetState_VerticesCount(AValue : Integer);
```

Description

Example

See also

ISch_Polygon interface

SetState_Vertex method

(ISch_Polygon interface)

Syntax

```
Procedure SetState_Vertex (i : Integer; ALocation : TLocation);
```

Description

Example

See also

ISch_Polygon interface

SetState_Transparent method

(ISch_Polygon interface)

Syntax

```
Procedure SetState_Transparent (B : Boolean);
```

Description

Example

See also

ISch_Polygon interface

SetState_IsSolid method

(ISch_Polygon interface)

Syntax

```
Procedure SetState_IsSolid (AValue : Boolean);
```

Description

Example

See also

ISch_Polygon interface

GetState_VerticesCount method

(ISch_Polygon interface)

Syntax

```
Function GetState_VerticesCount : Integer;
```

Description

Example

See also

ISch_Polygon interface

GetState_Vertex method

(ISch_Polygon interface)

Syntax

```
Function GetState_Vertex(i : Integer) : TLocation;
```

Description

Example

See also

ISch_Polygon interface

GetState_Transparent method

(ISch_Polygon interface)

Syntax

```
Function GetState_Transparent : Boolean;
```

Description

Example

See also

ISch_Polygon interface

GetState_LineWidth method

(ISch_Polygon interface)

Syntax

```
Function GetState_LineWidth : TSize;
```

Description

This GetState_LineWidth procedure gets the width of the border line around the line. The width is determined by the TSize type.

Example

```
LineWidth := Polygon.GetState_LineWidth;
```

See also

ISch_Polygon interface

GetState_IsSolid method

(ISch_Polygon interface)

Syntax

```
Function GetState_IsSolid : Boolean;
```

Description

Example

See also

ISch_Polygon interface

RemoveVertex method

(ISch_Polygon interface)

Syntax

```
Function RemoveVertex (Var Index : Integer) : Boolean;
```

Description

Example

See also

ISch_Polygon interface

InsertVertex method

(ISch_Polygon interface)

Syntax

```
Function InsertVertex ( Index : Integer) : Boolean;
```

Description

Example

See also

ISch_Polygon interface

ClearAllVertices method

(ISch_Polygon interface)

Syntax

```
Procedure ClearAllVertices;
```

Description

Example

See also

ISch_Polygon interface

Properties

VerticesCount property

(ISch_Polygon interface)

Syntax

```
Property VerticesCount : Integer Read GetState_VerticesCount Write Setstate_VerticesCount;
```

Description

Example

See also

ISch_Polygon interface

Transparent property

(ISch_Polygon interface)

Syntax

```
Property Transparent : Boolean Read GetState_Transparent Write SetState_Transparent;
```

Description

Example

See also

ISch_Polygon interface

LineWidth property

(ISch_Polygon interface)

Syntax

```
Property LineWidth : TSize Read GetState_LineWidth Write SetState_LineWidth;
```

Description

The LineWidth property defines the border width of the polygon with one of the following values from the TSize enumerated type. This property is supported by the GetState_LineWidth and SetState_LineWidth methods.

Example

Polygon.LineWIdth := eSmall;

See also

TSize type

ISch_Polygon interface

IsSolid property

(ISch_Polygon interface)

Syntax

```
Property IsSolid : Boolean Read GetState_IsSolid Write SetState_IsSolid;
```

Description

Example

See also

ISch_Polygon interface

Vertex property

(ISch_Polygon interface)

Syntax

```
Property Vertex[i : Integer] : TLocation Read GetState_Vertex Write SetState_Vertex;
```

Description

Example

See also

ISch_Polygon interface

TLocation type

ISch_BasicPolyline Interface

Overview

Lines are graphical drawing objects with any number of joined segments.

Notes

The ISch_BasicPolyline interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Polygon

ISch_BasicPolyline

ISch_Polyline

ISch_BasicPolyline methods

SetState_LineStyle

GetState_LineStyle

ISch_BasicPolyline properties

LineStyle

See also

ISch_GraphicalObject interface

ISch_Polygon interface

ISch_Polyline interface

Methods

GetState_LineStyle method

(ISch_BasicPolyline interface)

Syntax

```
Function GetState_LineStyle : TLineStyle;
```

Description

Example

See also

ISch_BasicPolyline interface

SetState_LineStyle method

(ISch_BasicPolyline interface)

Syntax

```
Procedure SetState_LineStyle(AValue : TLineStyle);
```

Description

Example

See also

ISch_BasicPolyline interface

Properties

LineStyle property

(ISch_BasicPolyline interface)

Syntax

```
Property LineStyle : TLineStyle Read GetState_LineStyle Write SetState_LineStyle;
```

Description

Example

See also

ISch_BasicPolyline interface

ISch_Polyline Interface

Overview

Lines are graphical drawing objects with any number of joined segments.

Notes

The ISch_Polyline interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Polygon

ISch_BasicPolyline

ISch_Polyline

ISch_Polyline methods

GetState_StartLineShape
SetState_StartLineShape
GetState_EndLineShape
SetState_EndLineShape
GetState_LineShapeSize
SetState_LineShapeSize

ISch_Polyline properties

StartLineShape
EndLineShape
LineShapeSize

See also

ISch_GraphicalObject interface
ISch_Polygon interface
ISch_BasicPolyline interface

Methods

GetState_StartLineShape method

(ISch_Polyline interface)

Syntax

```
Function  GetState_StartLineShape        : TLineShape;
```

Description

Example

See also

ISch_Polyline interface

GetState_EndLineShape method

(ISch_Polyline interface)

Syntax

```
Function  GetState_EndLineShape          : TLineShape;
```

Description

Example

See also

ISch_Polyline interface

GetState_LineShapeSize method

(ISch_Polyline interface)

Syntax

```
Function  GetState_LineShapeSize         : TSize;
```

Description

Example

See also

ISch_Polyline interface

SetState_StartLineShape method

(ISch_Polyline interface)

Syntax

```
Procedure SetState_StartLineShape(AValue : TLineShape);
```

Description

Example

See also

ISch_Polyline interface

SetState_EndLineShape method

(ISch_Polyline interface)

Syntax

```
Procedure SetState_EndLineShape  (AValue : TLineShape);
```

Description

Example

See also

ISch_Polyline interface

SetState_LineShapeSize method

(ISch_Polyline interface)

Syntax

```
Procedure SetState_LineShapeSize (AValue : TSize);
```

Description

Example

See also

ISch_Polyline interface

Properties

LineStyle property

(ISch_Polyline interface)

Syntax

```
Property LineStyle : TLineStyle Read GetState_LineStyle Write SetState_LineStyle;
```

Description

Example

See also

ISch_Polyline interface

ISch_Port Interface

Overview

A port is used to connect a net on one sheet to Ports with the same name on other sheets. Ports can also connect from child sheets to Sheet entries, in the appropriate sheet symbol on the parent sheet.

The port cross referencing information for ports on different schematics linked to sheet entries of a sheet symbol can be added to schematic sheets by executing the Reports » Port Cross Reference » Add To Sheet or Add to Project command within Schematic Editor in Altium Designer.

Notes

To obtain the cross reference field of a port, the design project needs to be compiled first and then port cross-referencing information added to the project or the sheet.

Port cross references are a calculated attribute of ports, they can not be edited and are not stored with the design.

The location of each port reference is determined by the location of the port on the sheet and the position of the connecting wire.

The CrossReference property returns the name of the sheet the port is linked to and the grid where the port is located at. **Example** : 4 Port Serial Interface [3C].

The ISch_Port hierarchy is as follows;

ISch_GraphicalObject

ISch_ParametrizedGroup

ISch_Port

ISch_Port methods

SetState_Name
SetState_Style
SetState_IOType
SetState_Alignment
SetState_TextColor
SetState_Width
SetState_CrossRef
SetState_UniqueId
SetState_ConnectedEnd
SetState_OverrideDisplayString

GetState_Name
GetState_Style
GetState_IOType

ISch_Port properties

Name
Style
IOType
Alignment
TextColor
Width
CrossReference
UniqueId
ConnectedEnd
OverrideDisplayString

GetState_Alignment
GetState_TextColor
GetState_Width
GetState_CrossRef
GetState_UniqueId
GetState_ConnectedEnd
GetState_OverrideDisplayString

IsVertical

See also

ISch_GraphicalObject interface
ISch_ParametrizedGroup interface

Methods

SetState_Width method

(ISch_Port interface)

Syntax

```
Procedure SetState_Width (AValue : TCoord);
```

Description

This SetState_Width procedure sets the width of the port object in a TCoord value. Use one of the following conversion functions to convert from a unit value to a TCoord value. For example MilsToCoord or DXPsToCoord functions.

Example

```
Port.SetState_Width(MilsToCoord(50));
```

See also

TCoord type.
Conversion functions
ISch_Port interface

SetState_UniqueId method

(ISch_Port interface)

Syntax

```
Procedure SetState_UniqueId (AValue : WideString);
```

Description

The SetState_UniqueID procedure sets the new ID for the port. All parameters, sheet symbols, ports, pins, components, openbus links, openbus ports and openbus components have Unique IDs. Unique IDs are used to maintain design synchronization in design projects.

The Unique ID (UID) is an system generated value that uniquely identifies this current port. It is used for linking to a PCB document and for project management. Enter a new UID value or click the **Reset** button to generate a new UID for this design object from the Change Properties dialog. You can also globally reset UIDs of components and sheet symbols from the Schematic Editor's **Tools » Convert » Reset Component Unique IDs** menu.

Example

```
UID := WSM.DM_GenerateUniqueID; // interface and method from Workspace Manager API.
Port.SetState_UniqueID(UID);
```

See also

ISch_Port interface

SetState_TextColor method

(ISch_Port interface)

Syntax

```
Procedure SetState_TextColor (AValue : TColor);
```

Description

The SetState_TextColor procedure sets the color (a value of TColor type) for the Port's Name string.

Notes

The TColor value specifies a 6 digit hexadecimal number of the $FFFFFF format. For example the color blue would be RGB:0,0,255 and Hex:FF0000 therefore the converted decimal value would be 16711680. The following formula may be used to calculate the required value, R+256*(G+(256*B)).

Example

```
Port.SetState_TextColor(0); // sets the text color to black.
```

See also

TColor type

ISch_Port interface

SetState_Style method

(ISch_Port interface)

Syntax

```
Procedure SetState_Style (AValue : TPortArrowStyle);
```

Description

The SetState_Style procedure sets the style of the port. This style is determined by the TPortArrowStyle type. This style defines the graphical style of the port.

Example

```
Port.SetState_Style(ePortLeft);
```

See also

TPortArrowStyle type

ISch_Port interface

SetState_Name method

(ISch_Port interface)

Syntax

```
Procedure SetState_Name (AValue : WideString);
```

Description

The SetState_Name procedure sets the new name for the Port object.

Example

```
Port.SetState_Name('Port Name');
```

See also

ISch_Port interface

SetState_IOType method

(ISch_Port interface)

Syntax

```
Procedure SetState_IOType (AValue : TPortIO);
```

Description

The SetState_IOType procedure defines the electrical properties of the port with the TPortIO type. Available Port IO types are: Input, Output, Bi-directional and Unspecified.

The setting of this IO Type does not influence the connectivity of the circuit, but is considered during the running of an electrical rules check, which can be set to detect incompatible port directions.

Example

```
Port.SetState_IOType(ePortBidirectional);
```

See also

ISch_Port interface

SetState_CrossRef method

(ISch_Port interface)

Syntax

```
Procedure SetState_CrossRef (AValue : WideString);
```

Description

Example

See also

ISch_Port interface

SetState_ConnectedEnd method

(ISch_Port interface)

Syntax

```
Procedure SetState_ConnectedEnd(AValue : TPortConnectedEnd);
```

Description

The SetState_ConnectedEnd procedure sets the ConnectedEnd type of the port object which determines how the port is graphically connected.

Example

```
Port.SetState_ConenctedEnd(ePortConnectedEnd_Origin);
```

See also

TPortConnectedEnd;

ISch_Port interface

SetState_Alignment method

(ISch_Port interface)

Syntax

```
Procedure SetState_Alignment (AValue : THorizontalAlign);
```

Description

The SetState_Alignment function determines how the port's Name is aligned with respect to the ends of the port itself. The Name string can be left justified, centered or right justified with respect to the port object.

Example

```
Port.SetState_Alignment(eHorizontalCentreAlign);
```

See also

THorizontalAlign type

ISch_Port interface

SetState_OverrideDisplayString method

(ISch_Port interface)

Syntax

```
Procedure SetState_OverrideDisplayString(AValue  : WideString);
```

Description

The SetState_OverrrideDisplayString function sets the override display string which overrides the Name string.

Example

```
Port.SetState_OverrideDisplayString('Override Name');
```

See also

ISch_Port interface

GetState_Style method

(ISch_Port interface)

Syntax

```
Function GetState_Style : TPortArrowStyle;
```

Description

The GetState_Style procedure gets the style of the port. This style is determined by the TPortArrowStyle type. This style defines the graphical style of the port object.

Example

```
Port.GetState_Style(ePortLeft);
```

See also

TPortArrowStyle type

ISch_Port interface

GetState_Name method

(ISch_Port interface)

Syntax

```
Function GetState_Name : WideString;
```

Description

The GetState_Name procedure gets the name for the port object.

Example

```
PortName := Port.GetState_Name;
```

See also

ISch_Port interface

GetState_Width method

(ISch_Port interface)

Syntax

```
Function GetState_Width : TCoord;
```

Description

The GetState_Width function gets the width of the port in TCoord type. Use one of the following conversion functions to convert from a TCoord value to one of these Unit values. For example CoordToMils or CoordToDxps functions.

Example

```
Port.Width(DXPsToCoord(50));
```

See also

Conversion functions

ISch_Port interface

GetState_UniqueId method

(ISch_Port interface)

Syntax

```
Function GetState_UniqueId : WideString;
```

Description

The GetState_UniqueID procedure gets the ID for the port. All parameters, sheet symbols, ports, pins, components, openbus links, openbus ports and openbus components have Unique IDs. Unique IDs are used to maintain design synchronization in design projects.

The Unique ID (UID) is an system generated value that uniquely identifies this current port. It is used for linking to a PCB document and for project management. Enter a new UID value or click the **Reset** button to generate a new UID for this design

object from the Change Properties dialog. You can also globally reset UIDs of components and sheet symbols from the Schematic Editor's **Tools » Convert » Reset Component Unique IDs** menu.

Example

```
UID := Port.GetState_UniqueID;
```

See also

ISch_Port interface

GetState_TextColor method

(ISch_Port interface)

Syntax

```
Function GetState_TextColor : TColor;
```

Description

The GetState_TextColor procedure gets the color (a value of TColor type) from the Port's Name string.

Notes

The TColor value specifies a 6 digit hexadecimal number of the $FFFFFF format. For example the color blue would be RGB:0,0,255 and Hex:FF0000 therefore the converted decimal value would be 16711680. The following formula may be used to calculate the required value, R+256*(G+(256*B)).

Example

```
Color := Port.GetState_TextColor;
```

See also

TColor type

ISch_Port interface

GetState_IOType method

(ISch_Port interface)

Syntax

```
Function GetState_IOType : TPortIO;
```

Description

The GetState_IOType function retrieves the electrical properties of the port of the TPortIO type. Available Port IO types are: Input, Output, Bi-directional and Unspecified.

The setting of this IO Type does not influence the connectivity of the circuit, but is considered during the running of an electrical rules check, which can be set to detect incompatible port directions.

Example

```
IOType := Port.GetState_IOType;
```

See also

ISch_Port interface

GetState_CrossRef method

(ISch_Port interface)

Syntax

```
Function GetState_CrossRef : WideString;
```

Description

The GetState_CrossRef function returns the text of the parameter associated with the port. The Parameter has a Name of 'CrossRef'.

Example

See also

ISch_Port interface

GetState_ConnectedEnd method

(ISch_Port interface)

Syntax

```
Function GetState_ConnectedEnd : TPortConnectedEnd;
```

Description

The GetState_ConnectedEnd procedure gets the ConnectedEnd type of the port object which determines how the port is graphically connected.

Example

```
ConnectedEnd := Port.GetState_ConnectedEnd;
```

See also

ISch_Port interface

GetState_Alignment method

(ISch_Port interface)

Syntax

```
Function GetState_Alignment : THorizontalAlign;
```

Description

The GetState_Alignment function determines how the port's Name is aligned with respect to the ends of the port itself. The Name string can be left justified, centered or right justified in respect to the port object.

Example

```
Align := Port.GetState_Alignment;
```

See also

ISch_Port interface

GetState_OverrideDisplayString method

(ISch_Port interface)

Syntax

```
Function  GetState_OverrideDisplayString : WideString;
```

Description

The GetState_OverrrideDisplayString function returns the override display string which overrides the Name string.

Example

```
DisplayString := Port.GetState_OverrideDisplayString;
```

See also

ISch_Port interface

IsVertical method

(ISch_Port interface)

Syntax

```
Function IsVertical : Boolean;
```

Description

This function returns a Boolean value that determines whether the port object is aligned vertically or not.

Example

```
Vertical := Port.IsVertical;
```

See also

ISch_Port interface

Properties

Width property

(ISch_Port interface)

Syntax

```
Property Width : TCoord Read GetState_Width Write SetState_Width ;
```

Description

Example

See also

ISch_Port interface

UniqueId property

(ISch_Port interface)

Syntax

```
Property UniqueId : WideString Read GetState_UniqueId Write SetState_UniqueId ;
```

Description

The UniqueID property sets the new ID for the port. All parameters, sheet symbols, ports, pins, components, openbus links, openbus ports and openbus components have Unique IDs. Unique IDs are used to maintain design synchronization in design projects.

The Unique ID (UID) is an system generated value that uniquely identifies this current port. It is used for linking to a PCB document and for project management. Enter a new UID value or click the **Reset** button to generate a new UID for this design object from the Change Properties dialog. You can also globally reset UIDs of components and sheet symbols from the Schematic Editor's **Tools » Convert » Reset Component Unique IDs** menu.

Example

```
UID := WSM.DM_GenerateUniqueID; // interface and method from Workspace Manager API.
Port.UniqueID(UID);
```

See also

ISch_Port interface

TextColor property

(ISch_Port interface)

Syntax

```
Property TextColor : TColor Read GetState_TextColor Write SetState_TextColor;
```

Description

The TextColor property determines the color (a value of TColor type) of the Port's Name string. This property is supported by the GetState_TextColor and SetState_TextColor methods.

Notes

The TColor value specifies a 6 digit hexadecimal number of the $FFFFFF format. For example the color blue would be RGB:0,0,255 and Hex:FF0000 therefore the converted decimal value would be 16711680. The following formula may be used to calculate the required value, R+256*(G+(256*B)).

Example

```
Color := Port.TextColor;
```

See also

TColor type

ISch_Port interface

Style property

(ISch_Port interface)

Syntax

```
Property Style : TPortArrowStyle Read GetState_Style Write SetState_Style ;
```

Description

The Style property determines the style of the port object. This style is determined by the TPortArrowStyle type. This style defines the graphical style of the port object.

Example

```
Port.Style := ePortLeft;
```

See also

TPortArrowStyle type

ISch_Port interface

Name property

(ISch_Port interface)

Syntax

```
Property Name : WideString Read GetState_Name Write SetState_Name ;
```

Description

The Name property determines the name for the port object. This property is supported by the GetState_Name and SetState_Name methods.

Example

```
PortName := Port.Name;
```

See also

ISch_Port interface

IOType property

(ISch_Port interface)

Syntax

```
Property IOType : TPortIO Read GetState_IOType Write SetState_IOType ;
```

Description

The IOType property defines the electrical properties of the port with the TPortIO type. Available Port IO types are: Input, Output, Bi-directional and Unspecified.

The setting of this IO Type does not influence the connectivity of the circuit, but is considered during the running of an electrical rules check, which can be set to detect incompatible port directions.

Example

```
PortIOType := Port.IOType;
```

See also

TPortIO type

ISch_Port interface

CrossReference property

(ISch_Port interface)

Syntax

```
Property CrossReference : WideString Read GetState_CrossRef Write SetState_CrossRef ;
```

Description

Port Cross References are text that show which schematic sheets the ports are linked to with the zone reference information in brackets. For example a port with A[0..2] name on 4 Port UART and Line Drivers.SchDoc will have a string with "ISA and Address Decoding[4C]" and the 4C string represents the location (reference zone markers around the schematic sheet) of the matching port on ISA and Address Decoding.SchDoc. The string in the [] bracket is dependent on the Port Cross References options in the **Schematic - General** page of the *Preferences* dialog.

Port Cross References are generated from the **Reports » Port Cross Reference » Add to Sheet** or **Reports » Port Cross Reference » Add to Project** commands in the Schematic Editor.

The CrossReference property is supported by the GetState_CrossRef and SetState_CrossRef methods. The CrossRef string is also represented as a parameter associated with this port object AFTER the port cross reference command from the Reports menu has been invoked.

Example

```
Var
    Port         : ISch_Port;
    CurrentSheet : ISch_Document;
    Iterator     : ISch_Iterator;
    Report         : TStringList;
    ReportDocument : IServerDocument;
    S              : WideString;
Begin
    // Obtain the current schematic sheet interface.
    CurrentSheet := SchServer.GetCurrentSchDocument;
    If CurrentSheet = Nil Then Exit;

    Report := TStringList.Create;
    Iterator := CurrentSheet.SchIterator_Create;
    Iterator.AddFilter_ObjectSet(MkSet(ePort));

    Try
        Port := Iterator.FirstSchObject;
        While Port <> Nil Do
        Begin
            If Port.Getstate_CrossRef <> '' Then
                Report.Add('Port:' + Port.Name + '''s cross reference: ' +
Port.GetState_CrossRef)
            Else
                Report.Add('Port:' + Port.Name + ' does not have a cross reference.');

            Port := Iterator.NextSchObject;
        End;
    Finally
        CurrentSheet.SchIterator_Destroy(Iterator);
    End;

    S := 'C:\PortReport.Txt';
    Report.SaveToFile(S);
    Report.Free;

    ReportDocument := Client.OpenDocument('Text', S);
    If ReportDocument <> Nil Then
        Client.ShowDocument(ReportDocument);
End;
```

See also

ISch_Port interface

GetState_CrossRef and SetState_CrossRef methods of ISch_Port interface.

ConnectedEnd property

(ISch_Port interface)

Syntax

```
Property ConnectedEnd : TPortConnectedEnd Read GetState_ConnectedEnd Write
SetState_ConnectedEnd;
```

Description

The ConnectedEnd property determines how a port object is connected graphically with the `TPortConnectedEnd` type. This property is supported by the GetState_ConnectedEnd and SetState_ConnectedEnd methods.

Example

```
Port.ConnectedEnd := ePortConnectedEnd_Extremity; // connected at the other end
```

See also

TPortConnectedEnd type

ISch_Port interface

Alignment property

(ISch_Port interface)

Syntax

```
Property Alignment : THorizontalAlign Read GetState_Alignment Write SetState_Alignment;
```

Description

The Alignment property determines how the port's Name is aligned with respect to the ends of the port itself. The Name string can be left justified, centered or right justified. This property is supported by the GetState_Alignment and SetState_Alignment methods.

Example

```
Port.Alignment := eHorizontalCentreAlign;
```

See also

THorizontalAlign type

ISch_Port interface

OverrideDisplayString property

(ISch_Port interface)

Syntax

```
Property OverrideDisplayString : WideString        Read GetState_OverrideDisplayString Write
SetState_OverrideDisplayString;
    End;
```

Description

The OverrrideDisplayString property determines the override display string which overrides the Name string. This property is supported by the GetState_OverrideDisplayString and SetState_OverrideDisplayString methoes.

Example

```
DisplayString := SheetEntry.GetState_OverrideDisplayString;
```

See also

ISch_Port interface

ISch_PowerObject Interface

Overview

Power ports are special symbols that represent a power supply and are always identified by their net names. The Text property is the net name of the power object.

Notes

The ISch_PowerObject interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Label

ISch_PowerObject

ISch_PowerObject methods

SetState_Style

GetState_Style

SetState_ShowNetName

GetState_ShowNetName

ISch_PowerObject properties

Style

ShowNetName

See also

ISch_GraphicalObject interface

ISch_Label interface

Methods

SetState_ShowNetName method

(ISch_PowerObject interface)

Syntax

```
Procedure SetState_ShowNetName(AValue : Boolean)
```

Description

Example

See also

ISch_PowerObject interface

SetState_Style method

(ISch_PowerObject interface)

Syntax

```
Procedure SetState_Style(AStyle : TPowerObjectStyle);
```

Description

The SetState_Style procedure sets the style of the power object. This style is determined by the TPowerObjectStyle type. This style defines the graphical style of the power object. Available styles are: Circle, Arrow, Wave, Bar, Power Ground, Signal Ground and Earth. Note: The graphical style of a power object has no influence on the net to which it is assigned and does not define any electrical characteristics of the object.

Example

PowerObject.SetState_Style(ePowerGndEarth);

See also

ISch_PowerObject interface

GetState_Style method

(ISch_PowerObject interface)

Syntax

```
Function GetState_Style : TPowerObjectStyle;
```

Description

The GetState_Style function gets the style of the power object. This style is determined by the TPowerObjectStyle type. This style defines the graphical style of the power object. Available styles are: Circle, Arrow, Wave, Bar, Power Ground, Signal Ground and Earth. Note: The graphical style of a power object has no influence on the net to which it is assigned and does not define any electrical characteristics of the object.

Example

PowerStyle := PowerObject.GetState_Style;

See also

TPowerObjectStyle type

ISch_PowerObject interface

GetState_ShowNetName method

(ISch_PowerObject interface)

Syntax

```
Function  GetState_ShowNetName        : Boolean;
```

Description

Example

See also

ISch_PowerObject interface

Properties

Style property

(ISch_PowerObject interface)

Syntax

```
Property Style : TPowerObjectStyle Read GetState_Style Write SetState_Style;
```

Description

This property denotes the style of the power object. This property is supported by the GetState_Style and SetState_Style methods.

Example

See also

ISch_PowerObject interface

TPowerObjectStyle type

ShowNetName property

(ISch_PowerObject interface)

Syntax

```
Property ShowNetName : Boolean Read GetState_ShowNetName Write SetState_ShowNetName;
```

Description

This property denotes the visibility of the net name of the power object. This property is supported by the GetState_ShowNetName and SetState_ShowNetName methods.

Example

See also

ISch_PowerObject interface

TPowerObjectStyle type

ISch_Probe Interface

Overview

A probe is a special marker which is placed on a schematic document to identify nodes for digital simulation.

Notes

The ISch_Probe interface hierarchy is as follows;

ISch_GraphicalObject

ISch_ParametrizedGroup

ISch_ParameterSet

ISch_Probe

ISch_Probe methods **ISch_Probe properties**

See also

ISch_GraphicalObject interface

ISch_ParametrizedGroup interface

ISch_ParameterSet interface

ISch_Rectangle Interface

Overview

Rectangles are drawing objects which are unfilled or filled graphic elements.

Notes

The ISch_Rectangle interface hierarchy is as follows;

ISch_GraphicalObject interface

ISch_Rectangle interface

ISch_Rectangle methods

SetState_Corner
SetState_LineWidth
SetState_IsSolid
SetState_Transparent
GetState_Corner
GetState_LineWidth
GetState_IsSolid
GetState_Transparent

ISch_Rectangle properties

Corner
LineWidth
IsSolid
Transparent

See also

ISch_GraphicalObject interface

Methods

SetState_Transparent method

(ISch_Rectangle interface)

Syntax

```
Procedure SetState_Transparent(B : Boolean);
```

Description

Example

See also

ISch_Rectangle interface

SetState_LineWidth method

(ISch_Rectangle interface)

Syntax

```
Procedure SetState_LineWidth (ASize : TSize);
```

Description

The SetState_LineWidth procedure sets the line width for the border of the rectangle object. The Line width is determined by the TSize type.

Example

```
Rectangle.SetState_LineWidth(eSmall);
```

See also

TSize type.

ISch_Rectangle interface

SetState_IsSolid method

(ISch_Rectangle interface)

Syntax

```
Procedure SetState_IsSolid (B : Boolean);
```

Description

Example

See also

ISch_Rectangle interface

SetState_Corner method

(ISch_Rectangle interface)

Syntax

```
Procedure SetState_Corner (ALocation : TLocation);
```

Description

Example

See also

ISch_Rectangle interface

GetState_Transparent method

(ISch_Rectangle interface)

Syntax

```
Function GetState_Transparent : Boolean;
```

Description

Example

See also

ISch_Rectangle interface

GetState_LineWidth method

(ISch_Rectangle interface)

Syntax

```
Function GetState_LineWidth : TSize;
```

Description

The GetState_LineWidth function returns the line width of the rectangle’s border. The line width is determined by the TSize type.

Example

```
Width := Rectangle.GetState_LineWidth;
```

See also

TSize type.

ISch_Rectangle interface

GetState_IsSolid method

(ISch_Rectangle interface)

Syntax

```
Function GetState_IsSolid : Boolean;
```

Description

Example

See also

ISch_Rectangle interface

GetState_Corner method

(ISch_Rectangle interface)

Syntax

```
Function GetState_Corner : TLocation;
```

Description

Example

See also

ISch_Rectangle interface

Properties

LineWidth property

(ISch_Rectangle interface)

Syntax

```
Property LineWidth : TSize Read GetState_LineWidth Write SetState_LineWidth;
```

Description

The LineWidth property defines the border width of the rectangle with one of the following values from the TSize enumerated type. This property is supported by the GetState_LineWidth and SetState_LineWidth methods.

Example

Rect.LineWidth := eSmall;

See also

TSize type.

ISch_Rectangle interface

IsSolid property

(ISch_Rectangle interface)

Syntax

```
Property IsSolid : Boolean Read GetState_IsSolid Write SetState_IsSolid;
```

Description

Example

See also

ISch_Rectangle interface

Corner property

(ISch_Rectangle interface)

Syntax

```
Property Corner : TLocation Read GetState_Corner Write SetState_Corner;
```

Description

Example

See also

ISch_Rectangle interface

Transparent property

(ISch_Rectangle interface)

Syntax

```
Property Transparent : Boolean Read GetState_Transparent Write SetState_Transparent;
```

Description

Example

See also

ISch_Rectangle interface

ISch_RectangularGroup Interface

Overview

The ISch_RectangularGroup interface represents a group rectangular object with the size of the object with XSize and YSize dimensions. The Origin of the rectangular object is the Location property from the ISch_GraphicalObject interface.

The ISch_RectangularGroup interface is an ancestor interface for the ISch_SheetSymbol, ISch_HarnessConnector and IOpenBus_Component interfaces.

Notes

The interface hierarchy for the ISch_RectangularGroup interface is as follows;

ISch_GraphicalObject

ISch_ParametrizedGroup

ISch_RectangularGroup

ISch_RectangularGroup methods	ISch_RectangularGroup properties
SetState_XSize	XSize
SetState_YSize	YSize
GetState_XSize	
GetState_YSize	

See also

ISch_GraphicalObject interface

ISch_ParametrizedGroup interface

IOpenBus_Component interface

ISch_HarnessConnector interface

ISch_SheetSymbol interface

Methods

SetState_YSize method

(ISch_RectangularGroup interface)

Syntax

```
Procedure SetState_YSize(Value : TCoord);
```

Description

This function sets the YSize dimension of the rectangular group object such as the sheet symbol.

Example

```
SheetSymbol.SetState_XSize(MilsToCoord(150));
SheetSymbol.SetState_YSize(MilsToCoord(50));
```

See also

SetState_XSize method

ISch_RectangularGroup interface

SetState_XSize method

(ISch_RectangularGroup interface)

Syntax

```
Procedure SetState_XSize(Value : TCoord);
```

Description

This function sets the XSize dimension of the rectangular group object such as the sheet symbol.

Example

```
SheetSymbol.SetState_XSize(MilsToCoord(150));
SheetSymbol.SetState_YSize(MilsToCoord(50));
```

See also

GetState_YSize method

ISch_RectangularGroup interface

GetState_YSize method

(ISch_RectangularGroup interface)

Syntax

```
Function GetState_YSize : TCoord;
```

Description

This function retrieves the YSize dimension of the rectangular group object such as the sheet symbol. THis function is used by the YSize property.

Example

```
AXSize := SheetSymbol.SetState_XSize;
AYSize := SheetSymbol.SetState_YSize;
```

See also

GetState_YSize method

ISch_RectangularGroup interface

GetState_XSize method

(ISch_RectangularGroup interface)

Syntax

```
Function GetState_XSize : TCoord;
```

Description

This function retrieves the XSize dimension of the rectangular group object such as the sheet symbol. THis function is used by the XSize property.

Example

```
AXSize := SheetSymbol.SetState_XSize;
AYSize := SheetSymbol.SetState_YSize;
```

See also

ISch_RectangularGroup interface

Properties

YSize property

(ISch_RectangularGroup interface)

Syntax

```
Property YSize : TCoord Read GetState_YSize Write SetState_YSize;
```

Description

Example

```
SheetSymbol.SetState_XSize(MilsToCoord(150));
SheetSymbol.SetState_YSize(MilsToCoord(50));
```

See also

ISch_RectangularGroup interface

XSize property

(ISch_RectangularGroup interface)

Syntax

```
Property XSize : TCoord Read GetState_XSize Write SetState_XSize;
```

Description

The XSize property sets or gets the XSize dimension of the rectangular group object such as a sheet symbol. The XSize and YSize values determines the size of the rectangular group object in the X and Y directions.

The Location property from the `ISch_GraphicalObject` interface determines the origin of the rectangular group object.

Example

```
SheetSymbol.XSize := MilsToCoord(150);
SheetSymbol.YSize := MilsToCoord(50);
```

See also

SetState_XSize method

SetState_YSize method

ISch_RectangularGroup interface

ISch_RoundRectangle Interface

Overview

Rounded rectangles are drawing objects which are unfilled or filled graphic elements.

Notes

The ISch_RoundRectangle interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Rectangle

ISch_RoundRectangle

ISch_RoundRectangle methods

SetState_CornerXRadius

SetState_CornerYRadius

GetState_CornerXRadius

GetState_CornerYRadius

ISch_RoundRectangle properties

CornerXRadius

CornerYRadius

Methods

SetState_CornerXRadius method

(ISch_RoundRectangle interface)

Syntax

```
Procedure SetState_CornerXRadius(ADistance : TDistance);
```

Description

Example

See also

ISch_RoundRectangle interface

GetState_CornerYRadius method

(ISch_RoundRectangle interface)

Syntax

```
Function GetState_CornerYRadius : TDistance;
```

Description

Example

See also

ISch_RoundRectangle interface

GetState_CornerXRadius method

(ISch_RoundRectangle interface)

Syntax

```
Function GetState_CornerXRadius : TDistance;
```

Description

Example

See also

ISch_RoundRectangle interface

SetState_CornerYRadius method

(ISch_RoundRectangle interface)

Syntax

```
Procedure SetState_CornerYRadius(ADistance : TDistance);
```

Description

Example

See also

ISch_RoundRectangle interface

See also

ISch_GraphicalObject interface

ISch_Rectangle interface

Properties

CornerYRadius property

(ISch_RoundRectangle interface)

Syntax

```
Property CornerYRadius : TDistance Read GetState_CornerYRadius Write SetState_CornerYRadius;
```

Description

Example

See also

ISch_RoundRectangle interface

CornerXRadius property

(ISch_RoundRectangle interface)

Syntax

```
Property CornerXRadius : TDistance Read GetState_CornerXRadius Write SetState_CornerXRadius;
```

Description

Example

See also

ISch_RoundRectangle interface

ISch_SheetEntry Interface

Overview

A sheet entry within a Sheet Symbol object creates a connection between the net touching on the parent sheet, to a Port with the same name on the child sheet.

Notes

The **ISch_SheetEntry** interface hierarchy is as follows;

ISch_GraphicalObject

    ISch_SheetEntry

ISch_SheetEntry methods

GetState_DistanceFromTop
GetState_IOType
GetState_Name
GetState_OverrideDisplayString
GetState_OwnerSchSheetSymbol
GetState_OwnerSchSheetSymbol
GetState_Side
GetState_Style
GetState_TextColor

SetState_DistanceFromTop
SetState_IOType
SetState_Name

ISch_SheetEntry properties

DistanceFromTop
IOType
Name
OverrideDisplayString
OwnerSheetSymbol
Side
Style
TextColor

SetState_OverrideDisplayString
SetState_Side
SetState_Style
SetState_TextColor

IsVertical

See also

ISch_SheetEntry interface

Methods

SetState_Style method

(ISch_SheetEntry interface)

Syntax

```
Procedure SetState_Style (Value : TPortArrowStyle);
```

Description

The SetState_Style procedure sets the style of the sheet entry. This style is determined by the TPortArrowStyle type. This style defines the graphical style of the sheet entry only if the **I/O Type** property is set to Unspecified. The **IO Type** of the Sheet Entry overrides the **Style** property if the I/O Type is set to one of the specified IO types then changing the Style will not update the graphical content of the sheet entry.

Example

SEntry.SetState_Style(ePortLeft);

See also

TPortArrowStyle type

ISch_SheetEntry interface

SetState_Side method

(ISch_SheetEntry interface)

Syntax

```
Procedure SetState_Side(Value : TLeftRightSide);
```

Description

The SetState_Side procedure sets the orientation of the sheet entry in respect to the associated Sheet symbol.

Example

```
SheetEntry.SetState_Side(eLeftSide);
```

See also

TLeftRightSide type.

ISch_SheetEntry interface.

SetState_Name method

(ISch_SheetEntry interface)

Syntax

```
Procedure SetState_Name(Value : WideString);
```

Description

The SetState_Name procedure sets the new name for the Sheet Entry.

Example

```
SheetEntry.SetState_Name('HarnessType2');
```

See also

ISch_SheetEntry interface

SetState_DistanceFromTop method

(ISch_SheetEntry interface)

Syntax

```
Procedure SetState_DistanceFromTop(Value : TCoord);
```

Description

The SetState_DistanceFromTop function sets the distance from this sheet entry to the top edge of the sheet symbol in a value that's dependent on the grid units. For example if the grid was in DXP Defaults (10 DXP units = 100 mils for example) and the Entry is 10 Units away from the Top part of the Sheet Symbol then you would use the DxpToCoords function to translate the 10 grid units into a coordinate value.

Example

```
SheetEntry.SetState_DistanceFromTop(DxpsToCoord(10));
```

See also

DXPsToCoord function

Measurement Conversion functions

ISch_SheetEntry interface

SetState_TextColor method

(ISch_SheetEntry interface)

Syntax

```
Procedure SetState_TextColor(Value : TColor);
```

Description

The SetState_TextColor procedure sets the color (a value of TColor type) for the Sheet Entry's Name string.

Notes

The TColor value specifies a 6 digit hexadecimal number of the $FFFFFF format. For example the color blue would be RGB:0,0,255 and Hex:FF0000 therefore the converted decimal value would be 16711680. The following formula may be used to calculate the required value, R+256*(G+(256*B)).

Example

```
SheetEntry.SetState_TextColor(0); // sets the text color to black.
```

See also

TColor type

ISch_SheetEntry interface

SetState_IOType method

(ISch_SheetEntry interface)

Syntax

```
Procedure SetState_IOType (Value : TPortIO);
```

Description

The SetState_IOType procedure sets the IO of the sheet entry. This IO Type defines the electrical properties of the sheet entry. Available IOs are: Input, Output, Bi-directional and Unspecified. The IO setting does not influence the connectivity of the circuit, but is considered during the running of an electrical rules check, which can be set to detect incompatible port directions.

Note, the I/O Type of the Sheet Entry overrides the Style property. If the I/O Type is set to Unspecified you can set the appropriate Style for this sheet entry. However if the I/O Type is set to one of the specified I/O types then changing the Style will not update the graphical content of the sheet entry.

Example

```
SheetEntry.SetStateIOType(ePortBidirectional);
```

See also

IPortIO type

ISch_SheetEntry interface

SetState_OverrideDisplayString method

(ISch_SheetEntry interface)

Syntax

```
Procedure SetState_OverrideDisplayString(Value : WideString );
```

Description

The SetState_OverrideDisplayString procedure sets a new value consisting of alph-numeric characters for the Override Display string.

Example

```
SheetEntry.SetState_OverrideDisplayString('New Override String');
```

See also

ISch_HarnessEntry interface

GetState_TextColor method

(ISch_SheetEntry interface)

Syntax

```
Function GetState_TextColor : TColor;
```

Description

The GetState_TextColor function returns the color of the text used for the Name of the Sheet Entry.

Example

```
Color := SheetEntry.GetState_TextColor;
```

See also

TColor type

ISch_SheetEntry

GetState_Style method

(ISch_SheetEntry interface)

Syntax

```
Function GetState_Style : TPortArrowStyle;
```

Description

The GetState_Style function gets the style of the sheet entry. This style is determined by the TPortArrowStyle type. This style defines the graphical style of the sheet entry only if the **I/O Type** property is set to Unspecified. The **IO Type** of the Sheet Entry overrides the **Style** property if the I/O Type is set to one of the specified IO types then changing the Style will not update the graphical content of the sheet entry.

Example

```
Style := SEntry.GetState_Style;
```

See also

TPortArrowStyle type.

ISch_SheetEntry interface

GetState_Side method

(ISch_SheetEntry interface)

Syntax

```
Function GetState_Side : TLeftRightSide;
```

Description

The GetState_Side function returns the orientation of the sheet entry in respect to the associated sheet symbol as a `TLeftRightSide` type.

Example

```
Side := SheetEntry.GetState_Side;
```

See also

TLeftRightSide type

ISch_SheetEntry interface

GetState_SchOwnerSheetSymbol method

(ISch_SheetEntry interface)

Syntax

```
Function GetState_SchOwnerSheetSymbol : ISch_SheetSymbol;
```

Description

The GetState_SchOwnerSheetSymbol function returns the sheet symbol interface (ISch_Sheet Symbol) that this sheet entry is associated with.

Example

```
OwnerSheetSymbol := SheetEntry.GetState_SchOwnerSheetSymbol;
```

See also

ISch_SheetEntry interface

ISch_SheetSymbol interface

GetState_Name method

(ISch_SheetEntry interface)

Syntax

```
Function GetState_Name : WideString;
```

Description

The GetState_Name function returns the name of the sheet entry. Normally the name is a number but can be alphanumeric.

Example

```
EntryName := SheetEntry.GetStateName
```

See also

Name property.

ISch_SheetEntry interface

GetState_IOType method

(ISch_SheetEntry interface)

Syntax

```
Function GetState_IOType : TPortIO;
```

Description

The GetState_IOType procedure gets the IO type of the sheet entry. This IO Type defines the electrical properties of the sheet entry. Available IOs are: Input, Output, Bi-directional and Unspecified. The IO setting does not influence the connectivity of the circuit, but is considered during the running of an electrical rules check, which can be set to detect incompatible port directions.

Note, the I/O Type of the Sheet Entry overrides the Style property. If the I/O Type is set to Unspecified you can set the appropriate Style for this sheet entry. However if the I/O Type is set to one of the specified I/O types then changing the Style will not update the graphical content of the sheet entry.

Example

```
IOType := SheetEntry.GetState_IOType;
```

See also

TPortIO type

ISch_SheetEntry interface

GetState_DistanceFromTop method

(ISch_SheetEntry interface)

Syntax

```
Function  GetState_DistanceFromTop : TCoord;
```

Description

The GetState_DistanceFromTop function returns the distance from this sheet entry to the top edge of the sheet symbol in a

value that's dependent on the grid units. For example if the grid was in DXP Defaults (10 DXP units = 100 mils for example) and the Entry is 10 Units away from the Top part of the Sheet Symbol.

Example

```
Distance := SheetEntry.GetState_DistanceFromTop;
```

See also

ISch_SheetEntry interface

ISch_SheetSymbol interface.

GetState_OverrideDisplayString method

(ISch_SheetEntry interface)

Syntax

```
Function  GetState_OverrideDisplayString : WideString;
```

Description

The GetState_OverrrideDisplayString function returns the override display string which overrides the Name string.

Example

```
DisplayString := SheetEntry.GetState_OverrideDisplayString;
```

See also

ISch_SheetEntry interface

IsVertical method

(ISch_SheetEntry interface)

Syntax

```
Function IsVertical : Boolean;
```

Description

This function returns a Boolean value that determines whether the sheet entry is aligned vertically or not.

Example

```
Vertical := SheetEntry.IsVertical;
```

See also

ISch_SheetEntry interface

Properties

TextColor

(ISch_SheetEntry interface)

Syntax

```
Property TextColor : TColor Read GetState_TextColor Write SetState_TextColor;
```

Description

The TextColor property defines the color (a value of TColor type) for the Harness Entry's Name string. This property is supported by the GetState_TextColor and SetState_TextColor methods.

Notes

The TColor value specifies a 6 digit hexadecimal number of the $FFFFFF format. For example the color blue would be RGB:0,0,255 and Hex:FF0000 therefore the converted decimal value would be 16711680. The following formula may be used to calculate the required value, R+256*(G+(256*B)).

Example

```
SheetEntry.TextColor := 0; // sets the name color to black.
```

See also

TColor type

ISch_SheetEntry interface

Style property

(ISch_SheetEntry interface)

Syntax

```
Property Style : TPortArrowStyle Read GetState_Style Write SetState_Style;
```

Description

The Style property determines the style of the sheet entry and is determined by the TPortArrowStyle type. This style defines the graphical style of the sheet entry only if the **I/O Type** property is set to Unspecified. The **IO Type** of the Sheet Entry overrides the **Style** property if the I/O Type is set to one of the specified IO types then changing the Style will not update the graphical content of the sheet entry.

Example

```
SEntry.Style := ePortLeft;
```

See also

TPortArrowStyle type

ISch_SheetEntry interface

Side

(ISch_SheetEntry interface)

Syntax

```
Property Side : TLeftRightSide Read GetState_Side Write SetState_Side;
```

Description

The Side property defines the orientation of the sheet entry in respect to the associated sheet symbol. This property is supported by the GetState_Side and SetState_Side methods.

Example

```
SheetEntry.Side := eLeftSide;
```

See also

ISch_SheetEntry interface

OwnerSheetSymbol property

(ISch_SheetEntry interface)

Syntax

```
Property OwnerSheetSymbol : ISch_SheetSymbol Read GetState_SchOwnerSheetSymbol;
```

Description

The OwnerSheetSymbol property retrieves the Sheet Symbol interface this Sheet entry is associated with. This property is supported by the GetState_OwnerSheetSymbol method.

Example

```
SheetSymbol := SheetEntry.GetState_OwnerSheetSymbol;
```

See also

ISch_SheetEntry interface

Name

(ISch_SheetEntry interface)

Syntax

```
Property Name : WideString Read GetState_Name Write SetState_Name;
```

Description

The Name property defines the name of the sheet entry. Normally the name property is a number but can be alphanumeric. This property is supported by the GetState_Name and SetState_Name methods.

Example

```
SheetEntry.Name := 'EntryType_2';
```

See also

ISch_SheetEntry interface

IOType property

(ISch_SheetEntry interface)

Syntax

```
Property IOType : TPortIO Read GetState_IOType Write SetState_IOType;
```

Description

The IOType property determines the IO of the sheet entry. This IO Type defines the electrical properties of the sheet entry. Available IOs are: Input, Output, Bi-directional and Unspecified. The IO setting does not influence the connectivity of the circuit, but is considered during the running of an electrical rules check, which can be set to detect incompatible port directions.

Note, the I/O Type of the Sheet Entry overrides the Style property. If the I/O Type is set to Unspecified you can set the appropriate Style for this sheet entry. However if the I/O Type is set to one of the specified I/O types then changing the Style will not update the graphical content of the sheet entry.

Example

```
SheetEntry.IOType := ePortOutput;
```

See also

ISch_SheetEntry interface

DistanceFromTop

(ISch_SheetEntry interface)

Syntax

```
Property DistanceFromTop : TCoord Read GetState_DistanceFromTop Write
SetState_DistanceFromTop;
```

Description

The DistanceFromTop property defines the location of the sheet entry in respect to the associated sheet symbol. This property is supported by the GetState_DistanceFromTop and SetState_DistanceFromTop methods.

Example

```
SheetEntry.DistanceFromTop := DxpsToCoord(10);
```

See also

ISch_SheetEntry interface

OverrideDisplayString property

(ISch_SheetEntry interface)

Syntax

```
Property OverrideDisplayString : WideString Read GetState_OverrideDisplayString Write
SetState_OverrideDisplayString;
```

Description

The OverrideDisplayString property defines the OverRideDisplayString property. This property is supported by the GetState_OverrirdeDisplayString and SetState_OverrirdeDisplayString methods.

Example

```
SheetEntry.OverrideDisplayString('Display String overridden.');
```

See also

ISch_SheetEntry interface

ISch_SheetFileName Interface

Overview

A sheet filename object is part of a complex text object interface and is attached to the sheet symbol object.

Notes

The ISch_SheetFileName interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Label
ISch_ComplexText
ISch_SheetFileName

ISch_SheetFileName methods

ISch_SheetFileName properties

See also

ISch_GraphicalObject interface

ISch_Label interface

ISch_ComplexText interface

ISch_SheetName Interface

Overview

A sheetname is part of a complex text object interface and is associated with a sheet symbol object.

Notes

The ISch_SheetName interface hierarchy is as follows;

ISch_GraphicalObject
ISch_Label
ISch_ComplexText
ISch_SheetName

ISch_SheetName methods

ISch_SheetName properties

See also

ISch_GraphicalObject interface

ISch_Label interface

ISch_ComplexText interface

ISch_SheetSymbol Interface

Overview

Sheet symbols represent other schematic sheets (often referred to as a child sheet). The link between a sheet symbol and other schematic sheets is the FileName attribute, which must be the same as the name of the child sheet.

Notes

The ISch_SheetSymbol interface hierarchy is as follows;

ISch_GraphicalObject
ISch_ParametrizedGroup
ISch_RectangularGroup
ISch_SheetSymbol

ISch_SheetSymbol methods

SetState_UniqueId
SetState_LineWidth
SetState_IsSolid
SetState_ShowHiddenFields
GetState_UniqueId
GetState_LineWidth

ISch_SheetSymbol properties

UniqueId
LineWidth
IsSolid
ShowHiddenFields
SheetFileName
SheetName

GetState_IsSolid

GetState_ShowHiddenFields

GetState_SchSheetFileName

GetState_SchSheetName

See also

ISch_GraphicalObject interface

ISch_ParametrizedGroup interface

ISch_RectangularGroup interface

Methods

SetState_UniqueId method

(ISch_SheetSymbol interface)

Syntax

```
Procedure SetState_UniqueId (Value : WideString);
```

Description

The SetState_UniqueID procedure sets the new ID for the sheet symbol. All parameters, sheet symbols, ports, pins, components, openbus links, openbus ports and openbus components have Unique IDs. Unique IDs are used to maintain design synchronization in design projects.

The Unique ID (UID) is an system generated value that uniquely identifies this current sheet symbol. It is used for linking to a PCB document and for project management. Enter a new UID value or click the **Reset** button to generate a new UID for this design object from the Change Properties dialog. You can also globally reset UIDs of components and sheet symbols from the Schematic Editor's **Tools » Convert » Reset Component Unique IDs** menu.

Example

```
UID := WSM.DM_GenerateUniqueID; // interface and method from Workspace Manager API.
SheetSymbol.SetState_UniqueID(UID);
```

See also

ISch_SheetSymbol interface

SetState_ShowHiddenFields method

(ISch_SheetSymbol interface)

Syntax

```
Procedure SetState_ShowHiddenFields(Value : Boolean);
```

Description

The SetState_ShowHiddenFields procedure determines the visibility of the text fields associated with the sheet symbol, such as its name and filename. If the Value is true, the hidden fields of the sheet symbol will be displayed on the schematic sheet. If the value is False, the hidden text fields are not shown on the schematic.

Example

```
SSheet.SetState_ShowHiddenFields(True); //shows hidden text fields for this sheet symbol.
```

See also

ISch_SheetSymbol interface

SetState_LineWidth method

(ISch_SheetSymbol interface)

Syntax

```
Procedure SetState_LineWidth (Value : TSize);
```

Description

This SetState_LineWidth procedure sets the width of the border line around the sheet symbol. The width is determined by the TSize type.

Example

```
SSheet.SetState_LineWidth(eSmall);
```

See also

TSize type.

ISch_SheetSymbol interface

SetState_IsSolid method

(ISch_SheetSymbol interface)

Syntax

```
Procedure SetState_IsSolid (Value : Boolean);
```

Description

The SetState_IsSolid procedure sets a Boolean value which denotes that the sheet symbol object has a solid internal fill or not.

Example

```
SSymbol.SetState_IsSolid(True);
SSymbol.AreaColor := 0;
```

See also

ISch_SheetSymbol interface

GetState_UniqueId method

(ISch_SheetSymbol interface)

Syntax

```
Function GetState_UniqueId : WideString;
```

Description

The GetState_UniqueID function retrieves the Unique ID for the sheet symbol. All parameters, sheet symbols, ports, pins, components, openbus links, openbus ports and openbus components have Unique IDs. Unique IDs are used to maintain design synchronization in design projects.

The Unique ID (UID) is an system generated value that uniquely identifies this current sheet symbol. It is used for linking to a PCB document and for project management. Enter a new UID value or click the **Reset** button to generate a new UID for this design object from the Change Properties dialog. You can also globally reset UIDs of components and sheet symbols from the Schematic Editor's **Tools » Convert » Reset Component Unique IDs** menu.

Example

UID := SheetSymbol.GetState_UniqueID;

See also

ISch_SheetSymbol interface

GetState_ShowHiddenFields method

(ISch_SheetSymbol interface)

Syntax

```
Function GetState_ShowHiddenFields : Boolean;
```

Description

The GetState_ShowHiddenFields procedure determines the visibility of the text fields associated with the sheet symbol, such as its name and filename. If the Value is true, the hidden fields of the sheet symbol will be displayed on the schematic sheet. If the value is False, the hidden text fields are not shown on the schematic.

Example

```
ShowHiddenFields := SSheet.GetState_ShowHiddenFields;
```

See also

ISch_SheetSymbol interface

GetState_SchSheetName method

(ISch_SheetSymbol interface)

Syntax

```
Function GetState_SchSheetName : ISch_SheetName;
```

Description

The GetState_SchSheetName function returns the ISch_SheetName interface object which represents the Designator object associated with the sheet symbol. The ISch_Sheetname interface is inherited from the ISch_ComplexText and ISch_Label interfaces.

Example

```
SheetName := SSheet.GetState_SchSheetName;
If SheetName <> Nil Then
    Showmessage(SheetName.Text);
```

See also

ISch_SheetName interface;

ISch_SheetSymbol interface

GetState_SchSheetFileName method

(ISch_SheetSymbol interface)

Syntax

```
Function GetState_SchSheetFileName : ISch_SheetFileName;
```

Description

The GetState_SchSheetFileName function returns the ISch_SheetFileName interface object which represents the FileName text object associated with the sheet symbol. The ISch_SheetFileName interface is inherited from the ISch_ComplexText and ISch_Label interfaces.

Example

```
SheetFileName := SSheet.GetState_SchSheetFileName;
If SheetFileName <> Nil Then
    Showmessage(SheetFileName.Text);
```

See also

ISch_SheetFileName interface;

ISch_SheetSymbol interface

GetState_LineWidth method

(ISch_SheetSymbol interface)

Syntax

```
Function GetState_LineWidth : TSize;
```

Description

The GetState_LineWidth function returns the size of the border of the sheet symbol. The Size value is of TSize type.

Example

```
LineWidth := SSheet.GetState_LineWidth;
```

See also

TSize type

ISch_SheetSymbol interface

GetState_IsSolid method

(ISch_SheetSymbol interface)

Syntax

```
Function GetState_IsSolid : Boolean;
```

Description

The GetState_IsSolid function returns a Boolean value whether the sheet symbol object has a solid internal fill or not.

Example

```
If Pie.GetState_IsSolid Then
    Pie. AreaColor := 0; // black fill
```

See also

ISch_SheetSymbol interface

Properties

ShowHiddenFields property

(ISch_SheetSymbol interface)

Syntax

```
Property ShowHiddenFields : Boolean Read GetState_ShowHiddenFields Write
SetState_ShowHiddenFields;
```

Description

The ShowHiddenFields property determines the visibility of the text fields associated with the sheet symbol, such as its name and filename. If the Value is true, the hidden fields of the sheet symbol will be displayed on the schematic sheet. If the value is False, the hidden text fields are not shown on the schematic.

Example

```
SSheet.ShowHiddenFields := True;
```

See also

ISch_SheetSymbol interface

SheetName property

(ISch_SheetSymbol interface)

Syntax

```
Property SheetName : ISch_SheetName Read GetState_SchSheetName;
```

Description

The SchSheetName property denotes the Designator Name text object which is represented by the ISch_SheetName interface object associated with the sheet symbol. The ISch_SheetName interface is inherited from the ISch_ComplexText and ISch_Label interfaces. This property is supported by GetState_SchSheetname method.

Example

```
SheetFileName := SSheet.SchSheetFileName;
If SheetFileName <> Nil Then
    Showmessage(SheetFileName.Text);
```

See also

ISch_SheetSymbol interface

SheetFileName property

(ISch_SheetSymbol interface)

Syntax

```
Property SheetFileName : ISch_SheetFileName Read GetState_SchSheetFileName;
```

Description

The SchSheetFileName property denotes the FileName text object which is represented by the ISch_SheetFileName interface object associated with the sheet symbol. The ISch_SheetFileName interface is inherited from the ISch_ComplexText and ISch_Label interfaces. This property is supported by GetState_SchSheetFileName method.

Example

```
SheetFileName := SSheet.SchSheetFileName;
If SheetFileName <> Nil Then
    Showmessage(SheetFileName.Text);
```

See also

ISch_SheetSymbol interface

LineWidth property

(ISch_SheetSymbol interface)

Syntax

```
Property LineWidth : TSize Read GetState_LineWidth Write SetState_LineWidth;
```

Description

The **LineWidth** property defines the border width of the sheet symbol with one of the following values from the **TSize** enumerated type. This property is supported by the **GetState_LineWidth** and **SetState_LineWidth** methods.

Example

See also

ISch_SheetSymbol interface

TSize type

IsSolid property

(ISch_SheetSymbol interface)

Syntax

```
Property IsSolid : Boolean Read GetState_IsSolid Write SetState_IsSolid;
```

Description

Description

The IsSolid property denotes whether the sheet symbol object has a solid fill or not. This property is supported by the GetState_IsSolid and SetState_IsSolid methods.

Example

```
SScheet.IsSolid := True;
```

See also

ISch_SheetSymbol interface

UniqueId property

(ISch_SheetSymbol interface)

Syntax

```
Property UniqueId : WideString Read GetState_UniqueId Write SetState_UniqueId;
```

Description

The SetState_UniqueID property sets the new ID for the sheet symbol. All parameters, sheet symbols, ports, pins, components, openbus links, openbus ports and openbus components have Unique IDs. Unique IDs are used to maintain design synchronization in design projects.

The Unique ID (UID) is an system generated value that uniquely identifies this current sheet symbol. It is used for linking to a PCB document and for project management. Enter a new UID value or click the **Reset** button to generate a new UID for this design object from the Change Properties dialog. You can also globally reset UIDs of components and sheet symbols from the Schematic Editor's **Tools » Convert » Reset Component Unique IDs** menu.

Example

UID := WSM.DM_GenerateUniqueID; // interface and method from Workspace Manager API.

SheetSymbol.UniqueID(UID);

See also

ISch_SheetSymbol interface

ISch_Symbol Interface

Overview

The symbol objects are special markers used for components in the Schematic Library.

Notes

Descended from ISch_GraphicalObject

ISch_Symbol methods

SetState_Orientation
SetState_Symbol
SetState_IsMirrored
SetState_LineWidth
SetState_ScaleFactor
GetState_Orientation
GetState_Symbol
GetState_IsMirrored
GetState_LineWidth
GetState_ScaleFactor

ISch_Symbol properties

Orientation
Symbol
IsMirrored
LineWidth
ScaleFactor

See also

ISch_GraphicalObject interface

Methods

SetState_Symbol method

(ISch_Symbol interface)

Syntax

```
Procedure SetState_Symbol (AValue : TIeeeSymbol);
```

Description

Example

See also

ISch_Symbol interface

SetState_ScaleFactor method

(ISch_Symbol interface)

Syntax

```
Procedure SetState_ScaleFactor(AValue : TCoord);
```

Description

Example

See also

ISch_Symbol interface

SetState_Orientation method

(ISch_Symbol interface)

Syntax

```
Procedure SetState_Orientation(AValue : TRotationBy90);
```

Description

Example

See also

ISch_Symbol interface

SetState_LineWidth method

(ISch_Symbol interface)

Syntax

```
Procedure SetState_LineWidth (AValue : TSize);
```

Description

Example

See also

ISch_Symbol interface

SetState_IsMirrored method

(ISch_Symbol interface)

Syntax

```
Procedure SetState_IsMirrored (AValue : Boolean);
```

Description

Example

See also

ISch_Symbol interface

GetState_Symbol method

(ISch_Symbol interface)

Syntax

```
Function GetState_Symbol : TIeeeSymbol;
```

Description

Example

See also

ISch_Symbol interface

GetState_ScaleFactor method

(ISch_Symbol interface)

Syntax

```
Function GetState_ScaleFactor : TCoord;
```

Description

Example

See also

ISch_Symbol interface

GetState_Orientation method

(ISch_Symbol interface)

Syntax

```
Function GetState_Orientation : TRotationBy90;
```

Description

Example

See also

ISch_Symbol interface

GetState_LineWidth method

(ISch_Symbol interface)

Syntax

```
Function GetState_LineWidth : TSize;
```

Description

Example

See also

ISch_Symbol interface

GetState_IsMirrored method

(ISch_Symbol interface)

Syntax

```
Function GetState_IsMirrored : Boolean;
```

Description

Example

See also

ISch_Symbol interface

Properties

Symbol property

(ISch_Symbol interface)

Syntax

```
Property Symbol : TIeeeSymbol Read GetState_Symbol Write SetState_Symbol ;
```

Description

Example

See also

ISch_Symbol interface

ScaleFactor property

(ISch_Symbol interface)

Syntax

```
Property ScaleFactor : TCoord Read GetState_ScaleFactor Write SetState_ScaleFactor;
```

Description

Example

See also

ISch_Symbol interface

Orientation property

(ISch_Symbol interface)

Syntax

```
Property Orientation : TRotationBy90 Read GetState_Orientation Write SetState_Orientation;
```

Description

Example

See also

ISch_Symbol interface

LineWidth property

(ISch_Symbol interface)

Syntax

```
Property LineWidth : TSize Read GetState_LineWidth Write SetState_LineWidth ;
```

Description

The **LineWidth** property defines the border width of the circle with one of the following values from the **TSize** enumerated type. This property is supported by the **GetState_LineWidth** and **SetState_LineWidth** methods.

Example

See also

ISch_Symbol interface

TSize type

IsMirrored property

(ISch_Symbol interface)

Syntax

```
Property IsMirrored : Boolean Read GetState_IsMirrored Write SetState_IsMirrored ;
```

Description

Example

See also

ISch_Symbol interface

ISch_Template Interface

Overview

The schematic templates represent the sheet border, title block and graphics for a schematic document.

Notes

The **ISch_Template** interface hierarchy is as follows;

ISch_GraphicalObject

    ISch_Template

ISch_Template methods

SetState_FileName

GetState_FileName

ISch_Template properties

FileName

See also

ISch_GraphicalObject interface

Methods

SetState_FileName method

(ISch_Template interface)

Syntax

```
Procedure SetState_FileName(AValue : WideString);
```

Description

Example

See also

ISch_Template interface

GetState_FileName method

(ISch_Template interface)

Syntax

```
Function GetState_FileName : WideString;
```

Description

Example

See also

ISch_Template interface

Properties

FileName property

(ISch_Template interface)

Syntax

```
Property FileName : WideString Read GetState_FileName Write SetState_FileName;
```

Description

Example

See also

ISch_Template interface

ISch_TextFrame Interface

Overview

Text frames hold multiple lines of free text.

Notes

ISch_TextFrame interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Rectangle

ISch_TextFrame

- The FontID property denotes the font type of the TextFrame object. Windows True Type fonts are fully supported. The FontID value denotes which font has been used. The FontID is the index to an entry in the font table in the Schematic editor. Each font used in the Schematic editor has its own FontID.
- When a new font is used (through a Change Font dialog), a new FontID is added to the internal table in the Schematic editor. The FontID value can be extracted from the following Schematic objects (TextField, Sheet, Annotation, TextFrame and NetLabel objects).

ISch_TextFrame methods	**ISch_TextFrame properties**
SetState_FontId	FontId
SetState_TextColor	TextColor
SetState_Alignment	Alignment
SetState_WordWrap	WordWrap
SetState_ShowBorder	ShowBorder
SetState_ClipToRect	ClipToRect
GetState_FontId	Text
GetState_TextColor	
GetState_Alignment	
GetState_WordWrap	
GetState_ShowBorder	
GetState_ClipToRect	

See also

Methods

SetState_WordWrap method

(ISch_TextFrame interface)

Syntax

```
Procedure SetState_WordWrap (AValue : Boolean);
```

Description

Example

See also

ISch_TextFrame interface

SetState_TextColor method

(ISch_TextFrame interface)

Syntax

```
Procedure SetState_TextColor (AValue : TColor);
```

Description

Example

See also

ISch_TextFrame interface

SetState_ShowBorder method

(ISch_TextFrame interface)

Syntax

```
Procedure SetState_ShowBorder (AValue : Boolean);
```

Description

Example

See also

ISch_TextFrame interface

SetState_FontId method

(ISch_TextFrame interface)

Syntax

```
Procedure SetState_FontId (AValue : Integer);
```

Description

Example

See also

ISch_TextFrame interface

SetState_ClipToRect method

(ISch_TextFrame interface)

Syntax

```
Procedure SetState_ClipToRect (AValue : Boolean);
```

Description

Example

See also

ISch_TextFrame interface

SetState_Alignment method

(ISch_TextFrame interface)

Syntax

```
Procedure SetState_Alignment (AValue : THorizontalAlign);
```

Description

Example

See also

ISch_TextFrame interface

GetState_WordWrap method

(ISch_TextFrame interface)

Syntax

```
Function GetState_WordWrap : Boolean;
```

Description

Example

See also

ISch_TextFrame interface

GetState_TextColor method

(ISch_TextFrame interface)

Syntax

```
Function GetState_TextColor : TColor;
```

Description

Example

See also

ISch_TextFrame interface

GetState_ShowBorder method

(ISch_TextFrame interface)

Syntax

```
Function GetState_ShowBorder : Boolean;
```

Description

Example

See also

ISch_TextFrame interface

GetState_FontId method

(ISch_TextFrame interface)

Syntax

```
Function GetState_FontId : Integer;
```

Description

Example

See also

ISch_TextFrame interface

GetState_ClipToRect method

(ISch_TextFrame interface)

Syntax

```
Function GetState_ClipToRect : Boolean;
```

Description

Example

See also

ISch_TextFrame interface

GetState_Alignment method

(ISch_TextFrame interface)

Syntax

```
Function GetState_Alignment : THorizontalAlign;
```

Description

Example

See also

ISch_TextFrame interface

Properties

FontId property

(ISch_TextFrame interface)

Syntax

```
Property FontId : Integer Read GetState_FontId Write SetState_FontId;
```

Description

Example

See also

ISch_TextFrame interface

WordWrap property

(ISch_TextFrame interface)

Syntax

```
Property WordWrap : Boolean Read GetState_WordWrap Write SetState_WordWrap;
```

Description

Example

See also

ISch_TextFrame interface

TextColor property

(ISch_TextFrame interface)

Syntax

```
Property TextColor : TColor Read GetState_TextColor Write SetState_TextColor;
```

Description

Example

See also

ISch_TextFrame interface

Text property

(ISch_TextFrame interface)

Syntax

```
Property Text : WideString Read GetState_Text Write SetState_Text;
```

Description

Example

See also

ISch_TextFrame interface

ShowBorder property

(ISch_TextFrame interface)

Syntax

```
Property ShowBorder : Boolean Read GetState_ShowBorder Write SetState_ShowBorder;
```

Description

Example

See also

ISch_TextFrame interface

ClipToRect property

(ISch_TextFrame interface)

Syntax

```
Property ClipToRect : Boolean Read GetState_ClipToRect Write SetState_ClipToRect;
```

Description

Example

See also

ISch_TextFrame interface

Alignment property

(ISch_TextFrame interface)

Syntax

```
Property Alignment : THorizontalAlign Read GetState_Alignment Write SetState_Alignment;
```

Description

Example

See also

ISch_TextFrame interface

ISch_Wire Interface

Overview

Wires are straight line segments which are placed on a schematic document to create the electrical connections.

Notes

The `ISch_Wire` interface is descended from the immediate ancestor `ISch_BasicPolyline` interface and the interface hierarchy is as follows;

ISch_GraphicalObject

ISch_Polygon

ISch_BasicPolyline

ISch_Wire

ISch_Wire methods

GetState_CompilationMaskedSegment

SetState_CompilationMaskedSegment

ISch_Wire properties

CompilationMaskedSegment

Fetch the vertices of existing wires example

```
Procedure FetchVertices();
Var
    Index      : Integer;
    Wire       : ISch_Wire;
    Iterator   : ISch_Iterator;
    WireCount  : Integer;
    ALocation  : TLocation;
    SchDoc     : ISch_Document;
    Document   : IServerDocument;
    ReportList : TStringList;
Begin
    If SchServer = Nil Then Exit;
    SchDoc := SchServer.GetCurrentSchDocument;
    If SchDoc = Nil Then Exit;

    // Set up an iterator to look for port objects only.
    Iterator := SchDoc.SchIterator_Create;
    Iterator.AddFilter_ObjectSet(MkSet(eWire));

    WireCount := 0;
    ReportList := TStringList.Create;
    ReportList.Add('Wires'' Vertex report:');
    ReportList.Add('_________________________');
    ReportList.Add('');

    // Using a Try Finally block to avoid exception errors.
    Try
        Wire := Iterator.FirstSchObject;
        While Wire <> Nil Do
        Begin
            Inc(WireCount);
            ReportList.Add('Wire #' + IntToStr(WireCount));
            For Index := 1 To Wire.VerticesCount Do
```

```
            Begin
                ALocation := Wire.Vertex[Index];
                ReportList.Add('X: ' + IntToStr(ALocation.X) + ', Y: ' +
IntToStr(ALocation.Y));
            End;

            ReportList.Add('');
            Wire := Iterator.NextSchObject;
        End;

    Finally
        SchDoc.SchIterator_Destroy(Iterator);
    End;

    ReportList.SaveToFile('C:\WireVertexReport.Txt');
    ReportList.Free;

    // Display the report containing parameters for each component found.
    Document := Client.OpenDocument('Text','C:\WireVertexReport.txt');
    If Document <> Nil Then
        Client.ShowDocument(Document);
End;
```

See also

ISch_GraphicalObject interface

ISch_Polygon interface

ISch_BasicPolyline interface

Methods

GetState_CompilationMaskedSegment method

(ISch_Wire interface)

Syntax

```
Function GetState_CompilationMaskedSegment(AIndex : Integer) : Boolean;
```

Description

Example

See also

ISch_Wire interface

SetState_CompilationMaskedSegment method

(ISch_Wire interface)

Syntax

```
Procedure SetState_CompilationMaskedSegment(AIndex : Integer; AValue : Boolean);
```

Description

Example

See also

ISch_Wire interface

Properties

CompilationMaskedSegment property

(ISch_Wire interface)

Syntax

```
Property CompilationMaskedSegment[AIndex : Integer] : Boolean Read
GetState_CompilationMaskedSegment Write SetState_CompilationMaskedSegment;
```

Description

Example

See also

ISch_Wire interface

Schematic Constants

Internal Unit constants

```
cUnits : Array [TUnit] Of TDynamicString   = ('mil', 'mm', 'in', 'cm', '', 'm',
'AutoImperial', 'AutoMetric');
cUnitSystems : Array[TUnitSystem] Of TUnitSet = ([eMil, eIN, eDXP, eAutoImperial], [eMM,  eCM,
eM, eAutoMetric]);
cAutoUnits = [eAutoImperial, eAutoMetric];
cDefaultUnit             : Array[TUnitSystem] Of TUnit = (eDXP, eMM);
cDefaultGridSettingsUnit : Array[TUnitSystem] Of TUnit = (eMil, eMM);

//1 DXP 2004 SP1 Internal Unit =
// 100000 DXP 2004 SP2 Internal Unit (= 10 mils)
cBaseUnit           = 100000;

//1 mil = 10000 DXP 2004 SP2 internal units
cInternalPrecision  = 10000;

//Size of workspace in DXP 2004 SP1 base logical unit
cMaxWorkspace       = 6500;

//Size of workspace in DXP 2004 SP1 base logical unit
cMinWorkspace       = 10;

//Size of workspace in the new logical unit - max
cMaxWorkspaceSize   = cMaxWorkspace*cBaseUnit;

//Size of workspace in the new logical unit - min
cMinWorkspaceSize   = cMinWorkspace*cBaseUnit;
CMaxTextParamLength = 32000;

cSchInternalTolerance_Metric = 2*cInternalPrecision;

//0 for imperial and 0.004318mm for metric
cSchInternalTolerance : Array[TUnitSystem] Of TCoord = (0, cSchInternalTolerance_Metric);

cSymbolLineWidthArray : Array [TSize] of  Integer  = (0,1*cBaseUnit,3*cBaseUnit,5*cBaseUnit);

cDefaultCustomSizeX_Sheet    : Array[TUnitSystem] Of Integer = (1500*cBaseUnit,   30*c10_0MM);
cDefaultCustomSizeY_Sheet    : Array[TUnitSystem] Of Integer = (950 *cBaseUnit,   20*c10_0MM);
cDefaultCustomSizeX_Library  : Array[TUnitSystem] Of Integer = (2000*cBaseUnit,   40*c10_0MM);
cDefaultCustomSizeY_Library  : Array[TUnitSystem] Of Integer = (2000*cBaseUnit,   40*c10_0MM);
cDefaultCustomMarginWidth    : Array[TUnitSystem] Of Integer = (20  *cBaseUnit,      c5_0MM );

cPolylineCutterBoxHeight      =  3 *cBaseUnit;
cDefaultSheetFileNamePosition       : Array[TUnitSystem] Of Integer = (10 *cBaseUnit, c2_5MM);
```

```
cBusEntryLength                    : Array[TUnitSystem] Of Integer = (10 *cBaseUnit, c2_0MM);
cDefaultPortWidth                  : Array[TUnitSystem] Of Integer = (50 *cBaseUnit,
c10_0MM);
cDefaultSheetSymbolXSize           : Array[TUnitSystem] Of Integer = (80 *cBaseUnit,
5*c7_5MM);
cDefaultSheetSymbolYSize           : Array[TUnitSystem] Of Integer = (50 *cBaseUnit,
5*c5_0MM);
cDefaultSheetEntryGridSize         : Array[TUnitSystem] Of Integer = (10 *cBaseUnit, c2_5MM);
cDefaultPolylineCutterFixedLength  : Array[TUnitSystem] Of Integer = (10 *cBaseUnit, c2_5MM);
cDefaultAutoPanJumpDistance        : Array[TUnitSystem] Of Integer = (30 *cBaseUnit, c7_5MM);
cDefaultAutoPanShiftJumpDistance   : Array[TUnitSystem] Of Integer = (100*cBaseUnit,
c25_0MM);
cDefaultPinLength                  : Array[TUnitSystem] Of Integer = (30 *cBaseUnit,
c0_50MM);
cDefaultCircleRadius               : Array[TUnitSystem] Of Integer = (100*cBaseUnit, c7_5MM);
cDefaultArcRadius                  : Array[TUnitSystem] Of Integer = (10 *cBaseUnit, c5_0MM);
cDefaultStartAngle                 = 30;
cDefaultEndAngle                   = 330;
cDefaultEllipseRadius              : Array[TUnitSystem] Of Integer = (20 * cBaseUnit,
c5_0MM);
cDefaultEllipseSecondaryRadius     : Array[TUnitSystem] Of Integer = (10 * cBaseUnit,
c2_5MM);
cDefaultEllipticalArcSecondaryRadius: Array[TUnitSystem] Of Integer = (10 * cBaseUnit,
c2_5MM);
cDefaultRectangleCornerX           : Array[TUnitSystem] Of Integer = (50 * cBaseUnit,
c5_0MM);
cDefaultRectangleCornerY           : Array[TUnitSystem] Of Integer = (50 * cBaseUnit,
c5_0MM);
cDefaultIEESymbolScale             : Array[TUnitSystem] Of Integer = (10 * cBaseUnit,
c2_5MM);
cDefaultRoundRectCornerXRadius     : Array[TUnitSystem] Of Integer = (20 * cBaseUnit,
c0_50MM);
cDefaultRoundRectCornerYRadius     : Array[TUnitSystem] Of Integer = (20 * cBaseUnit,
c0_50MM);
cDefaultLabelXSize                 : Array[TUnitSystem] Of Integer = (40 * cBaseUnit,
c0_25MM);
cDefaultLabelYSize                 : Array[TUnitSystem] Of Integer = (10 * cBaseUnit,
c0_50MM);
cIEESymbolScale_Min                = 1   * cBaseUnit;
cIEESymbolScale_Max                = 200 * cBaseUnit;
cIEESymbolScale_Step               = 1  * cBaseUnit;

cDuplicateOffsetX                  : Array[TUnitSystem] Of Integer = ( 20 * cBaseUnit,
c5_0MM);
cDuplicateOffsetY                  : Array[TUnitSystem] Of Integer = (-20 * cBaseUnit, -
c5_0MM);

cJumpLocationZoomRectWidth         = 200 * cBaseUnit;
cJumpLocationZoomRectHeight        = 200 * cBaseUnit;
cSheetSymbolBoundingRectInflate    = 20  * cBaseUnit;
```

```
cPinFullBoundingRectInflate         = 5   * cBaseUnit;
cPolylineBoundingRectInflate        = 10  * cBaseUnit;
cFindReplaceRectInflate             = 50  * cBaseUnit;
cPinIEEESymbolRectInflateBy         = 6   * cBaseUnit;
cPortWidthInflate                   = 10  * cBaseUnit;
cMinPortWidth                       = 30  * cBaseUnit;
cMinSheetSymbolBorderGap            : Array[TUnitSystem] Of Integer = (10 * cBaseUnit,
c2_5MM);
```

MM to Internal Units Values

Each Millimetre constant value is expressed in internal units (rounded to nearest integer value).

```
c0_25MM = 98425;
c0_50MM = 196850;
c0_75MM = 295275;
c1_00MM = 393701;
c1_5MM  = 590551;
c2_0MM  = 787402;
c2_5MM  = 984252;
c3_0MM  = 1181102;
c3_5MM  = 1377953;
c4_0MM  = 1574803;
c4_5MM  = 1771654;
c5_0MM  = 1968504;
c5_5MM  = 2165354;
c6_0MM  = 2362205;
c6_5MM  = 2559055;
c7_0MM  = 2755906;
c7_5MM  = 2952756;
c8_0MM  = 3149606;
c8_5MM  = 3346457;
c9_0MM  = 3543307;
c9_5MM  = 3740157;
c10_0MM = 3937008;
c15_0MM = 5905512;
c20_0MM = 7874016;
c25_0MM = 9842520;
c30_0MM = 11811024;
c35_0MM = 13779528;
c40_0MM = 15748031;
c45_0MM = 17716535;
c50_0MM = 19685039;
c55_0MM = 21653543;
c60_0MM = 23622047;
c65_0MM = 25590551;
c70_0MM = 27559055;
c75_0MM = 29527559;
```

```
c80_0MM = 31496063;
c85_0MM = 33464567;
c90_0MM = 35433071;
c95_0MM = 37401575;
c100_0MM  = 39370078;
c1000_0MM = 393700787;
```

Other Constants

cMaxShortStringLength

```
cMaxShortStringLength = 254;
```

cOldSheetEntryGrid

```
cOldSheetEntryGrid = 10;
```

cOldMaxPolygonVertices

```
cOldMaxPolygonVertices = 50;
cCharacterApproximativeWidth  = 8  * cBaseUnit;
cCharacterApproximativeHeight = 10 * cBaseUnit;
cCharacterWidthTolerance      = 4  * cBaseUnit;
cConnectionDrawingThreshold   = 3;

cPinBoundingRectInflate = 2  *cBaseUnit;
cMinWireUnderlineWidth  = 5  *cBaseUnit;
cMinBusUnderlineWidth   = 7  *cBaseUnit;
cCompilationMaskedPopupString = 'Removed by Compilation Mask';
```

LibPrimitiveSet

```
LibPrimitiveSet:  TObjectSet = [eRectangle,
                                eLine,
                                eArc,
                                eBus,
                                eBusEntry,
                                eEllipticalArc,
                                eRoundRectangle,
                                eImage,
                                ePie,
                                eEllipse,
                                ePolygon,
                                ePolyline,
                                ePort,
                                eBezier,
                                eLabel,
                                eNetlabel,
                                eTextFrame,
                                eSymbol,
                                ePin,
                                eParameterSet
                                eWire];
```

```
cObjectInspectorViewname    = 'SchObjectInspector';
cLibObjectInspectorViewname = 'SchLibObjectInspector';

cGroundTypeSet = [ePowerGndPower, ePowerGndSignal, ePowerGndEarth];

CLineShapeArrowRatio = 2;
CLineShapeSizeCoefs : Array[TSize] Of Integer = (1, 2, 3, 4);

cNoUnionIndex = 0;

cStringIncrementStyleStrings : Array[TStringIncrementStyle] Of String = ('None','Horizontal
First', 'Vertical First');
```

cBooleanEditorAttributes

```
cBooleanEditorAttributes =
    [eObjectAttribute_IsHidden,
     eObjectAttribute_Locked,
     eObjectAttribute_Accessible,
     eObjectAttribute_Solid,
     eObjectAttribute_ShowName,
     eObjectAttribute_IsMirrored,
     eObjectAttribute_DesignatorLocked,
     eObjectAttribute_PartIdLocked,
     eObjectAttribute_PinsMoveable,
     eObjectAttribute_ImageKeepAspect,
     eObjectAttribute_ImageEmbed,
     eObjectAttribute_ParameterAllowLibrarySynchronize,
     eObjectAttribute_ParameterAllowDatabaseSynchronize,
     eObjectAttribute_TextAutoPosition,
     eObjectAttribute_PinShowDesignator,
     eObjectAttribute_ShowHiddenFields,
     eObjectAttribute_ShowHiddenPins,
     eObjectAttribute_ShowDesignator,
     eObjectAttribute_TextFrameWordWrap,
     eObjectAttribute_TextFrameShowBorder,
     eObjectAttribute_TextFrameClipToRect,
     eObjectAttribute_PowerObjectShowNetName];
```

cStringEditorAttributes

```
cStringEditorAttributes =
    [eObjectAttribute_LocationX,
     eObjectAttribute_LocationY,
     eObjectAttribute_CornerLocationX,
     eObjectAttribute_CornerLocationY,
     eObjectAttribute_Width,
     eObjectAttribute_Radius,
     eObjectAttribute_StartAngle,
```

```
     eObjectAttribute_EndAngle,
     eObjectAttribute_SecondaryRadius,
     eObjectAttribute_StringText,
     eObjectAttribute_Name,
     eObjectAttribute_Description,
     eObjectAttribute_ParameterValue,
     eObjectAttribute_ParameterName,
     eObjectAttribute_PinWidth,
     eObjectAttribute_PinDefaultValue,
     eObjectAttribute_PinDesignator,
     eObjectAttribute_PinHiddenNetName,
     eObjectAttribute_PinLength,
     eObjectAttribute_RoundRectangleCornerRadiusX,
     eObjectAttribute_RoundRectangleCornerRadiusY,
     eObjectAttribute_SchComponentLibReference,
     eObjectAttribute_SchComponentDesignator,
     eObjectAttribute_SheetEntryDistanceFromTop,
     eObjectAttribute_SymbolScaleFactor,
     eObjectAttribute_TaskHolderInstanceName,
     eObjectAttribute_SheetName,
     eObjectAttribute_OwnerName,
     eObjectAttribute_SchComponentComment,
     eObjectAttribute_SchComponentLibraryName,
     eObjectAttribute_SchComponentFootprint,
     eObjectAttribute_SelectedVertex_X,
     eObjectAttribute_SelectedVertex_Y,
     eObjectAttribute_SelectedVertex2_X,
     eObjectAttribute_SelectedVertex2_Y];
```

cComboBoxEditorAttributes

```
cComboBoxEditorAttributes =
    [eObjectAttribute_OwnerPartId,
     eObjectAttribute_OwnerPartDisplayMode,
     eObjectAttribute_LineStyle,
     eObjectAttribute_StartLineShape,
     eObjectAttribute_EndLineShape,
     eObjectAttribute_LineShapeSize,
     eObjectAttribute_Orientation,
     eObjectAttribute_Alignment,
     eObjectAttribute_BorderWidth,
     eObjectAttribute_LineWidth,
     eObjectAttribute_JunctionSize,
     eObjectAttribute_ParameterType,
     eObjectAttribute_ParameterReadOnlyState,
     eObjectAttribute_PinSwapId_Pin,
     eObjectAttribute_PinSwapId_Part,
     eObjectAttribute_PinSwapId_PartPin,
```

```
     eObjectAttribute_PinFormalType,
     eObjectAttribute_PinElectrical,
     eObjectAttribute_PinIeeeSymbolInner,
     eObjectAttribute_PinIeeeSymbolOuter,
     eObjectAttribute_PinIeeeSymbolInnerEdge,
     eObjectAttribute_PinIeeeSymbolOuterEdge,
     eObjectAttribute_SheetEntrySide,
     eObjectAttribute_PortArrowStyle,
     eObjectAttribute_PortIOType,
     eObjectAttribute_PowerObjectStyle,
     eObjectAttribute_CrossSheetConnectorStyle,
     eObjectAttribute_SchComponentDisplayMode,
     eObjectAttribute_SchComponentPartId,
     eObjectAttribute_SchComponentKind,
     eObjectAttribute_IeeeSymbol];
```

cColorEditorAttributes

```
cColorEditorAttributes =
    [eObjectAttribute_Color,
     eObjectAttribute_TextColor,
     eObjectAttribute_AreaColor];
```

cContextHelpStringsByObjectId

```
cContextHelpStringsByObjectId : Array[TObjectId] Of TDynamicString =
( 'FirstObjectID',
'ClipBoardContainer',
'Note',
'Probe',
'Rectangle',
'Line',
'ConnectionLine',
'BusEntry',
'Arc',
'EllipticalArc',
'RoundRectangle',
'Image',
'Pie',
'TextFrame',
'Ellipse',
'Junction',
'Polygon',
'Polyline',
'Wire',
'Bus',
'Bezier',
'Label',
'NetLabel',
```

```
'Designator',
'SchComponent',
'Parameter',
'ParameterSet',
'ParameterList',
'SheetName',
'SheetFileName',
'Sheet',
'SchLib',
'Symbol',
'NoERC',
'ErrorMarker',
'Pin',
'Port',
'PowerObject',
'SheetEntry',
'SheetSymbol',
'Template',
'TaskHolder',
'MapDefiner',
'ImplementationMap',
'Implementation',
'ImplementationsList',
'CrossSheetConnector',
'CompileMask',
'OpenBusComponent',
'OpenBusLink',
'OpenBusDesignator',
'HarnessConnector',
'HarnessEntry',
'HarnessConnectorType',
'SignalHarness',
'OpenBusPort',
'LastObjectId'
);
```

Power Object constants

```
cPowerObjectLineWidth  = 1 * cBaseUnit;
cPowerGndPowerXOffset1 = 0 * cBaseUnit;
cPowerGndPowerXOffset2 = 3 * cBaseUnit;
cPowerGndPowerXOffset3 = 6 * cBaseUnit;
cPowerGndPowerXOffset4 = 9 * cBaseUnit;
cPowerGndPowerYOffset1 = 10 * cBaseUnit;
cPowerGndPowerYOffset2 = 7  * cBaseUnit;
cPowerGndPowerYOffset3 = 4  * cBaseUnit;
cPowerGndPowerYOffset4 = 1  * cBaseUnit;
cPowerNameXOffset1     = 2  * cBaseUnit;
```

Parameter Set constants

```
cParameterSetLineWidth          = 1   *cBaseUnit;
cParameterSetLineLength         = 6   *cBaseUnit;
cParameterSetCircleRadius       = 6   *cBaseUnit;
cParameterSetCircleCenterOffset = 12  *cBaseUnit;
cParameterSetIOffsetX           = 12  *cBaseUnit;
cParameterSetIOffsetY           = 5   *cBaseUnit;
cParameterSetTextOffetX         = 20  *cBaseUnit;
cParameterSetParamDefaultLength = 5   *cBaseUnit;
cParameterSetParam000XOffset    = 32  *cBaseUnit;
cParameterSetParam090XOffset    = 4   *cBaseUnit;
cParameterSetParam090YOffset    = 24  *cBaseUnit;
cParameterSetParam180XOffset    = 12   *cBaseUnit
cParameterSetParam270XOffset    = 10   *cBaseUnit
cParameterSetParam270YOffset    = 22  *cBaseUnit;
cParameterSetParamYOffset       = 2   *cBaseUnit;
cParameterSetParamDeltaYOffset1 = 12  *cBaseUnit;
```

Title Block constants

```
cTitleBlockWidth               = 350 *cBaseUnit;
cTitleBlockWidth1              = 100 *cBaseUnit;
cTitleBlockWidth2              = 150 *cBaseUnit;
cTitleBlockWidth3              = 300 *cBaseUnit;
cTitleBlockHeight              = 80  *cBaseUnit;
cTitleBlockHeight1             = 50  *cBaseUnit;
cTitleBlockHeight2             = 20  *cBaseUnit;
cTitleBlockHeight3             = 10  *cBaseUnit;
cTitleBlockTextXPos_Title      = 345 *cBaseUnit;
cTitleBlockTextXPos_Number     = 295 *cBaseUnit;
cTitleBlockTextXPos_Revision   = 95  *cBaseUnit;
cTitleBlockTextXPos_Size       = 345 *cBaseUnit;
cTitleBlockTextXPos_SheetStyle = 340 *cBaseUnit;
cTitleBlockTextYPos_SheetStyle = 35  *cBaseUnit;
cTitleBlockTextXPos_Date1      = 345 *cBaseUnit;
cTitleBlockTextXPos_Date2      = 300 *cBaseUnit;
cTitleBlockTextXPos_SheetNbr   = 145 *cBaseUnit;
cTitleBlockTextXPos_File1      = 345 *cBaseUnit;
cTitleBlockTextXPos_File2      = 300 *cBaseUnit;
cTitleBlockTextXPos_DrawnBy    = 145 *cBaseUnit;
cTitleBlockTextYPos_TextLine1  = 20  *cBaseUnit;
cTitleBlockTextYPos_TextLine2  = 10  *cBaseUnit;
cAnsiTitleBlock1               = 175 *cBaseUnit;
cAnsiTitleBlock2               = 625 *cBaseUnit;
cAnsiTitleBlock3               = 425 *cBaseUnit;
cAnsiTitleBlock4               = 125 *cBaseUnit;
cAnsiTitleBlock5               = 63  *cBaseUnit;
```

```
cAnsiTitleBlock6              = 25  *cBaseUnit;
cAnsiTitleBlock7              = 387 *cBaseUnit;
cAnsiTitleBlock8              = 325 *cBaseUnit;
cAnsiTitleBlock9              = 276 *cBaseUnit;
cAnsiTitleBlock10             = 36  *cBaseUnit;
cAnsiTitleBlock11             = 420 *cBaseUnit;
cAnsiTitleBlock12             = 170 *cBaseUnit;
cAnsiTitleBlock13             = 420 *cBaseUnit;
cAnsiTitleBlock14             = 382 *cBaseUnit;
cAnsiTitleBlock15             = 271 *cBaseUnit;
cAnsiTitleBlock16             = 31  *cBaseUnit;
```

Differential Pair constants

```
cDifferentialPairWidth          = 21 * cBaseUnit + cParameterSetLineWidth Div 2;
cDifferentialPairHeight         =  9 * cBaseUnit + cParameterSetLineWidth;
cDifferentialPairShadowSize     = cParameterSetLineWidth;
cDiffPairPosNetNaming           = '_P';
cDiffPairNegNetNaming           = '_N';
cDefaultDiffPairName  : TDynamicString = 'DIFFPAIR';
cDiffPairParam        : TDynamicString = 'DifferentialPair';
cDefaultDiffPair      : TDynamicString = 'DefaultDiffPair'
```

Schematic Enumerated Types

The enumerated types are used for many of the schematic interfaces methods which are covered in this section. For example the `ISch_Port` interface has a ConnectedEnd property which returns a `TPortConnectedEnd` type. You can use this Enumerated Types section to check what the range is for the `TPortConnectedEnd` type.

TAngle

```
TAngle = TReal;
```

TAutoPanStyle

```
TAutoPanStyle = (
    eAutoPanOff,
    eAutoPanFixedJump,
    eAutoPanReCenter
     );
```

TCrossSheetConnectorStyle

```
TCrossSheetConnectorStyle = (
        eCrossSheetLeft,
        eCrossSheetRight
    );
```

TCoordRect

```
TCoordRect                  = Record
        Case Integer of
           0 :(left, bottom, right, top : TCoord);
           1 :(x1,   y1,     x2,    y2  : TCoord);
           2 :(Location1,    Location2  : TLocation);
End;
```

TCoord

TCoord = Integer;

TConnectivityScope

```
TConnectivityScope = (eConnectivity_ConnectionOnly, eConnectivity_WholeNet);
```

TConnectionNodeType

```
TConnectionNodeType = (eConnectionNode_IntraSheetLink, eConnectionNode_InterSheetLink,
eConnectionNode_Hidden);
```

TComponentDisplay

```
TComponentDisplay = (
    eCompBlock,
    eCompDevice,
    eCompPower,
    eCompSymbol
     );
```

TColor

Syntax

```
TColor  = Graphics.TColor;
```

Notes

The **TColor** value specifies a 6 digit hexadecimal number of the $FFFFFF format. For example the color blue would be RGB:0,0,255 and Hex:FF0000 therefore the converted decimal value would be 16711680. The following formula may be used to calculate the required value, R+256*(G+(256*B)).

This TColor value is defined from the Graphics Unit of the Borland Delphi's VCL library.

Examples

Color=0 is black, Color=255 is red, Color=65280 is green Color=16711680 is blue Color=16777215 is white. Decimal or hexadecimal values can be assigned.

See also

ISch_Preferences

IComponentPainterView

ISch_GraphicalObject

ISch_TextFrame

ISch_SheetEntry

ISch_HarnessEntry

ISch_Component

TChosenDocumentScope

```
TChosenDocumentScope = (eScope_None, eScope_SingleDocument, eScope_ProjectDocuments,
eScope_OpenDocuments);
```

TCursorMove

```
TCursorMove = (
    eCursorLeft,
    eCursorRight,
    eCursorTop,
    eCursorBottom);
```

TCursorShape

```
TCursorShape = (
       eLargeCursor90,
       eSmallCursor90,
       eSmallCursor45,
       eTinyCursor45);
```

TDistance

```
TDistance = Integer;
```

TDrawMode

```
TDrawMode = (
    eDrawFull,
    eDrawDraft,
    eDrawHidden);
```

TDrawQuality

```
TDrawQuality = (eFullQuality,eDraftQuality);
```

TDynamicStirng

```
TDynamicString = AnsiString;
```

TIeeeSymbol

```
TIeeeSymbol = (
    eNoSymbol,
    eDot,
    eRightLeftSignalFlow,
    eClock,
    eActiveLowInput,
    eAnalogSignalIn,
    eNotLogicConnection,
    eShiftRight,
    ePostPonedOutput,
    eOpenCollector,
    eHiz,
    eHighCurrent,
    ePulse,
    eSchmitt,
    eDelay,
    eGroupLine,
    eGroupBin,
    eActiveLowOutput,
    ePiSymbol,
    eGreaterEqual,
    eLessEqual,
    eSigma,
    eOpenCollectorPullUp,
    eOpenEmitter,
    eOpenEmitterPullUp,
    eDigitalSignalIn,
    eAnd,
    eInvertor,
    eOr,
    eXor,
    eShiftLeft,
    eInputOutput,
    eOpenCircuitOutput,
    eLeftRightSignalFlow,
    eBidirectionalSignalFlow);
```

TFindReplaceIdentifierScope

```
TFindReplaceIdentifierScope = (
       eFindReplace_AllIdentifiers,
       eFindReplace_NetIdentifiersOnly,
       eFindReplace_DesignatorsOnly);
```

THorizontalAlign

```
   THorizontalAlign = (
       eHorizontalCentreAlign, // eVerticalCentreAlign
```

```
       eLeftAlign,             // eTopAlign
       eRightAlign             // eBottomAlign
        );
```

THitTestResult

```
THitTestResult   = (eHitTest_Fail,
                    eHitTest_NoAction,
                    eHitTest_Move,
                    eHitTest_InPlaceEdit,
                    eHitTest_CopyPaste,
                    eHitTest_Resize_Any,
                    eHitTest_Resize_EndAngle,
                    eHitTest_Resize_StartAngle,
                    eHitTest_Resize_SecondaryRadius,
                    eHitTest_Resize_Radius,
                    eHitTest_Resize_CornerTopLeft,
                    eHitTest_Resize_CornerTopRight,
                    eHitTest_Resize_CornerBottomRight,
                    eHitTest_Resize_CornerBottomLeft,
                    eHitTest_Resize_SideLeft,
                    eHitTest_Resize_SideRight,
                    eHitTest_Resize_SideTop  ,
                    eHitTest_Resize_SideBottom,
                    eHitTest_Resize_Vertical,
                    eHitTest_Resize_Horizontal,
                    eHitTest_Resize_SE_NW,
                    eHitTest_Resize_SW_NE);
```

THitTestMode

```
THitTestMode = (
       eHitTest_AllObjects,
       eHitTest_OnlyAccessible
       );
```

TEditingAction

```
TEditingAction = (eEditAction_DontCare, eEditAction_Move,
eEditAction_Change,eEditAction_Delete,eEditAction_Select);
```

TFontName

```
TFontName = String[lf_FaceSize + 1];
```

TFontID

```
TFontID = Integer;
```

TFileName

```
TFileName = TString;
```

TGridPreset

```
TGridPreset = (eDXPPreset, eCoarse2, eCoarse3, eFine2, eFine3, eElectrical);
```

TIterationDepth

```
TIterationDepth     = (eIterateFirstLevel, eIterateFilteredLevels, eIterateAllLevels);
```

TLeftRightSide

```
TLeftRightSide = (
    eLeftSide,
    eRightSide,
    eTopSide,
    eBottomSide
     );
```

TLibraryAutoZoom

```
TLibraryAutoZoom = (lazNoZoomChange, lazRememberLast, lazCenter);
```

TLibraryScope

```
TLibraryScope = (lsCurrentComponnet, lsAllComponents);
```

TLinePlaceMode

```
TLinePlaceMode  = (eLineAnyAngle,
                   eLine90Start,
                   eLine90End,
                   eLine45Start,
                   eLine45End,
                   eLineArcStart,
                   eLineArcEnd,
                   eAutoWire );
```

TLineShape

```
   TLineShape = (
       eLineShapeNone,
       eLineShapeArrow,
       eLineShapeSolidArrow,
       eLineShapeTail,
       eLineShapeSolidTail,
       eLineShapeCircle,
       eLineShapeSquare
   );
```

TLineStyle

```
   TLineStyle = (
       eLineStyleSolid,
       eLineStyleDashed,
       eLineStyleDotted
        );
```

TLocation

Type

```
TLocation = TPoint;
```

Description

The `TLocation` type is used to define a point in X,Y coordinates for a design object.

Where the TPoint = packed record X: Longint; Y: Longint;end;

See also

ISch_GraphicalObject interface

ISch_Line

ISch_Rectangle

ISch_HarnessConnector

ISch_Polygon

IConnection

IConnectionArray

TMyRect

```
TMyRect = Record
    Left,Right,Top, Bottom, Width, Height : Integer;
End;
```

TOrcadFootprint

```
TOrcadFootPrint = (
    ePartfield1,
    ePartfield2,
    ePartfield3,
    ePartfield4,
    ePartfield5,
    ePartfield6,
    ePartfield7,
    ePartfield8,
    eIgnore);
```

TObjectAttribute

```
TObjectAttribute = (eObjectAttribute_ObjectId,
                    eObjectAttribute_DocumentName,
                    eObjectAttribute_Color,
                    eObjectAttribute_TextColor,
                    eObjectAttribute_AreaColor,
                    eObjectAttribute_LocationX,
                    eObjectAttribute_LocationY,
                    eObjectAttribute_CornerLocationX,
                    eObjectAttribute_CornerLocationY,
                    eObjectAttribute_OwnerPartId,
                    eObjectAttribute_OwnerPartDisplayMode,
                    eObjectAttribute_Width,
                    eObjectAttribute_Radius,
                    eObjectAttribute_Solid,
                    eObjectAttribute_Transparent,
                    eObjectAttribute_StartAngle,
                    eObjectAttribute_EndAngle,
                    eObjectAttribute_SecondaryRadius,
                    eObjectAttribute_StringText,
                    eObjectAttribute_LongStringText,
```

```
eObjectAttribute_LineStyle,
eObjectAttribute_StartLineShape,
eObjectAttribute_EndLineShape,
eObjectAttribute_LineShapeSize,
eObjectAttribute_IsHidden,
eObjectAttribute_FontId,
eObjectAttribute_Orientation,
eObjectAttribute_HorizontalJustification,
eObjectAttribute_VerticalJustification,
eObjectAttribute_TextHorizontalAnchor,
eObjectAttribute_TextVerticalAnchor,
eObjectAttribute_Alignment,
eObjectAttribute_BorderWidth,
eObjectAttribute_LineWidth,
eObjectAttribute_JunctionSize,
eObjectAttribute_Locked,
eObjectAttribute_Accessible,
eObjectAttribute_Name,
eObjectAttribute_OwnerName,
eObjectAttribute_Description,
eObjectAttribute_ShowName,
eObjectAttribute_IsMirrored,
eObjectAttribute_DesignatorLocked,
eObjectAttribute_PartIdLocked,
eObjectAttribute_PinsMoveable,
eObjectAttribute_FileName,
eObjectAttribute_TargetFileName,
eObjectAttribute_ImageKeepAspect,
eObjectAttribute_ImageEmbed,
eObjectAttribute_ParametersList,
eObjectAttribute_ParameterValue,
eObjectAttribute_ParameterName,
eObjectAttribute_ParameterType,
eObjectAttribute_ParameterReadOnlyState,
eObjectAttribute_ParameterAllowLibrarySynchronize,
eObjectAttribute_ParameterAllowDatabaseSynchronize,
eObjectAttribute_TextAutoposition,
eObjectAttribute_PinWidth,
eObjectAttribute_PinFormalType,
eObjectAttribute_PinDefaultValue,
eObjectAttribute_PinDesignator,
eObjectAttribute_PinHiddenNetName,
eObjectAttribute_PinShowDesignator,
eObjectAttribute_PinElectrical,
eObjectAttribute_PinLength,
eObjectAttribute_PinIeeeSymbolInner,
```

```
eObjectAttribute_PinIeeeSymbolOuter,
eObjectAttribute_PinIeeeSymbolInnerEdge,
eObjectAttribute_PinIeeeSymbolOuterEdge,
eObjectAttribute_PinSwapId_Pin,
eObjectAttribute_PinSwapId_Part,
eObjectAttribute_PinSwapId_PartPin,
eObjectAttribute_PortArrowStyle,
eObjectAttribute_PortIOType,
eObjectAttribute_PowerObjectStyle,
eObjectAttribute_PowerObjectShowNetName,
eObjectAttribute_CrossSheetConnectorStyle,
eObjectAttribute_RoundRectangleCornerRadiusX,
eObjectAttribute_RoundRectangleCornerRadiusY,
eObjectAttribute_SchComponentLibraryName,
eObjectAttribute_SchComponentLibReference,
eObjectAttribute_SchComponentDesignator,
eObjectAttribute_SchComponentDisplayMode,
eObjectAttribute_SchComponentPartId,
eObjectAttribute_SchComponentComment,
eObjectAttribute_SchComponentFootprint,
eObjectAttribute_SchComponentKind,
eObjectAttribute_ShowHiddenFields,
eObjectAttribute_ShowHiddenPins,
eObjectAttribute_ShowDesignator,
eObjectAttribute_SheetFileName,
eObjectAttribute_SheetName,
eObjectAttribute_SheetEntrySide,
eObjectAttribute_SheetEntryDistanceFromTop,
eObjectAttribute_IeeeSymbol,
eObjectAttribute_SymbolScaleFactor,
eObjectAttribute_TaskHolderProcess,
eObjectAttribute_TaskHolderInstanceName,
eObjectAttribute_TaskHolderConfiguration,
eObjectAttribute_TextFrameWordWrap,
eObjectAttribute_TextFrameShowBorder,
eObjectAttribute_TextFrameClipToRect,
eObjectAttribute_Author,
eObjectAttribute_Collapsed,
eObjectAttribute_ErrorKind,
eObjectAttribute_SelectedVertex_X,
eObjectAttribute_SelectedVertex_Y,
eObjectAttribute_SelectedVertex2_X,
eObjectAttribute_SelectedVertex2_Y,
eObjectAttribute_UnionIndex,
eObjectAttribute_DatabaseTableName,
eObjectAttribute_SchComponentUseLibraryName,
```

```
                    eObjectAttribute_SchComponentUseDBTableName,
                    eObjectAttribute_DesignItemID,
                    eObjectAttribute_OpenBusComponentKind,
                    eobjectAttribute_PrimaryConnectionPosition,
                    eObjectAttribute_HarnessConnectorSide,
                    eObjectAttribute_HarnessType,
                    eObjectAttribute_HideHarnessConnectorType,
                    eObjectAttribute_BusTextStyle,
                    eObjectAttribute_ArrowKind,
                    eObjectAttribute_OpenBusPortType,
                    eObjectAttribute_OpenBusPortLink,
                    eObjectAttribute_OpenBusLinkMasterPort,
                    eObjectAttribute_OpenBusLinkSlavePort
                    );
```

TObjectCreationMode

```
TObjectCreationMode = (eCreate_Default, eCreate_GlobalCopy);
```

TObjectId

```
TObjectId     = (eFirstObjectID,
                 eClipBoardContainer,
                 eNote,
                 eProbe,
                 eRectangle,
                 eLine,
                 eConnectionLine,
                 eBusEntry,
                 eArc,
                 eEllipticalArc,
                 eRoundRectangle,
                 eImage,
                 ePie,
                 eTextFrame,
                 eEllipse,
                 eJunction,
                 ePolygon,
                 ePolyline,
                 eWire,
                 eBus,
                 eBezier,
                 eLabel,
                 eNetLabel,
                 eDesignator,
                 eSchComponent,
                 eParameter,
                 eParameterSet,
                 eParameterList,
```

```
                eSheetName,
                eSheetFileName,
                eSheet,
                eSchLib,
                eSymbol,
                eNoERC,
                eErrorMarker,
                ePin,
                ePort,
                ePowerObject,
                eSheetEntry,
                eSheetSymbol,
                eTemplate,
                eTaskHolder,
                eMapDefiner,
                eImplementationMap,
                eImplementation,
                eImplementationsList,
                eCrossSheetConnector,
                eCompileMask,
                eOpenBusComponent,
                eOpenBusLink,
                eOpenBusDesignator,
                eHarnessConnector,
                eHarnessEntry,
                eHarnessConnectorType,
                eSignalHarness,
                eOpenBusPort,
                eLastObjectId
               );
```

TObjectSet

```
TObjectSet = Set Of TObjectID;
```

TOpenBusPortType

```
TOpenBusPortType      = (obptUnspecified, obptMaster, obptSlave);
```

TOpenBusComponentKind

```
TOpenBusComponentKind = (obckProcessor, obckArbiter, obckInterconnect, obckPeripheral,
obckMemory, obckConnector, obckTerminator);
```

TOpenBusPortKind

```
TOpenBusPortKind      = (obpkPeripheralMaster, obpkPeripheralSlave,
                         obpkArbiterMaster   , obpkArbiterSlave,
                         obpkInterconMaster  , obpkInterconSlave,
                         obpkConnectorMaster , obpkConnectorSlave)
```

TOpenBusInternalPinType

```
TOpenBusInternalPinType = (iptInterrupt, iptReset, iptClock);
```

TParameter_ReadOnlyState

```
TParameter_ReadOnlyState = (
       eReadOnly_None,
       eReadOnly_Name,
       eReadOnly_Value,
       eReadOnly_NameAndValue
       );
```

TParameterType

```
TParameterType = (eParameterType_String,
                  eParameterType_Boolean,
                  eParameterType_Integer,
                  eParameterType_Float);
```

TPinElectrical

```
TPinElectrical = (
       eElectricInput,
       eElectricIO,
       eElectricOutput,
       eElectricOpenCollector,
       eElectricPassive,
       eElectricHiZ,
       eElectricOpenEmitter,
       eElectricPower);
```

TPlacementMode

```
TPlacementMode   = (ePlacementMode_Single, ePlacementMode_Multiple);
```

TPolylineCutterMode

```
TPolylineCutterMode    = (eCutterSnapToSegment, eCutterGridSize, eCutterFixedLength);
```

TPortArrowStyle

```
TPortArrowStyle = (
       ePortNone,
       ePortLeft,
       ePortRight,
       ePortLeftRight,
       ePortNoneVertical,
       ePortTop,
       ePortBottom,
       ePortTopBottom
        );
```

TPortConnectedEnd

```
TPortConnectedEnd = (
    ePortConnectedEnd_None,
    ePortConnectedEnd_Origin,    //connected at port Location
    ePortConnectedEnd_Extremity, //connected at the other end
    ePortConnectedEnd_Both       //connected at both ends
```

```
    );
```

TPortIO

```
TPortIO = (
    ePortUnspecified,
    ePortOutput,
    ePortInput,
    ePortBidirectional
    );
```

TPowerObjectStyle

```
TPowerObjectStyle = (
       ePowerCircle,
       ePowerArrow,
       ePowerBar,
       ePowerWave,
       ePowerGndPower,
       ePowerGndSignal,
       ePowerGndEarth
       );
```

TProbeMethod

```
TProbeMethod = (
    eProbeMethodAllNets,
    eProbeMethodProbedNetsOnly
    );
```

TRotationBy90

```
TRotationBy90 =
    eRotate0,
    eRotate90,
    eRotate180,
    eRotate270
    );
```

TPrintKind

```
TPrintKind = (ePrintKind_FullColor,ePrintKind_GrayScale,ePrintKind_Monochrome);
```

TPlacementResult

```
TPlacementResult = (eSingleObjectPlacementProcessAborted,eWholeObjectPlacementAborted,
eObjectPlacementSuccessfull);
```

TReal

```
TReal = Double;
```

TRectangleStyle

```
TRectangleStyle = (
    eRectangleHollow,
    eRectangleSolid
    );
```

TSchDropAction

```
TSchDropAction = (eDropAction_None,
                  eDropAction_AskOpenOrInsertText,
                  eDropAction_WarnBinaryAsText,
                  eDropAction_OpenInEditor,
                  eDropAction_OpenAsText,
                  eDropAction_Insert);
```

TSelectionState

```
TSelectionState  = (eSelectionState_None,
                    eSelectionState_FirstSelected,
                    eSelectionState_MultiSelected,
                    eSelectionState_VerticesSelected);
```

TSelectionMatch

```
TypeTSelectionMatch = (
    eMatchSelected,
    eMatchedNotSelected,
    eMatchAnySelection
     );
```

TSheetDocumentBorderStyle

```
TSheetDocumentBorderStyle = (
    eSheetStandard,
    eSheetAnsi
     );
```

TSheetOrientation

```
TSheetOrientation = (eLandscape, ePortrait);
```

TSheetStyle

```
TSheetStyle = (
    eSheetA4,
    eSheetA3,
    eSheetA2,
    eSheetA1,
    eSheetA0,
    eSheetA,
    eSheetB,
    eSheetC,
    eSheetD,
    eSheetE,
    eSheetLetter,
    eSheetLegal,
    eSheetTabloid,
    eSheetOrcadA,
    eSheetOrcadB,
    eSheetOrcadC,
    eSheetOrcadD,
```

```
    eSheetOrcadE
     );
```

TShowCutterMarkersMode

```
TShowCutterMarkersMode = (eMarkersNever, eMarkersAlways, eMarkersOnPolyline);
```

TShowCutterBoxMode

```
TShowCutterBoxMode     = (eBoxNever, eBoxAlways, eBoxOnPolyline);
```

TSide

```
TSide = (
    eLeft,
    eBottom,
    eRight,
    eTop
     );
```

TSize

```
TSize = (
    eZeroSize,
    eSmall,
    eMedium,
    eLarge
     );
```

TSignalLayer

```
TSignalLayer = (
    eNoSignalLayer,
    eTopSignalLayer,
    eMidSignalLayer1,
    eMidSignalLayer2,
    eMidSignalLayer3,
    eMidSignalLayer4,
    eMidSignalLayer5,
    eMidSignalLayer6,
    eMidSignalLayer7,
    eMidSignalLayer8,
    eMidSignalLayer9,
    eMidSignalLayer10,
    eMidSignalLayer11,
    eMidSignalLayer12,
    eMidSignalLayer13,
    eMidSignalLayer14,
    eBottomSignalLayer,
    eMultiSignalLayer,
    ePowerLayer1,
    ePowerLayer2,
    ePowerLayer3,
    ePowerLayer4
```

```
    );
```

TStdLogicState

```
TStdLogicState = (eStdLogic_Unitialized,
                  eStdLogic_ForcingUnknown,
                  eStdLogic_Forcing0,
                  eStdLogic_Forcing1,
                  eStdLogic_HiZ,
                  eStdLogic_WeakUnknown,
                  eStdLogic_Weak0,
                  eStdLogic_Weak1,
                  eStdLogic_DontCare);
```

TStringIncrementStyle

```
TStringIncrementStyle = (eSIS_None, eSIS_HorizontalFirst, eSIS_VerticalFirst);
```

TTextHorzAnchor

```
TTextHorzAnchor = (
       eTextHorzAnchor_None,
       eTextHorzAnchor_Both,
       eTextHorzAnchor_Left,
       eTextHorzAnchor_Right
       );
```

TTextJustification

```
TTextJustification = (
    eJustify_BottomLeft,
    eJustify_BottomCenter,
    eJustify_BottomRight,
    eJustify_CenterLeft,
    eJustify_Center,
    eJustify_CenterRight,
    eJustify_TopLeft,
    eJustify_TopCenter,
    eJustify_TopRight
    );
```

TTextVertAnchor

```
TTextVertAnchor = (
       eTextVertAnchor_None,
       eTextVertAnchor_Both,
       eTextVertAnchor_Top,
       eTextVertAnchor_Bottom
       );
```

TUpperLowerCase

```
TUpperLowerCase = (eUpperCase, eLowerCase, eAnyCase);
```

TUnit

```
TUnit = (eMil, eMM, eIN, eCM, eDXP, eM, eAutoImperial, eAutoMetric);
```

TUnitSet

```
TUnitSet = Set Of TUnit;
```

TUnitSystem

```
TUnitSystem = (eImperial, eMetric);
```

TVerticalAlign

```
TVerticalAlign = (
    eVerticalCentreAlign,
    eTopAlign,
    eBottomAlign
     );
```

TVisibleGrid

```
TVisibleGrid = (
    eDotGrid,
    eLineGrid
     );
```

TVHOrientation

```
THVOrientation = (
    eHorizontal,
    eVertical
     );
```

TWidthArray

```
TWidthArray = Array [TSize] of Integer;
```

Schematic Functions

SchServer Interface

```
Function SchServer : ISch_ServerInterface;
```

Description

The SchServer function returns the interface of the loaded Schematic Editor module in Altium Designer. To work with Schematic objects, you need to have access to the `ISch_ServerInterface` interface first. To obtain the current schematic document, invoke the `SchServer.GetCurrentSchDocument` for instance.

Refer to the `ISch_ServerInterface`'s methods and properties for more information.

Example 1

```
    // Grab current schematic document.
    SchDoc := SchServer.GetCurrentSchDocument;
    If SchDoc = Nil Then Exit;

    // Component is a container that has child objects
    // Create component, and its rectangle, pin and parameter objects.
    Component := SchServer.SchObjectFactory (eSchComponent, eCreate_Default);
```

Example 2

```
   Try
       SchServer.ProcessControl.PreProcess(SchDoc, '');

       // Add the parameter to the pin with undo stack also enabled
       Param.Name := 'Added Parameter';
       Param.Text := 'Param added to the pin. Press Undo and this will disappear.  Press undo
twice to remove the component';
       Param.Location := Point(InchesToCoord(3), InchesToCoord(2.4));

       Pin.AddSchObject(Param);
       SchServer.RobotManager.SendMessage(Component.I_ObjectAddress, c_BroadCast,
SCHM_PrimitiveRegistration, Param.I_ObjectAddress);
   Finally
       SchServer.ProcessControl.PostProcess(SchDoc, '');
   End;
```

See also

ISch_ServerInterface interface

General functions

AlignToGridClosest

```
Function AlignToGridClosest  (AValue    : TCoord; AGridSize : TCoord) : TCoord;
```

AlignToGridDecrease

```
Function AlignToGridDecrease (AValue    : TCoord; AGridSize : TCoord) : TCoord;
```

AlignToGridIncrease

```
Function AlignToGridIncrease (AValue    : TCoord;
                              AGridSize : TCoord) : TCoord;
```

GetState_AllImplementations

```
Function GetState_AllImplementations (Const ASchComponent  : ISch_Component)       : TList;
```

GetState_PinsForCurrentMode

```
Function GetState_PinsForCurrentMode (Const ASchComponent  : ISch_Component)       : TList;
```

GetState_AllPins

```
Function GetState_AllPins            (Const ASchComponent  : ISch_Component)       : TList;
```

GetState_AllParameters

```
Function GetState_AllParameters      (Const ASchObject     : ISch_BasicContainer)  : TList;
```

HitTestResultToCursor

```
Function  HitTestResultToCursor(T : THitTestResult): TCursor;
```

GetDefaultSchSheetStyle

```
Function  GetDefaultSchSheetStyle : TSheetStyle;
```

GetWholeAndFractionalPart_DXP2004SP2_To_DXP2004SP1

```
Procedure GetWholeAndFractionalPart_DXP2004SP2_To_DXP2004SP1(ACoord : TCoord; Var AWholePart,
AFractionalPart : Integer);
```

GetCoord_DXP2004SP1_To_DXP2004SP2

```
Function  GetCoord_DXP2004SP1_To_DXP2004SP2(AWholePart, AFractionalPart : Integer) : TCoord;
```

ConvertFileName_99SEToDXP2004

```
Function ConvertFileName_99SEToDXP2004(Const AOriginalName, ADocKind : TDynamicString) :
TDynamicString;
```

GetResolvedSheetFileName

```
Function GetResolvedSheetFileName(Const AOriginalSFN : TDynamicString; Const AProject :
IProject) : TDynamicString;
```

Sch_GetOwnerProject

```
Function Sch_GetOwnerProject(Const AContainer : ISch_BasicContainer) : IProject;
```

Measurement Conversion functions

```
//Imperial functions
Function  CoordToMils        (    C : TCoord) : TReal;
Function  CoordToDxps        (    C : TCoord) : TReal;
Function  CoordToInches      (    C : TCoord) : TReal;
Function  MilsToCoord        (    M : TReal)  : TCoord;
Function  DxpsToCoord        (    M : TReal)  : TCoord;
Function  InchesToCoord      (    M : TReal)  : TCoord;

//Metric functions
Function  CoordToMMs         (    C : TCoord) : TReal;
Function  CoordToCMs         (    C : TCoord) : TReal;
Function  CoordToMs          (    C : TCoord) : TReal;
Function  MMsToCoord         (    M : TReal)  : TCoord;
Function  CMsToCoord         (    M : TReal)  : TCoord;
Function   MsToCoord         (    M : TReal)  : TCoord;
```

```
Function  MetricString(Var S : TDynamicString; DefaultUnits : TUnit) : Boolean;
Function  ImperialString(Var S : TDynamicString; DefaultUnits : TUnit) : Boolean;

Function  CoordUnitToString       (C : TCoord; U : TUnit) : TDynamicString;

Function  CoordUnitToStringWithAccuracy  (ACoord         : TCoord;
                                          AUnit          : TUnit;
                                          ARounding      : Integer;
                                          AFixedDecimals : Integer) : TDynamicString;

Function  ExtractValueAndUnitFromString(AInString : TDynamicString;
                                        ADefaultUnit : TUnit;
                                    Var AValue       : TDynamicString;
                                    Var AUnit        : TUnit) : Boolean;

Function  StringToCoordUnit       (S : TDynamicString; Var C : TCoord; ADefaultUnit : TUnit) :
Boolean;

Function  CoordUnitToString       (C : TCoord; U : TUnit) : TDynamicString;

Function  CoordUnitToStringFixedDecimals (C : TCoord; U : TUnit; AFixedDecimals : Integer) :
TDynamicString;

Function  CoordUnitToStringNoUnit (C : TCoord; U : TUnit) : TDynamicString;
Function  CoordUnitToStringWithAccuracy  (ACoord         : TCoord;
                                          AUnit          : TUnit;
                                          ARounding      : Integer;
                                          AFixedDecimals : Integer) : TDynamicString;

Function  GetDisplayStringFromLocation(ALocation : TLocation; AUnit : TUnit) : TDynamicString;

Function GetCurrentDocumentUnit : TUnit;
Function GetCurrentDocumentUnitSystem : TUnitSystem;
Function GetSchObjectOwnerDocumentUnit(Const AObject : ISch_BasicContainer) : TUnit;
```

Conversion functions

```
Function  GetStateString_ObjectId                (N : TObjectId                ) : TString;
Function  GetStateString_HorizontalAlign         (N : THorizontalAlign         ) : TString;
Function  GetStateString_IeeeSymbol              (N : TIeeeSymbol              ) : TString;
Function  GetStateString_LeftRightSide           (N : TLeftRightSide           ) : TString;
Function  GetStateString_LineStyle               (N : TLineStyle               ) : TString;
Function  GetStateString_PinElectrical           (N : TPinElectrical           ) : TString;
Function  GetStateString_PortArrowStyle          (N : TPortArrowStyle          ) : TString;
Function  GetStateString_PortIO                  (N : TPortIO                  ) : TString;
Function  GetStateString_PowerObjectStyle        (N : TPowerObjectStyle        ) : TString;
Function  GetStateString_CrossSheetConnectorStyle (N : TCrossSheetConnectorStyle ) : TString;
Function  GetStateString_RotationBy90            (N : TRotationBy90            ) : TString;
```

```
Function  GetStateString_Justification           (N : TTextJustification       ) : TString;
Function  GetStateString_HorizontalJustification (N : TTextJustification       ) : TString;
Function  GetStateString_VerticalJustification   (N : TTextJustification       ) : TString;
Function  GetStateString_SheetStyle              (N : TSheetStyle              ) : TString;
Function  GetStateString_Size                    (N : TSize                    ) : TString;
Function  GetStateString_Location                (N : TLocation                ) : TString;
Function  GetStateString_DisplayMode             (N : TDisplayMode             ) : TString;

Function  GetStateString_LineShape    (N : TLineShape) : TString;
Function  GetStateString_ObjectIdPlural(N : TObjectId) : TString;
```

Justification functions

```
Function  IsJustified_Left    (N : TTextJustification) : Boolean;
Function  IsJustified_HCenter (N : TTextJustification) : Boolean;
Function  IsJustified_Right   (N : TTextJustification) : Boolean;
Function  IsJustified_Bottom  (N : TTextJustification) : Boolean;
Function  IsJustified_VCenter (N : TTextJustification) : Boolean;
Function  IsJustified_Top     (N : TTextJustification) : Boolean;

Procedure GetOrdinalValueFromHorizontalJustification(J : TTextJustification;Var I : Integer);
Procedure GetOrdinalValueFromVerticalJustification  (J : TTextJustification;Var I : Integer);
Procedure GetHorizontalJustificationFromOrdinalValue(I : Integer; Var J : TTextJustification);
Procedure GetVerticalJustificationFromOrdinalValue  (I : Integer; Var J : TTextJustification);
```

Revision History

Date	Version No.	Revision
22-Nov-2005	V1.0	New product release
15-Dec-2005	V1.1	Updated for Altium Designer 6
15-Feb-2006	V1.2	Revised for Altium Designer 6
28-Jun-2006	V1.3	Updated for Altium Designer 6.3
26-Mar-2008	V1.4	Updated Page Size to A4 and object interfaces declarations updated.
20-Apr-2008	V1.5	Updated path references.
27-Aug-2008	V1.6	Schematic API updates.
25-Sept-2008	V1.7	ISch_Junction and formatting updates.

Software, hardware, documentation and related materials:

Copyright © 2008 Altium Limited. All Rights Reserved.

The material provided with this notice is subject to various forms of national and international intellectual property protection, including but not limited to copyright protection. You have been granted a non-exclusive license to use such material for the purposes stated in the end-user license agreement governing its use. In no event shall you reverse engineer, decompile, duplicate, distribute, create derivative works from or in any way exploit the material licensed to you except as expressly permitted by the governing agreement. Failure to abide by such restrictions may result in severe civil and criminal penalties, including but not limited to fines and imprisonment. Provided, however, that you are permitted to make one archival copy of said materials for back up purposes only, which archival copy may be accessed and used only in the event that the original copy of the materials is inoperable. Altium, Altium Designer, Board Insight, DXP, Innovation Station, LiveDesign, NanoBoard, NanoTalk, OpenBus, P-CAD, SimCode, Situs, TASKING, and Topological Autorouting and their respective logos are trademarks or registered trademarks of Altium Limited or its subsidiaries. All other registered or unregistered trademarks referenced herein are the property of their respective owners and no trademark rights to the same are claimed. v8.0 31/3/08