Table of Contents

Documentation for OpenSTAAD

Section 1

Introduction

OpenSTAAD is a library of exposed functions allowing engineers access to STAAD.*Pro*'s internal functions and routines as well as its graphical commands. With *OpenSTAAD*, any user can use practically any programming language (including C, C++, VB, VBA, FORTRAN, Java and Delphi) to tap into STAAD's database and seamlessly link input and output data to third-party applications. OpenSTAAD also empowers the STAAD user to create VBA-like macros within the STAAD.Pro environment to perform such tasks as automating repetitive modeling or post-processing tasks or embedding customized design routines. Following an open architecture paradigm, *OpenSTAAD* was built using ATL COM and COM+ standards as specified by *Microsoft, Inc*. This allows *OpenSTAAD* to be used in any environment whether it is in a customized application written in C++ or in a macro application like *Excel* or *AutoCAD*. *OpenSTAAD* can also be used to link STAAD data to Web-based applications using ActiveX, HTML and ASP.

OpenSTAAD allows engineers and other users to link in-house or third-party applications with STAAD.*Pro*. For example, a user might create a spreadsheet in Excel to analyze and design a circular base plate using support reactions from STAAD. With *OpenSTAAD*, a simple macro can be written in Excel or within the STAAD environment to retrieve the appropriate STAAD data and automatically link the results. If the STAAD file changes, so will the *Excel* sheet! With a built-in VBA editor, macros can be written inside STAAD using VBA to create new dialog boxes or menu items which run design codes or specific structural components (like certain connections) that automatically link to STAAD's familiar reporting tables. A cumbersome export/import link between two or three software is no longer required. *OpenSTAAD* is currently being designed to work on the new .NET platform as well as on PocketPCs running *Windows CE*.

Programming Languages

Although *OpenSTAAD* supports all major programming languages used today, it is very difficult to document the usage of each and every function in all of these languages. Most of the example programs or code snippets for each documented *OpenSTAAD* function are written in *VBA for Excel* or *AutoCAD VBA*. The reason is

that with OpenSTAAD 2.0 and higher, STAAD.Pro is equipped with a functional VBA editor capable of writing macros only in VBA. In future revisions of this *OpenSTAAD* documentation, examples in other languages such as or C++ will be added, and use of *OpenSTAAD* in other languages or environments like the Web will be highlighted.

REI will not provide direct support on how to write VBA macros or syntax on other programming languages. There are, however, several useful and free sites on the Web to assist a beginner on writing macros in STAAD, Excel, AutoCAD or any other VBA compliant software. It should be noted that the VBA language is the same from software to software. However, the functions, objects and core libraries will obviously vary.

The sites recommended by REI are as follows:

http://www.excel-vba.com

http://www.beyondtechnology.com/vba.shtml

http://www.afralisp.com

http://www.wrox.com

Also, it is worth noting that Excel has a nice *Record Macro* feature. You can run the recorder and then select any commands from Excel's menus and toolbars. The corresponding VBA syntax will automatically be generated. Look under 'Recording Macros' in the Excel help for additional information.

OpenSTAAD User Forum and Code Resource

Research Engineers hosts an *OpenSTAAD* Macro Exchange on the company's web site at www.reiworld.com. The Macro Exchange allows you to upload and download code in any programming language. You may also join the OpenSTAAD Beta Group if you are interested in previewing new OpenSTAAD functions and providing your feedback during the development process.

Code uploaded to the Macro Exchange web page will not be posted immediately, but will be subjected to a delay of several hours to accommodate a screening process. Code submitted to the site will be tested to try to ensure that it runs without crashing and is not intended to do anything malicious. Research Engineers will not accept responsibility for any mistake, error or misrepresentation in or as a result of the usage of software code obtained from the OpenSTAAD Macro Exchange.

Application Program Interface (API)

The *OpenSTAAD* library of functions is classified under the following general categories:

- STAAD File Input and Output (I/O)
- Structure Geometry
- Member Specifications
- Properties
- Loads
- Output Results
    - Nodes
    - Beams
    - Plates
    - Solids
- STAAD Pre-Processor
- STAAD Post-Processor
- Creating Dialog Boxes and Menu Items

Instantiating OpenSTAAD Library for Use

The first thing necessary to access STAAD project data from within another application is to *instantiate*, or create an instance of *OpenSTAAD* within the other application. In Visual Basic for Applications (VBA), this may be done by creating an object variable and then assigning to it the OpenSTAAD object. The VBA `CreateObject` function may be used for this. In OpenSTAAD, there are two fundamental objects which control distinct parts of STAAD.Pro. These objects are the actual STAAD.Pro application/environment and access to STAAD.Pro results.

The object which controls the STAAD.Pro environment is referred to as `StaadPro.OpenSTAAD`. This object must be created in order to get access to any of the internal graphical functions within STAAD.Pro (including the creating of menu items and dialog boxes). The following VBA function can be used to instantiate or create this object:

```
Set objSTAADGUI = CreateObject("StaadPro.OpenSTAAD")
```

The object `objSTAADGUI` can be used to drill down into STAAD's viewing, geometry modeling, results grid and post-processing functions.

The object which enables access to the results database in STAAD.Pro is referred to as `OpenSTAAD.Output.1`. This object must be created in order to get access to any of the results output created by STAAD.Pro for a particular model. The following VBA function can be used to instantiate or create this object:

```
Set objSTAADResults = CreateObject("OpenSTAAD.Output.1")
```

If you want to work with more than one STAAD file at a time, you can create more than one instance of an OpenSTAAD object within your application.

At the conclusion of your OpenSTAAD application, the OpenSTAAD object(s) should be terminated, to unlock the handles within the application to the OpenSTAAD functions, and to free system resources.

Example

```
Sub How2Begin

'Create an Output variable to hold your OpenSTAAD object(s).

  Dim objSTAADResults
  Dim objSTAADResults2
  Dim objSTAADGUI

'Launch OpenSTAAD Objects.

  Set objSTAADResults = CreateObject("OpenSTAAD.Output.1")

'If we want to work with more than one STAAD file at a time or with the GUI and
'results objects concurrently, we can instantiate a multiple instances of an
'OpenSTAAD object:

  Set objSTAADResults2 = CreateObject("OpenSTAAD.Output.1")
  Set objSTAADGUI = CreateObject("StaadPro.OpenSTAAD ")

'At the end of your application, remember to terminate the OpenSTAAD objects.

  Set objOpenSTAAD = Nothing
  Set objOpenSTAAD2 = Nothing
  Set objSTAADGUI = Nothing

End Sub
```

Function Return Value

If the function return value for an OpenSTAAD function is equal to 0, it means that OpenSTAAD was unable to execute the function. Check to be sure that you have passed all required parameters to the function. Make sure that all parameters being passed are valid.

A return value of 1 indicates that OpenSTAAD successfully executed the function.

Unless specified otherwise, results returned by a function are stored in variable names passed to it for the purpose. A few of the OpenSTAAD Application functions return the results as the return value of the function. In those cases, the above comments regarding the function return value do not apply.

All values are given in units of kips and inches unless specified otherwise.

STAAD Nomenclature

In STAAD documentation and in the program menus and dialog boxes, the terms "member" and "beam" are used interchangeably. Use of the term "beam" should not be taken to imply that the member cannot take an axial load.

Similarly, the terms "joint" and "node" are used interchangeably in STAAD. Both terms refer to the connections between elements.

Connections are also referred to as incidences. The terms "member incidences," "plate incidences," and "solid incidences" refer to the nodes that connect these elements to other elements and supports.

OpenSTAAD Compatibility with STAAD.*Pro*

For optimum performance, *OpenSTAAD* should be run in STAAD.*Pro* 2002 or higher. The object which controls the STAAD.Pro graphical interface (`StaadPro.OpenSTAAD` -dialogs, modeling tools, etc.) is only available in STAAD.Pro 2003 or higher.

As software development proceeds, it is sometimes necessary to modify STAAD's database to accommodate new features or to maximize efficiency and increase processing speed. STAAD.*Pro* 2002 is the first STAAD release to include *OpenSTAAD*. All subsequent releases of *OpenSTAAD* will be backward-compatible with STAAD.*Pro* 2002.

It may be possible to run at least some of *OpenSTAAD*'s functions in earlier releases of STAAD.*Pro* or STAAD-III, but the results may be unpredictable, since *OpenSTAAD* did not exist at that time, and therefore compatibility with *OpenSTAAD* was not considered during the development of these earlier STAAD versions. Research Engineers cannot provide any technical support for *OpenSTAAD* running in STAAD versions prior to STAAD.*Pro* 2002.

In terms of STAAD's input file, STAAD.*Pro* is backward-compatible with all previous STAAD releases. To use *OpenSTAAD* on a project that was created using an earlier version of STAAD.*Pro* or STAAD-III, you can open your old input file in STAAD.*Pro* 2002 and run the analysis. This action will create a new database in a format fully compatible with *OpenSTAAD*.

OpenSTAAD is compatible with any version of Microsoft Excel that supports VBA macros. *OpenSTAAD* is also compatible with AutoCAD 2000 (or higher) VBA.

With OpenSTAAD 2.0 and higher (available with STAAD.Pro 2003 and higher), a VBA editor is embedded within the STAAD.Pro environment. The version of VBA which the editor supports is not 100% VBA compatible. There are a few functions which can be supported in the Microsoft version of VBA which are not supported in the VBA editor/compiler which comes with STAAD.Pro. Every attempt is being made to support these functions. Currently, the functions which are not supported are not yet documented in this manual.

OpenSTAAD Manual Organization

The OpenSTAAD manual will document all functions by the two objects it currently supports:

1) `OpenSTAAD.Output.1` – Access to the output

2) `StaadPro.OpenSTAAD` – Access to the STAAD.Pro application

The first object is outlined in Section 2 of this manual while the latter is outlined in Section 3. The two objects are not dependent on one another. However, the latter is only available in STAAD.Pro 2003 and higher.

Visual Basic Conventions

Comments

In VB or VBA, an apostrophe (‘) is used to denote a comment. Anything to the right of the apostrophe will be ignored by the program.

Declaring Arrays

VB/VBA is flexible in the way it allows arrays to be declared. Most examples involving arrays in this reference manual will conform to the C++ zero indexing convention. In an array of 6 values, the positions in the array are referred to as 0-5. Therefore an array of 6 values would be declared as follows:

```
Dim pdArray(5) As Double
```

or

```
Dim pdArray(0 To 5) As Double
```

In VB, we could also declare a 6-value array as:

```
Dim pdArray(1 To 6) As Double
```

In doing so, however, we might find that our loops and other statements used to access the various positions in the array might not work correctly in C++.

Line Continuation Character

A long coding statement can be written on more than one line, to make the code easier to read. The VB line continuation character consists of a space followed by an underscore at the end of the line. The line of code beneath the continuation character will be handled as though it was written on the same line as the line continuation character.

Example

```
objOpenSTAAD.GetNodeDisplacements  nNodeNo, nLC, pdDisps(0)

‘The line of code above may also be written as the line shown below:

objOpenSTAAD.GetNodeDisplacements _
                    nNodeNo, nLC, pdDisps(0)
```

Function List Organization

The list of functions is formatted in the following way. The name of each function appears in large bold font at the top of the page. The Visual Basic (VB) syntax for

the function appears next. All function syntax definitions consist of the function return value data type, the function name and then the parameters enclosed in parentheses, e.g.,

```
returnvaluedatatype functionname (param1, param2, …, paramn)
```

When actually writing the function, the parameters are not enclosed in parentheses, but they are separated/delimited by commas. The function and all its parameters are usually written on a single line, but for long lines of code, a line continuation character may be used to make the code easier to read. In VB, the line continuation character is a space followed by an underscore (_).

Following the function syntax definition is a description of each parameter required by the function. Note that the names of the parameters are just example names. If the parameter is a variable name for storing results returned by the function, or if you are passing a parameter required by the function as a variable, you can give it any legal variable name.

Next comes a remarks section which describes the purpose of the function, and provides general comments, constraints and recommendations for using it. Then an example of the function is provided. In some cases, the example is just a short code snippet; in other cases a fully viable macro or program is given. All code provided will be able to work within the STAAD macro editor or an external VBA editor (like Excel).

Finally, a list of related functions will be shown under the heading, “See Also.”

OpenSTAAD Functions – Results Object

Section 2

STAAD File I/O Functions

SelectSTAADFile

VB Syntax

integer SelectSTAADFile (string STAADFile)

Parameters

STAADFile

A string providing the file name and path of the STAAD file to be accessed by subsequent OpenSTAAD functions. This string must be enclosed in quotes.

Remarks

The *SelectSTAADFile* function is used to specify the STAAD output file to be used by succeeding functions. This function must precede any functions that act upon or retrieve information from the STAAD output file. It will normally be one of the first functions you will use in your application. You can open more than one STAAD file at a time by creating a second instance of the *OpenSTAAD* library from within your application.

It is highly recommended that before you conclude your application, you close any STAAD files you opened with this command. The CloseSTAADFile function is provided for this purpose.

Example

```
Dim objOpenSTAAD As Output

'Run an instance of OpenSTAAD

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")

'Open a STAAD file
```

```
objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp08.std"
```

See also

CloseSTAADFile
AreResultsAvailable

CloseSTAADFile

VB Syntax

integer CloseSTAADFile ()

Parameters

(none)

Remarks

This function closes a STAAD file that was opened with the SelectSTAADFile function. This function should always be used to close the STAAD file before exiting the macro, in order to free system resources and avoid conflicts with other processes.

Example

```
objOpenSTAAD.CloseSTAADFile
```

See also

SelectSTAADFile
AreResultsAvailable

AreResultsAvailable

VB Syntax

integer AreResultsAvailable (string lpszFileName, integer pnResult)

Parameters

lpszFileName

A string passed to the function specifying the filename and path of an existing STAAD input file.

pnResult

An integer variable name passed to the function for the function to use in storing the results it retrieves. pnResult = 0 if no; pnResult = 1 if yes.

Remarks

This function checks whether the analysis has been run and an analyis results file (*.ANL file) is available for an existing STAAD input file. The filename and path of an existing STAAD input file is passed to the function as the first parameter, lpszFileName. A second parameter, pnResult, specifying an integer variable name for storing the results is also passed to the function. The function then determines whether a corresponding analysis results file exists in the same directory as the specified STAAD input file. If the function determines that the results file exists, it stores a value of 1 in the pnResult integer variable. If the function determines that the results file does not exist, it stores a value of 0 in the pnResult integer variable.

Example

```
Sub Test4Results()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare an integer variable for storing the function output.

  Dim pnResult as Integer

'Run an instance of OpenSTAAD.

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")

'Determine whether examp01.std file exists in same directory as examp01.std
'and store the results in the pnResult integer variable.
'Note the use of the VB line continuation character, a space followed by an
```

```
' underscore in the following code, allowing a single code statement to
' be written on multiple lines.

  objOpenSTAAD.AreResultsAvailable _
            "C:\SPRO2004\STAAD\Examp\US\examp01.std", pnResult

'Close the STAAD file and release the handles to the OpenSTAAD library.

  ObjOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

SelectSTAADFile
CloseSTAADfile

AnalyzeStructure

VB Syntax

AnalyzeStructure (string lpszDesignCode)

Parameters

lpszDesignCode

A string passed to the function specifying the country's steel and/or concrete design code(s) to be used in the current analysis run. The possible values are:

1) "AUSTRALIAN"
2) "BRITISH (5400-8007)"
3) "BRITISH (5950.1990-8110)"
4) "BRITISH (5950.2000-8110)"
5) "CANADIAN"
6) "CHINA"
7) "DANISH"
8) "DUTCH"
9) "EUROCODE"
10) "FINNISH"
11) "FRENCH"
12) "GERMAN"
13) "INDIAN"
14) "JAPANESE"
15) "NORWAY"
19) "RUSSIAN"
20) "SINGAPORE"
21) "SPANISH"
22) "SWEDISH"
23) "US"

Remarks

This function will run the the STAAD file loaded using the function SelectSTAADFile. The actual STAAD engine will instantiate and produce a valid set of results in the same directory as the STAAD file. If there are errors in the input file, STAAD will generate an ANL file (with the same name) listing all the various errors.

Example

```
Sub RunEngine()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare an integer variable for storing the function output.

  Dim pnResult as Integer

'Run an instance of OpenSTAAD.
```

```
  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")

'Determine whether the results of examp01.std file exists in same directory as
examp01.std and store the results in the pnResult integer variable.
'Note the use of the VB line continuation character, a space followed by an
' underscore in the following code, allowing a single code statement to
' be written on multiple lines.

  objOpenSTAAD.AreResultsAvailable _
            "C:\SPRO2004\STAAD\Examp\US\examp01.std", pnResult

if (pnResult == 0) then
         'No results, so run the STAAD file
        objOpenSTAAD.AnalyzeStructure "US"
end if

'Close the STAAD file and release the handles to the OpenSTAAD library.

  ObjOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

SelectSTAADFile
CloseSTAADfile

Structure Geometry Functions

GetNodeCoordinates

VB Syntax

integer GetNodeCoordinates (integer nNode, double pX, double pY, double pZ)

Parameters

nNode

An integer variable corresponding to the number of the node for which the coordinates are to be retrieved by the function.

pX

A double (8-byte floating-point) variable that stores the x coordinates of the supported nodes.

pY

A double (8-byte floating-point) variable that stores the y coordinates of the supported nodes.

pZ

A double (8-byte floating-point) variable that stores the z coordinates of the supported nodes.

Remarks

This function retrieves the xyz coordinates for a given node in the structure.

Note that the coordinates are reported in units of inches.

Example

```
'This code snippet retrieves the coordinates for Node 6

 Dim pX as Double
 Dim pY as Double
 Dim pZ as Double

  objOpenSTAAD.GetNodeCoordinates 6, pX, pY, pZ
```

See Also

GetNodesCount
GetAllNodesCoordinates
GetNextNodeCoordinate
DoesNodeHaveSupport
GetNumberOfSupportedNodes
GetAllNodesThatAreSupported
RenumberNodes

GetNodesCount

VB Syntax

integer GetNodesCount (long pnNodes)

Parameters

pnNodes

A long variable for storing the number of nodes retrieved by the function.

Remarks

This function retrieves the number of nodes in the currently open STAAD file.

Example

```
Dim pnNodes As Long
 objOpenSTAAD.GetNodesCount pnNodes
```

See Also

GetNodeCoordinates
GetAllNodesCoordinates
GetNextNodeCoordinate
DoesNodeHaveSupport
GetNumberOfSupportedNodes
GetAllNodesThatAreSupported
RenumberNodes

GetAllNodesCoordinates

VB Syntax

integer GetAllNodesCoordinates (long pnNodeNumbers, double pX, double pY, double pZ)

Parameters

pnNodeNumbers

A dynamic long array variable that stores the node numbers.

pX

A dynamic double (8-byte floating-point) array variable that stores the x coordinates of the nodes.

pY

A dynamic double (8-byte floating-point) array variable that stores the y coordinates of the nodes.

pZ

A dynamic double (8-byte floating-point) array variable that stores the z coordinates of the nodes.

Remarks

This function retrieves the node number and xyz coordinates for all nodes in the currently open STAAD file. The node numbers and coordinates are stored in dynamic arrays, since the size of the array will vary, depending on the number of nodes in the structure. The OpenSTAAD function GetNodesCount can be used to retrieve the number of nodes in the structure. Then, the VB 'ReDim' function can be used to size the arrays.

Note that the coordinates are reported in units of inches.

Example

```
'This is a VB for Excel macro.

Sub AllNodes()

  Dim objOpenSTAAD As Output
  Dim pnNodes As Long

'The next 4 are dynamic array variables, use empty parentheses
' right now.  We will use GetNodesCount to find the size of the
' array, then we will use a VB ReDim function to size the array.

  Dim pnNodeNumbers() As Long
  Dim pX() As Double
  Dim pY() As Double
  Dim pZ() As Double

‘Create an instance of OpenSTAAD and open a STAAD file.

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp08.std"

'Find the number of nodes in the structure

  objOpenSTAAD.GetNodesCount pnNodes

'Now that we know how many nodes there are, we can
'REDIMension the arrays.

  ReDim pnNodeNumbers(0 To pnNodes)
  ReDim pX(0 To pnNodes), pY(0 To pnNodes), pZ(0 To pnNodes)

'Recall that when passing array names in VB, you also need to
' specify the starting point of the arrays (in parentheses).

  objOpenSTAAD.GetAllNodesCoordinates pnNodeNumbers(0), pX(0), pY(0), pZ(0)

‘Now write the results in columns 7-10 of the worksheet,
‘starting on Row 20
‘To specify the position in the arrays to write the results, use i-1.

  For i = 1 To pnNodes
    Cells(19 + i, 7).Value = pnNodeNumbers(i - 1)
    Cells(19 + i, 8).Value = pX(i - 1)
    Cells(19 + i, 9).Value = pY(i - 1)
    Cells(19 + i, 10).Value = pZ(i - 1)
  Next

'Before ending, close the STAAD file and release the link to OpenSTAAD.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetNodeCoordinates
GetNodesCount
GetNextNodeCoordinate
DoesNodeHaveSupport
GetNumberOfSupportedNodes
GetAllNodesThatAreSupported

RenumberNodes

GetNextNodeCoordinate

VB Syntax

integer GetNextNodeCoordinate (long nPreviousNodeNo, long pnNextNodeNo, double pX, double pY, double pZ)

Parameters

nPreviousNodeNo

A long variable specifying the node number which is the node prior (i.e., the antecedent, in ascending numerical order) to the node for which the number and coordinates are to be obtained. If this variable is set to a value of 0, the function will return the node number and coordinates for the node with the lowest valid node number.

pnNextNodeNo

A long variable in which the node number retrieved by the function will be stored.

pX

A double (8-byte floating-point) variable that stores the x coordinate corresponding to pnNextNodeNo.

pY

A double (8-byte floating-point) variable that stores the y coordinate corresponding to pnNextNodeNo.

pZ

A double (8-byte floating-point) variable that stores the z coordinate corresponding to pnNextNodeNo.

Remarks

This function retrieves the node number and xyz coordinates for the next valid node number in ascending numerical order from a given node. This function may be useful where it is desired to obtain the node numbers and/or coordinates for a limited range of nodes. It may also be helpful when it is desired to know the next node

number in a STAAD model where not all integers within a given range have been assigned to a node.

Note that coordinates are expressed in units of inches.

Example

```
Sub NextNode()

'This macro allows us to enter a starting node number in the worksheet.
'We also enter the number of nodes above the starting node we want the function
'to retrieve.  The node numbers and coordinates are then displayed in the
worksheet.

 Dim objOpenSTAAD As Output
 Dim pnPreviousNodeNo As Long
 Dim pnNextNodeNo As Long
 Dim pX As Double
 Dim pY As Double
 Dim pZ As Double

'Run an instance of OpenSTAAD

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")

'Open STAAD's Example 8 (US)

  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp08.std"

'Get the starting node number from Row 29, Column 5 of the Excel worksheet

  pnPreviousNodeNo = Cells(29, 5).Value

'Get from Row 29, Column 2 the number of succeeding nodes you want OpenSTAAD
' to retrieve.

  For i = 1 To Cells(29, 2).Value

    'Now execute the function.
    'Note the use of the VB line continuation character, a space followed by an
    ' underscore in the following code, allowing a single code statement to
    ' be written on multiple lines.


      objOpenSTAAD.GetNextNodeCoordinate _
                                pnPreviousNodeNo, pnNextNodeNo, pX, pY, pZ

    'Write the results starting in Row 31, Node No. in Column 1, X coord.
    ' in Col. 2, etc.

      Cells(30 + i, 1).Value = pnNextNodeNo
      Cells(30 + i, 2).Value = pX
      Cells(30 + i, 3).Value = pY
      Cells(30 + i, 4).Value = pZ

    'Now, to prepare to reiterate, increment the value of the previous
    ' node number to the node number you just wrote in the worksheet.

      pnPreviousNodeNo = pnNextNodeNo

    'Do it again until you've done it the number of times you specified in the
    ' worksheet cell, Row 29 Column 2.

  Next i

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing
```

```
End Sub
```

See Also

GetNodeCoordinates
GetNodesCount
GetNodesCoordinates
DoesNodeHaveSupport
GetNumberOfSupportedNodes
GetAllNodesThatAreSupported
RenumberNodes

DoesNodeHaveSupport

VB Syntax

integer DoesNodeHaveSupport (integer nNode integer pnSupported)

Parameters

nNode

An integer corresponding to the number of the node to be checked to see whether a support for the structure exists at that node.

pnSupported

An integer variable in which the results of the DoesNodeHaveSupport function will be stored. The function result will be either 0 (No) or 1 (Yes). If no support exists at the node, the function will assign a value of 0 to pnSupported. If a support does exist at the node, the function will assign a value of 1 to the pnSupported variable.

Remarks

This function checks a given node in the currently open STAAD file to see whether a support exists at that node.

Example

```
Sub Supported ()

 Dim objOpenSTAAD As Output
 Dim pnIsItSupported As Integer

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")

  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp08.std"

'Parameters are node number and name for variable to store results.
'Note that the variable name for storing the results of the function can
' have any name you desire.

  objOpenSTAAD.DoesNodeHaveSupport 4, pnIsItSupported

'Free system resources before closing.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetNodeCoordinates
GetNodesCount
GetNodesCoordinates
GetNextNodeCoordinate
GetNumberOfSupportedNodes
GetAllNodesThatAreSupported
RenumberNodes

GetNumberOfSupportedNodes

VB Syntax

integer GetNumberOfSupportedNodes (long pnNodes)

Parameters

pnNodes

A long variable for storing the number of supported nodes retrieved by the function.

Remarks

This function retrieves the number of nodes that are connected to supports for the structure. This function will normally be used prior to the GetAllNodesThatAreSupported function; it will determine the size of the arrays needed to capture the values returned by the GetAllNodesThatAreSupported function.

Example

```
objOpenSTAAD.GetNumberOfSupportedNodes pnNodes
```

See Also

GetNodeCoordinates
GetNodesCount
GetNodesCoordinates
GetNextNodeCoordinate
DoesNodeHaveSupport
GetAllNodesThatAreSupported
RenumberNodes

GetAllNodesThatAreSupported

VB Syntax:

integer GetAllNodesThatAreSupported (long pnNodeNum, double pX, double pY, double pZ)

Parameters

pnNodeNum

A dynamic long array variable that stores node numbers of the supported nodes.

pX

A dynamic double (8-byte floating-point) array variable that stores the x coordinates of the supported nodes.

pY

A dynamic double (8-byte floating-point) array variable that stores the y coordinates of the supported nodes.

pZ

A dynamic double (8-byte floating-point) array variable that stores the z coordinates of the supported nodes.

Remarks

This function retrieves the node number and xyz coordinates for all supported nodes in the currently open STAAD file. The node numbers and coordinates are stored in dynamic arrays, since the size of the array will vary, depending on the number of supported nodes in the structure. The OpenSTAAD function GetNumberOfSupportedNodes can be used to retrieve the number of supported nodes in the structure. Then, the VB 'ReDim' function can be used to size the arrays.

Example

```
Sub GetCoordOfSupportedNodes()

 Dim objOpenSTAAD As Output
 Dim pnNodes As Long
```

```
'The next 4 are dynamic array variables, use empty paren. right now.
'We will use GetNumberOfSupportedNodes to find the size of the array,
' then we will use a VB ReDim function to size the array.

 Dim pnNodeNo() As Long
 Dim pX() As Double
 Dim pY() As Double
 Dim pZ() As Double

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp08.std"

'Find the number of supported nodes in the structure.

  objOpenSTAAD.GetNumberOfSupportedNodes pnNodes

'Now that we know how many supported nodes there are, we can REDIMension
' the arrays.

  ReDim pnNodeNo(pnNodes)
  ReDim pX(pnNodes), pY(pnNodes), pZ(pnNodes)

'Get the coordinates for all the supported nodes.

  objOpenSTAAD.GetAllNodesThatAreSupported pnNodeNo(0), pX(0), pY(0), pZ(0)

'Now write the results in columns 1-4 of the worksheet, starting on Row 20.
'Don't forget to specify the arrays' starting points, use i-1.

  For i = 1 To pnNodes
    Cells(19 + i, 1).Value = pnNodeNo(i - 1)
    Cells(19 + i, 2).Value = pX(i - 1)
    Cells(19 + i, 3).Value = pY(i - 1)
    Cells(19 + i, 4).Value = pZ(i - 1)
  Next

  ObjOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetNodeCoordinates
GetNodesCount
GetNodesCoordinates
GetNextNodeCoordinate
DoesNodeHaveSupport
GetNumberOfSupportedNodes
RenumberNodes

RenumberNodes

VB Syntax:

integer RenumberNodes (integer nOldNodeNo, integer nNewNodeNo)

Parameters

nOldNodeNo

An integer variable passed to the function specifying the existing node number which is to be changed to a new number.

nNewNodeNo

An integer variable passed to the function specifying the new number to be used to replace the existing node number.

Remarks

This function assigns a new number to an existing node. The old node number and the new number are both passed to the function. The function then modifies the input file so that every reference to the old node number is changed to the new node number.

Example

```
Sub ChangeNodeNumber()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Change the number of Node no. 1 to Node no. 100.

  objOpenSTAAD.RenumberNodes 1, 100

'Close the STAAD file and release the handles to the OpenSTAAD library.

  ObjOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetNodeCoordinates
GetNodesCount
GetNodesCoordinates
GetNextNodeCoordinate
DoesNodeHaveSupport
GetNumberOfSupportedNodes
GetAllNodesThatAreSupported

GetMemberIncidences

VB Syntax

integer GetMemberIncidences (long nMemberNo, long pnStartNode, long pnEndNode)

Parameters

nMemberNo

A long variable greater than or equal to one which specifies the member number for which the member incidences are to be obtained.

pnStartNode

A long variable for storing the starting node number retrieved by the function.

pnEndNode

A long variable for storing the ending node number retrieved by the function.

Remarks

This function retrieves the member incidences (starting node and ending node) for a given member. For example, in the STAAD input file, the statement 'MEMBER INCIDENCE 5 1 8' means that Member 5 starts at Node 1 and Ends at Node 8. Using this example, if the variable nMemberNo is given the value of 5, the GetMemberIncidences function would set pnStartNode = 1 and pnEndNode = 8. The starting node is also called member end A in STAAD; the ending node is also called member end B in STAAD.

Example

```
'Given MEMBER INCIDENCE 5 1 8, Member 5 starts at Node 1 and Ends at Node 8.
'This code snippet retrieves the member incidences for member 5

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.GetMemberIncidences 5, pnStartNode, pnEndNode

'pnStartNode is now set to a value of 1, pnEndNode = 8.
```

See Also

GetMemberLength

GetMembersCount
GetAllMemberIncidences
GetNextMember
RenumberMembers

GetMembersCount

VB Syntax

integer GetMembersCount (long pnMembers)

Parameters

pnMembers

A long variable for storing the number of members retrieved by the function.

Remarks

This function retrieves the number of members in the currently open STAAD file.

This function will normally be used prior to the GetAllMembersIncidences function; it will determine the size of the dynamic arrays needed to capture the values retrieved by the GetAllMembersIncidences function.

Example

```
'This is just a code snippet, not a complete macro.

 Dim nHowMany As Long

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.GetMembersCount nHowMany

'OpenSTAAD retrieves the number of members in the model and stores
' that number as a long value in the variable named nHowMany.
'Note that you can choose any legal variable name, it need not be
' called pnMembers.
'This is true in general for all variable names.
```

See Also

GetMemberLength
GetMemberIncidences
GetAllMemberIncidences
GetNextMember
RenumberMembers

GetAllMembersIncidences

VB Syntax

integer GetAllMembersIncidences (long pnMemberNumbers, long pnStartNodeNos, long pnEndNodeNos)

Parameters

pnMemberNumbers

A dynamic long array variable used by the function to store the member numbers it retrieves.

pnStartNodeNos

A dynamic long array variable used by the function to store the starting node (a.k.a. Node A) number for each member.

pnEndNodeNos

A dynamic long array variable used by the function to store the ending node (a.k.a. Node B) number for each member.

Remarks

This function retrieves the member numbers and member incidences (starting node and ending node) for all members in the currently open STAAD file. For example, in the STAAD input file, the statement 'MEMBER INCIDENCE 5 1 8' means that Member 5 starts at Node 1 and Ends at Node 8. Using this example, if the GetAllMembersIncidences function sets a given position in the pnMemberNumbers array to a value of 5, the function would set the corresponding position in the pnStartNodeNos array to a value of 1 and in pnEndNodeNos to a value of 8.

The starting node is sometimes referred to in STAAD as member end A; the ending node is sometimes called member end B.

The member numbers, starting node numbers and ending node numbers are stored in dynamic arrays, since the size of the array will vary, depending on the number of members in the structure. The *OpenSTAAD* function *GetMembersCount* can be used to retrieve the number of members in the structure. Then, the VB 'ReDim' function can be used to size the arrays.

Example

```
Sub AllIncidences()

 Dim objOpenSTAAD As Output
 Dim pnMembers As Long

'The next 3 are dynamic array variables, use empty paren. right now
'We will use GetMembersCount to find the size of the array,
' then we will use a ReDim statement to size the array.

 Dim pnMemberNumbers() As Long
 Dim pnStartNodeNos() As Long
 Dim pnEndNodeNos() As Long


  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp08.std"

'Find the number of members in the structure

  objOpenSTAAD.GetMembersCount pnMembers

'Now that we know how many members there are, we can REDIMension the arrays.

  ReDim pnMemberNumbers(pnMembers)
  ReDim pnStartNodeNos(pnMembers), pnEndNodeNos(pnMembers)

'Now we're ready to get the member numbers and incidences and write them into
' the arrays.  When passing array names, also need to specify the starting
' position (in parentheses).
'Note the use of the VB line continuation character, a space followed by an
' underscore in the following code, allowing a single code statement to
' be written on multiple lines.

  objOpenSTAAD.GetAllMembersIncidences _
                   pnMemberNumbers(0),  pnStartNodeNos(0), pnEndNodeNos(0)

'Now write the results in columns 1-4 of the worksheet, starting on Row 50.

  For i = 1 To pnMembers

    'Don't forget to specify the arrays' starting points, use i-1.

      Cells(49 + i, 1).Value = pnMemberNumbers(i - 1)
      Cells(49 + i, 2).Value = pnStartNodeNos(i - 1)
      Cells(49 + i, 3).Value = pnEndNodeNos(i - 1)
   Next i

'Close the STAAD file and release the OpenSTAAD object.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberLength
GetMemberIncidences
GetMembersCount
GetNextMember
RenumberMembers

GetNextMember

VB Syntax

integer GetNextMember (long nPreviousMemberNo, long pnNextMemberNo, long pnStartNodeNo, long pnEndNodeNo)

Parameters

nPreviousMemberNo

A long variable specifying the member number corresponding to the member prior (i.e., the antecedent, in ascending numerical order) to the member for which the member number and incidences are to be obtained. If this variable is set to a value of 0, the function will return the member number and incidences for the member with the lowest valid member number.

pnNextMemberNo

A long variable for the function to use in storing the member number it retrieves from STAAD.*Pro*.

pnStartNodeNo

A long variable in which the function will store the node number for the starting node (end A) of the member.

pnEndNodeNo

A long variable in which the function will store the node number for the ending node (end B) of the member.

Remarks

This function retrieves the member number and member incidences (starting and ending node numbers) for the next valid member number in ascending numerical order from a given member. This function may be useful where it is desired to obtain the member numbers and/or incidences for a limited range of members. It may also be helpful when it is desired to know the next member number in a STAAD model where not all integers within a given range have been assigned to a member.

To obtain the member number and incidences for the member with the lowest valid member number, set the value of *nPreviousMemberNo* to 0.

Example

```
Sub NextMemb()

'This macro allows us to enter a starting member number in an Excel worksheet.
'We also enter the number of nodes above the starting node we want the function '
to retrieve.
'The node numbers and coordinates are then displayed in the worksheet.

 Dim objOpenSTAAD As Output
 Dim nPreviousMemberNo As Long
 Dim nHowMany As Long
 Dim pnNextMemberNo As Long
 Dim pnStartNodeNo As Long
 Dim pnEndNodeNo As Long

'Run an instance of OpenSTAAD

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")

'Open STAAD's Example 8 (US)

  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp08.std"

'Get the starting member number from Row 48, Column 10 of the Excel worksheet

  nPreviousMemberNo = Cells(48, 10).Value

'Get from Row 48, Column 6 the number of succeeding members you want
' OpenSTAAD to retrieve.
'Note that we could have said 'For i = 1 To Cells(48,6).Value' but doing it
' the following way is more efficient.  We only retrieve the value once,
' instead of doing it each time we iterate through the loop.

  nHowMany = Cells(48, 6).Value

  For i = 1 To nHowMany

    'Now execute the function.
    'Note the use of the VB line continuation character, a space followed by an
    ' underscore in the following code, allowing a single code statement to
    ' be written on multiple lines.

      objOpenSTAAD.GetNextMember _
                  nPreviousMemberNo, pnNextMemberNo, pnStartNodeNo, pnEndNodeNo

    'Write the results starting in Row 50, Member No. in Column 6, starting
    ' node in column 7, etc.

      Cells(49 + i, 6).Value = pnNextMemberNo
      Cells(49 + i, 7).Value = pnStartNodeNo
      Cells(49 + i, 8).Value = pnEndNodeNo


    'Now, to prepare to reiterate, increment the value of the previous
    ' node number. The next node number now becomes the previous node number for
    ' the next iteration.

      nPreviousMemberNo = pnNextMemberNo

    'Do it again until you've done it the number of times you specified in the
    ' worksheet cell, Row 48 Column 6.

  Next

'Close the STAAD file and release the handles to the OpenSTAAD library.
```

```
  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberIncidences
GetMembersCount
GetAllMembersIncidences
RenumberMembers

RenumberMembers

VB Syntax

integer RenumberMembers (integer nOldMembNo, integer nNewMembNo)

Parameters

nOldMembNo

An integer variable passed to the function specifying the existing member number which is to be changed to a new number.

nNewMembNo

An integer variable passed to the function specifying the new number to be used to replace the existing member number.

Remarks

This function assigns a new number to an existing member. The old member number and the new number are both passed to the function. The function then modifies the input file so that every reference to the old member number is changed to the new member number.

Example

```
Sub ChangeBeamNumber()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Change the number of Beam no. 1 to Beam no. 100.

  objOpenSTAAD.RenumberMembers 1, 100

'Close the STAAD file and release the handles to the OpenSTAAD library.

  ObjOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberIncidences
GetMembersCount
GetAllMembersIncidences
GetNextMember

GetPlateIncidences

VB Syntax

integer GetPlateIncidences (long nPlateNo, long pnNodeA, long pnNodeB, long pnNodeC, long pnNodeD)

Parameters

nPlateNo

A long value greater than 0 specifying the plate in the model for which the incidences are to be retrieved.

pnNodeA

A long variable for storing the first plate incidence node number (Node A) retrieved by the function.

pnNodeB

A long variable for storing the second plate incidence node number (Node B) retrieved by the function.

pnNodeC

A long variable for storing the third plate incidence node number (Node C) retrieved by the function.

pnNodeD

A long variable for storing the fourth plate incidence node number (Node D) retrieved by the function. If the plate specified by the *nPlateNo* parameter is a 3-noded plate (triangular), the function will store a value of 0 in the *pnNodeD* variable.

Remarks

This function retrieves the plate incidences for a given plate in the currently open STAAD model and stores the node numbers in four integer variables passed to the function as parameters. STAAD plate elements may be either 3-noded (triangular) or 4-noded (quadrilateral). If the plate incidences retrieved by the function are for a 3-noded plate (triangular), the function will store a value of 0 in the *pnNodeD* variable. Please note that even if all the plates in your STAAD model are triangular,

you must still pass a name for the pnNodeD variable to the function, because this function expects 5 parameters.

Example

```
Sub PlateIncid()

 Dim objOpenSTAAD As Output
 Dim pnNodeA As Long
 Dim pnNodeB As Long
 Dim pnNodeC As Long
 Dim pnNodeD As Long

'Run an instance of OpenSTAAD

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")

'Open STAAD's Example 10 (US).

  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp10.std"

'Retrieve the plate incidences for Plate No. 8.

  objOpenSTAAD.GetPlateIncidences 8, pnNodeA, pnNodeB, pnNodeC, pnNodeD

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

Get Plates Count
GetAllPlatesIncidences
GetPlateEdgeDistances

GetPlatesCount

VB Syntax

integer GetPlatesCount (long pnPlates)

Parameters

pnPlates

A long variable used by the function to store the number of plates.

Remarks

This function retrieves the number of plates in the currently open STAAD model, and stores that number in an integer variable.

Example

```
Sub HowManyPlates()

 Dim objOpenSTAAD As Output
 Dim pnPlates As Long

'Run an instance of OpenSTAAD

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")

'Open STAAD's Example 10 (US).

  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp10.std"

'Retrieve the number of plates in the examp10.std model.

  objOpenSTAAD.GetPlatesCount pnPlates

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetPlateIncidences
GetAllPlatesIncidences
GetPlateEdgeDistances

GetAllPlatesIncidences

VB Syntax

integer GetAllPlatesIncidences (long pnPlateNos, long pnNodeA, long pnNodeB, long pnNodeC, long pnNodeD)

Parameters

pnPlateNos

A dynamic long array variable used by the function to store the plate numbers it retrieves.

pnNodeA

A dynamic long array variable used by the function to store the number of Node A for each plate.

pnNodeB

A dynamic long array variable used by the function to store the number of Node B for each plate.

pnNodeC

A dynamic long array variable used by the function to store the number of Node C for each plate.

pnNodeD

A dynamic long array variable used by the function to store the number of Node D for each plate. If the plate is a 3-noded plate (triangular), the function will set this variable to a value of 0.

Remarks

This function retrieves the plate numbers and plate incidences for all plates in the currently open STAAD file.

The plate numbers and incidences (node numbers) are stored in dynamic arrays, since the size of the array will vary, depending on the number of plates in the structure. The *OpenSTAAD* function *GetPlatesCount* can be used to retrieve the number of plates in the structure. Then, the VB 'ReDim' function can be used to size the arrays.

STAAD plate elements can be 3-noded (triangular) or 4-noded (quadrilateral). If the plate incidences retrieved by the function are for a 3-noded plate (triangular), the function will store a value of 0 in the *pnNodeD* variable. Please note that even if all the plates in your STAAD model are triangular, you must still pass a name for the pnNodeD variable to the function, because this function expects 5 parameters.

Example

```
Sub AllPlatesIncid()

 Dim objOpenSTAAD As Output
 Dim pnPlates As Long

'The next 4 are dynamic array variables, use empty paren. right now.
'We will use GetPlatesCount to find out what size the arrays should be,
' then we will use a ReDim statement to size the arrays.

 Dim pnPlateNos() As Long
 Dim pnNodeA() As Long
 Dim pnNodeB() As Long
 Dim pnNodeC() As Long
 Dim pnNodeD() As Long

'Run an instance of OpenSTAAD

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")

'Open STAAD's Example 10 (US)

  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp10.std"

'Retrieve the number of plates in the examp10.std model.

  objOpenSTAAD.GetPlatesCount pnPlates

'Now that we know how many plates there are, we can REDIMension the arrays.

  ReDim pnPlateNos(pnPlates)
  ReDim pnNodeA(pnPlates), pnNodeB(pnPlates), pnNodeC(pnPlates)
  ReDim pnNodeD(pnPlates)

'When passing array names, also need to specify the starting position
' (in parentheses)
'Note the use of the VB line continuation character, a space followed by an
' underscore in the following code, allowing a single code statement to
' be written on multiple lines.

  objOpenSTAAD.GetAllPlatesIncidences _
                pnPlateNos(0), pnNodeA(0), pnNodeB(0), pnNodeC(0), pnNodeD(0)

'Now write the results in columns 1-5 of the worksheet, starting on Row 80.

  For i = 1 To pnPlates

    'Remember to specify the arrays' starting points, use i-1

      Cells(79 + i, 1).Value = pnPlateNos(i - 1)
      Cells(79 + i, 2).Value = pnNodeA(i - 1)
      Cells(79 + i, 3).Value = pnNodeB(i - 1)
      Cells(79 + i, 4).Value = pnNodeC(i - 1)
      Cells(79 + i, 5).Value = pnNodeD(i - 1)
  Next

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing
```

```
End Sub
```

See Also

GetPlateIncidences
GetPlatesCount
GetPlateEdgeDistances

GetPlateEdgeDistances

VB Syntax

integer GetPlateEdgeDistances (integer nPlateNo, double pdLengths)

Parameters

nPlateNo

An integer variable passed to the function to specify the number of the plate for which the function is to retrieve the edge distances.

double pdLengths

An array of four double values for the function to use in storing the edge distances it retrieves from STAAD.Pro.

Remarks

This function retrieves the edge distances for a given plate in the currently open STAAD file. The number of the plate for which the function is to retrieve the edge distances, and a variable name for storing the results are passed to the function. The function then stores the edge distances in the *pdLengths* array variable. If the plate has four sides (i.e. a four-noded plate), the array is filled in the order AB, BC, CD, DA. If the plate has only three sides (three node plate), the array is filled in the order AB, BC, CA, and a value of 0 is assigned to the fourth position in the *pdLengths* array.

Example

```
Sub PlateEdgeLengths()

 Dim objOpenSTAAD As Output
 Dim pnPlateNo As Integer
 Dim pdLengths(3) As Double

'Run an instance of OpenSTAAD

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")

'Open STAAD's Example 10 (US).

  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp10.std"

'Retrieve the edge distances for plate no. 23.

pnPlateNo = 23
objOpenSTAAD.GetPlateEdgeDistances pnPlateNo, pdLengths(0)
```

```
'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetPlateIncidences
GetPlatesCount
GetAllPlatesIncidences

GetSolidIncidences

VB Syntax

integer GetSolidIncidences (long nSolidNo, long pnNodeNos)

Parameters

nSolidNo

A long variable greater than or equal to one specifying the solid number for which the function is to retrieve the incidences.

pnNodeNos

An array of 8 long values which the function will use to store the solid incidences (node numbers) it retrieves from STAAD.*Pro*.

Remarks

This function retrieves the incidences for a given solid in the currently open STAAD model.

STAAD solid elements consist of 8 nodes. By collapsing various nodes together, an 8-noded solid element can be degenerated into forms with 4 to 7 nodes.

Example

```
Sub SolidIncid()

'Declare an OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare an array of 8 integer values for storing the function results.

 Dim pnNodeNos(0 To 7) As Long

'Run an instance of OpenSTAAD and open Example 24 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp24.std"

'Get the incidences for Solid No. 9 and store the values in the array pnNodeNos.

  objOpenSTAAD.GetSolidIncidences 9, pnNodeNos(0)

'Display the values from the array pnNodeNos in the worksheet, Row 2,
' Columns 1-8.

    For i = 1 To 8
      Cells(2, i).Value = pnNodeNos(i - 1)
```

```
    Next i

'Close the STAAD file and release the handles to OpenSTAAD.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetSolidsCount
GetAllSolidsIncidences

GetSolidsCount

VB Syntax

integer GetSolidsCount (long pnSolids)

Parameters

pnSolids

A long variable used by the function to store the number of solids it retrieves.

Remarks

This function retrieves the number of solid elements in the currently open STAAD model.

Example

```
Sub HowManySolids()

'Declare an OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare an integer variable to store the function results.

 Dim pnSolids As Long

'Run an instance of OpenSTAAD and open Example 24 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp24.std"

'Get the number of solids and store the value in the integer variable pnSolids.

  objOpenSTAAD.GetSolidsCount pnSolids

'Close the STAAD file and release the handles to OpenSTAAD.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetSolidIncidences
GetAllSolidsIncidences

GetAllSolidsIncidences

VB Syntax

integer GetAllSolidsIncidences (long pnSolidNos, long pnNodeA, long pnNodeB, long pnNodeC, long pnNodeD, long pnNodeE, long pnNodeF, long pnNodeG, long pnNodeH)

Parameters

pnSolidNos

A dynamic long array variable used by the function to store the solid element numbers it retrieves.

pnNodeA

A dynamic long array variable used by the function to store the number of Node A for each solid element.

pnNodeB

A dynamic long array variable used by the function to store the number of Node B for each solid element.

pnNodeC

A dynamic long array variable used by the function to store the number of Node C for each solid element.

pnNodeD

A dynamic long array variable used by the function to store the number of Node D for each solid element.

pnNodeE

A dynamic long array variable used by the function to store the number of Node E for each solid element.

pnNodeF

A dynamic long array variable used by the function to store the number of Node F for each solid element.

pnNodeG

A dynamic long array variable used by the function to store the number of Node G for each solid element.

pnNodeH

A dynamic long array variable used by the function to store the number of Node H for each solid element.

Remarks

This function retrieves the solid numbers and solid incidences for all solid elements in the currently open STAAD file.

The solid elements' numbers and incidences (node numbers) are stored in dynamic arrays, since the size of the array will vary, depending on the number of solids in the structure. The *OpenSTAAD* function *GetSolidsCount* can be used to retrieve the number of solid elements in the structure. Then, the VB 'ReDim' function can be used to size the arrays.

Example

```
Sub AllSolidIncid()

 Dim objOpenSTAAD As Output
 Dim pnSolids As Long

'The next 9 are dynamic array variables, use empty paren. right now.
'We will use GetSolidsCount to find what size arrays are needed,
' then we will use a ReDim statement to size the arrays.

 Dim pnSolidNos() As Long
 Dim pnNodeA() As Long
 Dim pnNodeB() As Long
 Dim pnNodeC() As Long
 Dim pnNodeD() As Long
 Dim pnNodeE() As Long
 Dim pnNodeF() As Long
 Dim pnNodeG() As Long
 Dim pnNodeH() As Long

'Run an instance of OpenSTAAD and open Example 24 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp24.std"

'Get the number of solids and store the value in the integer variable pnSolids.

  objOpenSTAAD.GetSolidsCount pnSolids

 'Now that we know how many solids there are in Example 24, we can
' REDIMension the arrays
```

```
  ReDim pnSolidNos(pnSolids)
  ReDim pnNodeA(pnSolids), pnNodeB(pnSolids)
  ReDim pnNodeC(pnSolids), pnNodeD(pnSolids)
  ReDim pnNodeE(pnSolids), pnNodeF(pnSolids)
  ReDim pnNodeG(pnSolids), pnNodeH(pnSolids)

'When passing array names, also need to specify the starting position
'  (in parentheses).
'Note the use of the VB line continuation character, a space followed by an
' underscore in the following code, allowing a single code statement to
' be written on multiple lines.

  objOpenSTAAD.GetAllSolidsIncidences pnSolidNos(0),  _
              pnNodeA(0), pnNodeB(0), pnNodeC(0), pnNodeD(0), _
              pnNodeE(0), pnNodeF(0), pnNodeG(0), pnNodeH(0)

'Now write the results in columns 1-9 of the worksheet, starting on Row 10.

  For i = 1 To pnSolids

    'Remember to specify the arrays' starting points, use i-1.

      Cells(9 + i, 1).Value = pnSolidNos(i - 1)
      Cells(9 + i, 2).Value = pnNodeA(i - 1)
      Cells(9 + i, 3).Value = pnNodeB(i - 1)
      Cells(9 + i, 4).Value = pnNodeC(i - 1)
      Cells(9 + i, 5).Value = pnNodeD(i - 1)
      Cells(9 + i, 6).Value = pnNodeE(i - 1)
      Cells(9 + i, 7).Value = pnNodeF(i - 1)
      Cells(9 + i, 8).Value = pnNodeG(i - 1)
      Cells(9 + i, 9).Value = pnNodeH(i - 1)
  Next i

'Close the STAAD file and release the handles to OpenSTAAD.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetSolidIncidences
GetSolidsCount

WriteGeometry

VB Syntax

integer WriteGeometry (string strFileName)

Parameters

strFileName

A string for the function to use to create a file in which it will store the geometry for the currently open STAAD project file. This VB string should be enclosed in quotation marks. It should include the complete path to the location of an existing directory on the computer, to specify to the function the location where the file it creates is to be stored, followed by the filename and desired extension, if any. Since this function writes the file in the same format as a STAAD.Pro input file, the extension STD would be a logical choice for the filename extension.

Remarks

This function retrieves the node coordinates, member incidences, plate incidences and solid incidences in the currently open STAAD project. A filename and path is passed to the function as a string, or in a string variable. The function creates the geometry file at the specified location and with the specified name. It then stores the node coordinates, member incidences, plate incidences and solid incidences in the geometry file in STAAD.Pro input file format.

Example

```
Sub CreateProjectGeometryFile()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Create a file named geometryfile.std.
'Store it in the temp folder on the C drive.
'Then write the project geometry into the file.

  objOpenSTAAD.WriteGeometry "C:\temp\geometryfile.std"

'Close the STAAD file and release the handles to the OpenSTAAD library.

  ObjOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

Functions Related to Groups

GetNumberOfGROUPS

VB Syntax

integer GetNumberOfGROUPS (integer pnGroups)

Parameters

pnGroups

An integer variable for the function to use in storing the number of groups it retrieves from STAAD.Pro.

Remarks

This function retrieves the number of groups in the currently open STAAD file and stores the result in the *pnGroups* variable. This number will be the sum of all types of groups in the project, including node, beam, plate, solid and geometry groups.

VBA is not case-sensitive, however if you are writing your OpenSTAAD program in an application such as C++ which *is* case-sensitive, you must be sure to use upper and lower case characters in this function's name as indicated, i.e. GetNumberOfGROUPS.

Example

```
Sub HowManyGroups()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare an integer variable name for storing the function results.

 Dim pnGroups As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve the number of groups in examp01.std.

  objOpenSTAAD.GetNumberOfGROUPS pnGroups

'Close the STAAD file and release the handles to the OpenSTAAD library.
```

```
  ObjOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetNumberOfGROUPTypes
GetGROUPNamesForType
GetNumberOfEntitiesInGROUP
GetAllEntitiesInGROUP

GetNumberOfGROUPTypes

VB Syntax

integer GetNumberOfGROUPTypes (integer pnNodeGroups, integer pnBeamGroups, integer pnPlateGroups, integer pnSolidGroups, integer pnGeomGroups)

Parameters

pnNodeGroups

An integer variable for the function to use in storing the number of node groups it retrieves from STAAD.Pro.

pnBeamGroups

An integer variable for the function to use in storing the number of beam groups it retrieves from STAAD.Pro.

pnPlateGroups

An integer variable for the function to use in storing the number of groups of plate elements it retrieves from STAAD.Pro.

pnSolidGroups

An integer variable for the function to use in storing the number of groups of solid elements it retrieves from STAAD.Pro.

pnGeomGroups

An integer variable for the function to use in storing the number of geoemetry groups it retrieves from STAAD.Pro.

Remarks

This function retrieves the number of groups of each type in the currently open STAAD file. There are five types of groups in STAAD.*Pro*: node, beam, plate, solid and geometry groups. Five variable names are passed to the function, e.g., pnNodeGroups, pnBeamGroups, pnPlateGroups, pnSolidGroups and pnGeomGroups (it is not necessary to use these particular names; any allowable VBA integer variable names may be used). The function then stores the number of

node groups in the first variable, the number of beam groups in the second variable, the number of plate groups in the third variable, the number of solid groups in the fourth variable, and the number of geometry groups in the fifth variable.

VBA is not case-sensitive, however if you are writing your OpenSTAAD program in an application such as C++ which *is* case-sensitive, you must be sure to use upper and lower case characters in this function's name as indicated, i.e. GetNumberOfGROUPTypes.

Example

```
Sub HowManyOfEachTypeOfGroup()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare 5 integer variable names for storing the function results.

 Dim pnNodeGroups As Integer
 Dim pnBeamGroups As Integer
 Dim pnPlateGroups As Integer
 Dim pnSolidGroups As Integer
 Dim pnGeomGroups As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve the number of each type of group in examp01.std.
'Note the use of the VB line continuation character, a space followed by an
' underscore in the following code, allowing a single code statement to
' be written on multiple lines.

  objOpenSTAAD.GetNumberOfGROUPTypes _
        pnNodeGroups, pnBeamGroups, pnPlateGroups, pnSolidGroups, pnGeomGroups

'Close the STAAD file and release the handles to the OpenSTAAD library.

  ObjOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetNumberOfGROUPS
GetGROUPNamesForType
GetNumberOfEntitiesInGROUP
GetAllEntitiesInGROUP

GetGROUPNamesForType

VB Syntax

integer GetGROUPNamesForType (integer nType, integer pnGroupCount, string pstrNames)

Parameters

nType

An integer variable passed to the function to specify the group type for which the function is to retrieve the group names. *nType* = 0 for node groups, 1 for beam groups, 2 for plate groups, 3 for solid groups, and 4 for geometry groups.

pnGroupCount

An integer variable for the function to use in storing the number of groups of type *nType* in the structure.

string pstrNames

A dynamic string array variable for the function to use in storing the names of the groups of type *nType* in the structure.

Remarks

This function retrieves the number of groups of a specified type and the names of those groups in the currently open STAAD file. There are five types of groups in STAAD.*Pro*: node, beam, plate, solid and geometry groups. The group type for which the number of groups and their names are to be retrieved is passed to the function as the first parameter *nType*, where *nType* = 0 for node groups, 1 for beam groups, 2 for plate groups, 3 for solid groups, and 4 for geometry groups. The function stores the number of groups of the type specified by *nType* in the integer variable *pnGroupCount*. The function stores the group names in a dynamic string array *pstrNames*. The size of the array will vary, depending on the number of groups of type *nType* in the structure. The OpenSTAAD function *GetNumberOfGROUPTypes* can be used to retrieve the number of groups of type *nType* in the structure. Then, the VB "ReDim" function can be used to size the *pstrNames* array before calling the *GetGROUPNamesForType* function.

VBA is not case-sensitive, however if you are writing your OpenSTAAD program in an application such as C++ which *is* case-sensitive, you must be sure to use upper

and lower case characters in this function's name as indicated, i.e. GetGROUPNamesForType.

Example

```
Sub NosAndNamesOfTypeGroup()

'This sample macro will demonstrate how to retrieve the number of beam groups
' and their names that exist in a dummy project we shall call myproject.

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare an integer variable name for storing the number of beam groups.

 Dim pnGroupCount As Integer

'The next variable is a dynamic array for storing the beam group names.  Use
' empty paren. right now.  We will use GetNumberOfGROUPTypes to find the number
' of beam groups in the structure, then we will use a VB ReDim function to size '
the array.

Dim pstrNames() As String

'Run an instance of OpenSTAAD and open a project file myproject.std.

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\Projects\MyProject.std"

'Determine the number of groups of each type in the structure.

  objOpenSTAAD.GetNumberOfGROUPTypes _
          pnNodeGroups, pnBeamGroups, pnPlateGroups, pnSolidGroups, pnGeomGroups

'Now that we know how many beam groups there are, we can resize pstrNames array.

  ReDim pstrNames(1 To pnBeamGroups)

'Retrieve the number of beam groups and their names in myproject.std.

  objOpenSTAAD.GetGROUPNamesForType 1, pnGroupCount, pstrNames(1)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  ObjOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetNumberOfGROUPS
GetNumberOfGROUPTypes
GetNumberOfEntitiesInGROUP
GetAllEntitiesInGROUP

GetNumberOfEntitiesInGROUP

VB Syntax

integer GetNumberOfEntitiesInGROUP (integer nType, string szGroupName, integer pnTotalCount)

Parameters

nType

An integer variable passed to the function to specify the group type for which the function is to retrieve the number of entities. *nType* = 0 for groups of node entities, 1 for groups of beam entities, 2 for groups of plate entities, 3 for groups of solid entities, and 4 for groups of geometry entities.

szGroupName

An string variable passed to the function to specify the name of the group for which the function is to retrieve the number of entities.

pnTotalCount

An integer variable for the function to use in storing the number of entities in the *szGroupName* group.

Remarks

This function retrieves the number of entities in a specified group in the currently open STAAD file. There are five types of groups in STAAD.*Pro*: node, beam, plate, solid and geometry groups. The group type for which the number of groups and their names are to be retrieved is passed to the function as the first parameter *nType*, where *nType* = 0 for node groups, 1 for beam groups, 2 for plate groups, 3 for solid groups, and 4 for geometry groups. A string variable specifying the group name for which the number of entities are to be retrieve is also passed to the function. The function then returns the number of entities in the group and stores that number in the *pnTotalCount* variable.

VBA is not case-sensitive, however if you are writing your OpenSTAAD program in an application such as C++ which *is* case-sensitive, you must be sure to use upper and lower case characters in this function's name as indicated, i.e. GetNumberOfEntitiesInGROUP.

Example

```
Sub HowManyInGroup()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare an integer variable name for storing the number of entities.

 Dim pnTotalCount As Integer

'Run an instance of OpenSTAAD and open a project file myproject.std.

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\Projects\MyProject.std"

'Determine the number of entities in the group of beams named _Members.

  objOpenSTAAD.GetNumberOfEntitiesInGroup 1, "_Members", pnTotalCount

'Close the STAAD file and release the handles to the OpenSTAAD library.

  ObjOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetNumberOfGROUPS
GetNumberOfGROUPTypes
GetGROUPNamesForType
GetAllEntitiesInGROUP

GetAllEntitiesInGROUP

VB Syntax

integer GetAllEntitiesInGROUP (integer nType, string szGroupName, integer pnCount, long pnList)

Parameters

nType

An integer variable passed to the function to specify the group type for which the function is to retrieve the number of entities in the group and the identification numbers assigned to those entities. *nType* = 0 for groups of node entities, 1 for groups of beam entities, 2 for groups of plate entities, 3 for groups of solid entities, and 4 for groups of geometry entities.

szGroupName

An string variable passed to the function to specify the name of the group for which the function is to retrieve the number of entities in the group and the identification numbers assigned to those entities.

pnCount

An integer variable for the function to use in storing the number of entities in the *szGroupName* group.

pnList

A dynamic long array variable for the function to use in storing the identification numbers of the entities in the *szGroupName* group.

Remarks

This function retrieves the number of entities in a specified group in the currently open STAAD file. There are five types of groups in STAAD.*Pro*: node, beam, plate, solid and geometry groups. The group type for which the number of entities and their identification numbers are to be retrieved is passed to the function as the first parameter *nType*, where *nType* = 0 for node groups, 1 for beam groups, 2 for plate groups, 3 for solid groups, and 4 for geometry groups. A string variable (*szGroupName*) specifying the group name for which the number of entities are to be retrieved is also passed to the function. The function stores the number of entities

in the group in the *pnCount* variable. The function then stores the identification numbers of the entities in the group in a dynamic long array (*pnList*). The size of the array will vary, depending on the number of entities in the group. The OpenSTAAD function *GetNumberOfEntitiesInGroup* can be used to retrieve the number of groups of type *nType* in the structure. Then, the VB "ReDim" function can be used to size the *pnList* array before calling the *GetAllEntitiesInGroup* function.

VBA is not case-sensitive, however if you are writing your OpenSTAAD program in an application such as C++ which *is* case-sensitive, you must be sure to use upper and lower case characters in this function's name as indicated, i.e. GetAllEntitiesInGROUP.

Example

```
Sub ListEntityNumbersInGroup()

'This sample macro will demonstrate how to retrieve the number of beams
' and their ID numbers from a group called _Members in a dummy project we
' shall call myproject.

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare integer variable names for storing the number of beams.

 Dim pnTotalCount As Integer
 Dim pnCount As Integer

'The next variable is a dynamic array for storing the beam numbers.  Use
' empty paren. right now.  We will use GetNumberOfEntitiesInGROUP to find the
' number of beams in the group, then we will use a VB ReDim function to size
' the array.

 Dim pnList() As String

'Run an instance of OpenSTAAD and open a project file myproject.std.

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\Projects\MyProject.std"

'Determine the number of entities in the group of beams named _Members.

  objOpenSTAAD.GetNumberOfEntitiesInGROUP 1, "_Members", pnTotalCount

'Now that we know how many beams there are in the _Members group, we can
' resize the pnList array.

  ReDim pnList(1 To pnTotalCount)

'Retrieve the number of beams and their ID numbers in the _Members group of the
' structure model myproject.

  objOpenSTAAD.GetGROUPNamesForType 1, "_Members", pnCount, pnList(1)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  ObjOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetNumberOfGROUPS
GetNumberOfGROUPTypes
GetGROUPNamesForType
GetNumberOfEntitiesInGROUP

Member Specifications Functions

GetSupportCondition

VB Syntax

integer GetSupportCondition (integer nNodeNo, integer pnSupportCond)

Parameters

nNodeNo

An integer value greater than 0 specifying the number of the node for which the function is to retrieve the support condition.

pnSupportCond

The name of an integer variable for the function to use in storing the support condition it retrieves.

0 = NoSupport	5 = InclinedPinned
1 = Fixed	6 = InclinedFixedBut
2 = Pinned	7 = Footing
3 = FixedBut	8 = ElasticMat
4 = InclinedFixed	9 = MultiLinear

Remarks

This function retrieves the support condition at a given node in the currently open STAAD model. The support condition is stored as an integer value representing one of ten possible support conditions.

0) NoSupport	No support exists at the node.
1) Fixed	The joint is fixed in all 6 degrees of freedom.
2) Pinned	No moments can be carried by this support.
3) FixedBut	A fixed support that has been released in specified global directions (Fx for force-X

	through Mz for moment Z). Spring constants may specify the support condition in a given global direction.
4) InclinedFixed	The support is fixed in all 6 degrees of freedom, but is inclined with respect to the global axes.
5) InclinedPinned	The support is inclined with respect to the global axes and it cannot carry any moments.
6) InclinedFixedBut	The support is inclined with respect to the global axes. It is fixed but with releases specified in certain global directions.
7) Footing	The support is modeled as a footing with a given influence area and sub-grade modulus (see Section 5.27.3 in the STAAD.Pro Technical Reference Manual).
8) ElasticMat	The support is modeled as an elastic mat. The program calculates the influence area of the support, and then the stiffness, given the sub-grade modulus and degrees of freedom in specified global directions (see Section 5.27.3 in the STAAD.Pro Technical Reference Manual).
9) MultiLinear	The support is modeled as a multi-linear spring (see Section 5.27.4 in the STAAD.Pro Technical Reference Manual).

Example

```
Sub SupCond()

 Dim objOpenSTAAD As Output

'Declare an integer variable for storing the function results.

 Dim pnSupportCond As Integer

'Run an instance of OpenSTAAD and open Example 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Get the support condition for Node 1 and store the result in the
' pnSupportCond integer variable.

  objOpenSTAAD.GetSupportCondition 1, pnSupportCond

'Close the STAAD file and release the handles to OpenSTAAD.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing
```

```
End Sub
```

See Also

GetSupportStiffness
GetSupportReactions

GetMemberOffsets

VB Syntax

GetMemberOffsets (integer nMemberNo, integer nEnd, integer pnGlobal, double pdX, double pdY, double pdZ)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to determine whether the member has releases.

nEnd

Specify a 0 to retrieve the member offsets for the starting end of the member (start joint number of member incidence) or a 1 to retrieve the member offsets for the end of the member (end joint number of member incidence).

pnGlobal

An integer variable for the function to use in storing the value it returns for the coordinate axis system, either a 0 or a 1. If the member offsets are specified in the global coordinate system, the function will return a 1 and store it in the *pnGlobal* variable; if the member offsets are specified in the member's local coordinate system, the function will return a 0 and store it in the *pnGlobal* variable.

pdX

A double (8-byte floating point) variable for the function to use in storing the member offset in the X direction it retrieves from STAAD.*Pro*.

pdY

A double (8-byte floating point) variable for the function to use in storing the member offset in the Y direction it retrieves from STAAD.*Pro*.

pdZ

A double (8-byte floating point) variable for the function to use in storing the member offset in the Z direction it retrieves from STAAD.*Pro*.

Remarks

This function retrieves the member offsets for a given end of a given member. The member number and member end are passed to the function as the first two parameters. The function returns an integer variable either a 0 or a 1 to indicate whether the offset distances are specified with respect to the global coordinate system, or in the member's local coordinate system. If the member offsets are specified in the global coordinate system, the function will return a 1 and store it in the *pnGlobal* variable; if the member offsets are specified in the member's local coordinate system, the function will return a 0 and store it in the *pnGlobal* variable. The function then returns the member offsets in the X, Y and Z directions and stores them in the pdX, pdY and pdZ variables respectively.

All values are given in units of kips and inches.

Example

```
Sub MembOffsets()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare variables for storing function results.

 Dim pnGlobal As Integer
 Dim pdX As Double
 Dim pdY As Double
 Dim pdZ As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 7 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp07.std"

'Determine member offsets for the starting end of member 6.

  objOpenSTAAD.GetMemberOffsets 6, 0, pnGlobal, pdX, pdY, pdZ

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

DoesMemberHaveReleases
IsStartEndReleased
IsEndEndReleased
GetDOFReleasedAtStartOfMember
GetDOFReleasedAtEndOfMember

DoesMemberHaveReleases

VB Syntax

integer DoesMemberHaveReleases (integer nMemberNo, integer pnStart, integer pnEnd)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to determine whether the member has releases.

pnStart

An integer variable name for storing the results returned by the function for the starting end (end A) of the member. A value of 0 stored in *pnStart* indicates that none of the degrees of freedom are released; a value of 1 indicates that one or more degrees of freedom are released.

pnEnd

An integer variable name for storing the results returned by the function for the ending end (end B) of the member. A value of 0 stored in *pnEnd* indicates that none of the degrees of freedom are released; a value of 1 indicates that one or more degrees of freedom are released.

Remarks

This function determines whether any degrees of freedom at either end of a given member have been released. The member number is passed to the function, then the function returns either a 0 or a 1 for each end of the member and stores those values in two variable names for the starting end (end A) and the ending end (end B) values respectively. A value of 0 indicates that none of the degrees of freedom are released; a value of 1 indicates that one or more degrees of freedom are released.

Note that Starting End and Ending End is based on the Member Incidence specification.

Example

```
Sub GotReleases()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare variables for storing function results.

 Dim pnStart As Integer
 Dim pnEnd As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Determine whether Member 3 has releases at either the start or the end.

  objOpenSTAAD.DoesMemberHaveReleases 3, pnStart, pnEnd

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberOffsets
IsStartEndReleased
IsEndEndReleased
GetDOFReleasedAtStartOfMember
GetDOFReleasedAtEndOfMember

IsStartEndReleased

VB Syntax

integer IsStartEndReleased (integer nMemberNo, integer pnReleased)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to determine whether the starting end is released.

pnReleased

An integer variable name for storing the results returned by the function. A value of 0 stored in *pnReleased* indicates that none of the degrees of freedom are released at the starting end of the member; a value of 1 indicates that one or more degrees of freedom are released.

Remarks

This function determines whether any degrees of freedom at the starting end (end A) of a given member have been released. The member number is passed to the function, then the function returns either a 0 or a 1 and stores the value in a variable name passed to it as the second parameter. A value of 0 indicates that none of the degrees of freedom are released; a value of 1 indicates that one or more degrees of freedom are released.

Note that Starting End and Ending End is based on the Member Incidence specification.

Example

```
Sub StartEndRel()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare an integer variable for storing function results.

 Dim pnReleased As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"
```

```
'Determine whether Member 3 has releases at the starting end.

  objOpenSTAAD.IsStartEndReleased 3, pnReleased

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberOffsets
DoesMemberHaveReleases
IsEndEndReleased
GetDOFReleasedAtStartOfMember
GetDOFReleasedAtEndOfMember

IsEndEndReleased

VB Syntax

integer IsEndEndReleased (integer nMemberNo, integer pnReleased)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to determine whether the end is released.

pnReleased

An integer variable name for storing the results returned by the function. A value of 0 stored in *pnReleased* indicates that none of the degrees of freedom are released at the ending end of the member; a value of 1 indicates that one or more degrees of freedom are released.

Remarks

This function determines whether any degrees of freedom at the ending end (end B) of a given member have been released. The member number is passed to the function, then the function returns either a 0 or a 1 and stores the value in a variable name passed to it as the second parameter. A value of 0 indicates that none of the degrees of freedom are released; a value of 1 indicates that one or more degrees of freedom are released at the ending end.

Note that Starting End and Ending End is based on the Member Incidence specification.

Example

```
Sub EndRel()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare integer variable for storing function results.

 Dim pnReleased As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"
```

```
'Determine whether Member 3 has releases at the ending end.

  objOpenSTAAD.IsEndEndReleased 3, pnReleased

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberOffsets
DoesMemberHaveReleases
IsStartEndReleased
GetDOFReleasedAtStartOfMember
GetDOFReleasedAtEndOfMember

GetDOFReleasedAtStartOfMember

VB Syntax

integer GetDOFReleasedAtStartOfMember (integer nMemberNo, integer pnDOFs)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to retrieve the degrees of freedom at the start of the member.

pnDOFs

An array of 6 integer values representing the 6 degrees of freedom FX, FY, FZ, MX, MY and MZ respectively at the starting end of the member. A 0 in the array indicates that the corresponding degree of freedom is fixed; a 1 indicates that the corresponding degree of freedom has been released.

Remarks

This function determines the degrees of freedom at the starting end (end A) of a member and stores the results in an array of 6 integer values. These values represent the 6 degrees of freedom FX, FY, FZ, MX, MY and MZ respectively. A 0 in the array indicates that the corresponding degree of freedom is fixed; a 1 indicates that the corresponding degree of freedom has been released.

For example, if the function returns values of 1,1,1,0,0,0 to the array, the starting end of the member is pinned; it can resist axial forces FX, FY and FZ, but not moments MX, MY and MZ.

Note that Starting End and Ending End is based on the Member Incidence specification.

Example

```
Sub DOFatStart()

'Declare an OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare an array of 6 integer values for storing the function results.

 Dim pnDOFs(6) As Integer

'Run an instance of OpenSTAAD and open Example 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Get the degrees of freedom for the starting end of member 5 and store
' the result in the pnDOFs integer array.

  objOpenSTAAD.GetDOFReleasedAtStartOfMember 5, pnDOFs(0)

  Cells(10, 12).Value = pnDOFs(0)
  Cells(11, 12).Value = pnDOFs(1)
  Cells(12, 12).Value = pnDOFs(2)
  Cells(13, 12).Value = pnDOFs(3)
  Cells(14, 12).Value = pnDOFs(4)
  Cells(15, 12).Value = pnDOFs(5)

'Close the STAAD file and release the handles to OpenSTAAD.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberOffsets
DoesMemberHaveReleases
IsStartEndReleased
IsEndEndReleased
GetDOFReleasedAtEndOfMember

GetDOFReleasedAtEndOfMember

VB Syntax

integer GetDOFReleasedAtEndOfMember (integer nMemberNo, integer pnDOFs)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to retrieve the degrees of freedom released at the end of the member.

pnDOFs

An array of 6 integer values representing the 6 degrees of freedom FX, FY, FZ, MX, MY and MZ respectively at the ending end (end B) of the member. A 0 in the array indicates that the corresponding degree of freedom is fixed; a 1 indicates that the corresponding degree of freedom has been released.

Remarks

This function determines the degrees of freedom at the ending end (end B) of a member and stores the results in an array of 6 integer values. These values represent the 6 degrees of freedom FX, FY, FZ, MX, MY and MZ respectively. A 0 in the array indicates that the corresponding degree of freedom is fixed; a 1 indicates that the corresponding degree of freedom has been released.

For example, if the function returns values of 1,1,1,0,0,0 to the array, the ending end of the member is pinned; it can resist axial forces FX, FY and FZ, but not moments MX, MY and MZ.

Note that Starting End and Ending End is based on the Member Incidence specification.

Example

```
Sub DOFatEnd()
```

```
'Declare an OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare an array of 6 integer values for storing the function results.

 Dim pnDOFs(6) As Integer

'Run an instance of OpenSTAAD and open Example 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Get the degrees of freedom for the starting end of member 5 and store
' the result in the pnDOFs integer array.

  objOpenSTAAD.GetDOFReleasedAtEndOfMember 5, pnDOFs(0)

  Cells(10, 12).Value = pnDOFs(0)
  Cells(11, 12).Value = pnDOFs(1)
  Cells(12, 12).Value = pnDOFs(2)
  Cells(13, 12).Value = pnDOFs(3)
  Cells(14, 12).Value = pnDOFs(4)
  Cells(15, 12).Value = pnDOFs(5)

'Close the STAAD file and release the handles to OpenSTAAD.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberOffsets
DoesMemberHaveReleases
IsStartEndReleased
IsEndEndReleased
GetDOFReleasedAtStartOfMember

IsPartiallyReleasedAtStartOfMember

VB Syntax

integer IsPartiallyReleasedAtStartOfMember (integer nMemberNo, integer pnPartialRel)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to determine whether the member is partially released at its starting end.

pnPartialRel

An integer variable name passed to the function for it to use in storing its results. A value of 0 indicates that none of the rotational degrees of freedom have been partially released; a value of 1 indicates that one or more degrees of freedom have been partially released.

Remarks

Moments at the end of a member may be released partially, to model partial fixity of connections. This function determines whether any of the 3 rotational degrees of freedom Mx, My or Mz have been partially released at the starting end (end A) of a given member. The function stores the result as either a 0 or a 1 in an integer variable name passed to it as a parameter. A value of 0 indicates that none of the rotational degrees of freedom have been partially released at the starting end; a value of 1 indicates that one or more degrees of freedom have been partially released.

Note that Starting End and Ending End is based on the Member Incidence specification.

Example

```
Sub StartPartRel()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare integer variable for storing function results.

 Dim pnPartialRel As Integer
```

```
'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Determine whether Member 3 has a partial release at the starting end.

  objOpenSTAAD.IsPartiallyReleasedAtStartOfMember 3, pnPartialRel

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

IsPartiallyReleasedAtEndOfMember

IsPartiallyReleasedAtEndOfMember

VB Syntax

integer IsPartiallyReleasedAtEndOfMember (integer nMemberNo, integer pnPartialRel)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to determine whether the member is partially released at its end.

pnPartialRel

An integer variable name passed to the function for it to use in storing its results. A value of 0 indicates that none of the rotational degrees of freedom have been partially released; a value of 1 indicates that one or more degrees of freedom have been partially released.

Remarks

Moments at the end of a member may be released partially, to model partial fixity of connections. This function determines whether any of the 3 rotational degrees of freedom Mx, My or Mz have been partially released at the ending end (end B) of a given member. The first parameter passed to the function represents the member number for which the function is to determine whether the starting end is partially released. The function stores the result as either a 0 or a 1 in an integer variable name passed to it as the second parameter. A value of 0 indicates that none of the rotational degrees of freedom have been partially released at the ending end; a value of 1 indicates that one or more degrees of freedom have been partially released.

Note that Starting End and Ending End is based on the Member Incidence specification.

Example

```
Sub EndPartRel()

'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare integer variable for storing function results.

 Dim pnPartialRel As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Determine whether Member 3 has a partial release at the ending end.

  objOpenSTAAD.IsPartiallyReleasedAtEndOfMember 3, pnPartialRel

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

IsPartiallyReleasedAtStartOfMember

IsSpringReleaseAtStartOfMember

VB Syntax

integer IsSpringReleaseAtStartOfMember (integer nMemberNo, integer pnSpringRel)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to determine whether the member has a spring release at its starting end.

pnSpringRel

The name of an integer variable passed to the function for it to use in storing the value it returns. The function will return a value of 0 if there is no spring release. It will return a value of 1 if there is a spring release.

Remarks

This function checks a given member in the currently open STAAD file to determine if there is a spring release at the starting end of the member. The function will return a value of 0 if there is no spring release at the starting end (end A) of the member. It will return a value of 1 if there is a spring release at the starting end.

Note that Starting End (end A) and Ending End (end B) is based on the Member Incidence specification.

Example

```
Public Sub IsSpringRelAtStart()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare an integer variable for storing the function results.

 Dim pnSpringRel As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Determine whether there is any spring release at the starting end of Member 3.
```

```
  objOpenSTAAD.IsSpringReleaseAtStartOfMember 3, pnSpringRel

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

IsSpringReleaseAtEndOfMember
GetSpringReleaseStiffnessesAtStartOfMember
GetSpringReleaseStiffnessesAtEndOfMember

IsSpringReleaseAtEndOfMember

VB Syntax

integer IsSpringReleaseAtEndOfMember (integer nMemberNo, integer pnSpringRel)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to determine whether the member has a spring release at its end.

pnSpringRel

The name of an integer variable passed to the function for it to use in storing the value it returns. The function will return a value of 0 if there is no spring release. It will return a value of 1 if there is a spring release.

Remarks

This function checks a given member in the currently open STAAD file to determine if there is a spring release at the ending end (end B) of the member. The function will return a value of 0 if there is no spring release at the ending end (end B) of the member. It will return a value of 1 if there is a spring release at the ending end.

Note that Starting End and Ending End is based on the Member Incidence specification.

Example

```
Public Sub IsSpringRelAtEnd()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare an integer variable for storing the function results.

 Dim pnSpringRel As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Determine whether there is any spring release at the ending end (end B)
' of Member 3.
```

```
  objOpenSTAAD.IsSpringReleaseAtEndOfMember 3, pnSpringRel

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

IsSpringReleaseAtStartOfMember
GetSpringReleaseStiffnessesAtStartOfMember
GetSpringReleaseStiffnessesAtEndOfMember

GetSpringReleaseStiffnessesAtStartOfMember

VB Syntax

integer GetSpringReleaseStiffnessesAtStartOfMember (integer nMemberNo, double pdFactors)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to retrieve the spring release stiffness factors.

pdFactors

An array of 6 double (8-byte floating point) values for the function to use in storing the 6 stiffness factors KFX, KFY, KFZ, KMX, KMY, and KMZ.

Remarks

This function retrieves the spring release stiffness factors KFX, KFY, KFZ, KMX, KMY, and KMZ at the starting end (end A) of a given member.

The spring release stiffness values will be stored in the *pdFactors* array in the following order:

pdFactors (0) = KFX

pdFactors (1) = KFY

pdFactors (2) = KFZ

pdFactors (3) = KMX

pdFactors (4) = KMY

pdFactors (5) = KMZ

All values are given in units of kips and inches.

Note that Starting End and Ending End is based on the Member Incidence specification.

Example

```
Public Sub SpringStiff()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Object

'Declare a 6-value double array for storing the function results.

 Dim pdFactors(0 To 5) As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve spring release stiffness factors at the starting end of Member 3.

  objOpenSTAAD.GetSpringReleaseStiffnessesAtStartOfMember 3, pdFactors(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

IsSpringReleaseAtStartOfMember
IsSpringReleaseAtEndOfMember
GetSpringReleaseStiffnessesAtEndOfMember

GetSpringReleaseStiffnessesAtEndOfMember

VB Syntax

integer GetSpringReleaseStiffnessesAtEndOfMember (integer nMemberNo, double pdFactors)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to retrieve the spring release stiffness factors.

pdFactors

An array of 6 double (8-byte floating point) values for the function to use in storing the 6 stiffness factors KFX, KFY, KFZ, KMX, KMY, and KMZ.

Remarks

This function retrieves the spring release stiffness factors KFX, KFY, KFZ, KMX, KMY, and KMZ at the ending end (end B) of a given member.

The spring release stiffness values will be stored in the *pdFactors* array in the following order:

pdFactors (0) = KFX

pdFactors (1) = KFY

pdFactors (2) = KFZ

pdFactors (3) = KMX

pdFactors (4) = KMY

pdFactors (5) = KMZ

All values are given in units of kips and inches.

Note that Starting End and Ending End is based on the Member Incidence specification.

Example

```
Public Sub SpringStiff()

'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Object

'Declare an array of 6 double values for storing the function results.

 Dim pdFactors(0 To 5) As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve spring release stiffness factors at the ending end (end B) of
' Member 3.

  objOpenSTAAD.GetSpringReleaseStiffnessesAtEndOfMember 3, pdFactors(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

IsSpringReleaseAtStartOfMember
IsSpringReleaseAtEndOfMember
GetSpringReleaseStiffnessesAtStartOfMember

GetSupportStiffnesses

VB Syntax

integer GetSupportStiffnesses (integer nNodeNo, double pdStiffnesses)

Parameters

nNodeNo

An integer variable greater than 0 specifying the number of the node for which the function is to retrieve the support stiffness factors.

pdStiffnesses

An array of 6 double (8-byte floating point) values for the function to use in storing the 6 stiffness factors KFX, KFY, KFZ, KMX, KMY, and KMZ.

Remarks

This function retrieves the stiffness factors KFX, KFY, KFZ, KMX, KMY, and KMZ at a given supported node.

The support stiffness values will be stored in the *pdStiffnesses* array in the following order:

pdStiffnesses (0) = KFX

pdStiffnesses (1) = KFY

pdStiffnesses (2) = KFZ

pdStiffnesses (3) = KMX

pdStiffnesses (4) = KMY

pdStiffnesses (5) = KMZ

All values are given in units of kips and inches.

Note that Starting End and Ending End is based on the Member Incidence specification.

Example

```
Public Sub SuppStiff()

'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Object

'Declare an array of 6 double values for storing the function results.

 Dim pdStiffnesses(0 To 5) As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve stiffness factors for the support at Node 2.

  objOpenSTAAD.GetSupportStiffnesses 2, pdStiffnesses(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetSupportCondition
GetSupportReactions

GetFullMemberReleaseInfoAtStart

VB Syntax

integer GetFullMemberReleaseInfoAtStart (integer nMemberNo, integer pnIsReleased, double pdSpringStiffnesses, double pdPartialMomRelFactors)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to retrieve the release information.

pnIsReleased

An integer variable name for storing the results returned by the function. A value of 0 stored in *pnIsReleased* indicates that none of the degrees of freedom are released at the starting end of the member; a value of 1 indicates that one or more degrees of freedom are released.

pdSpringStiffnesses

An array of 6 double (8-byte floating point) values for the function to use in storing the 6 stiffness factors KFX, KFY, KFZ, KMX, KMY, and KMZ.

pdPartialMomRelFactors

An array of 3 double (8-byte floating point) values for the function to use in storing the partial moment release factors for the rotational degrees of freedom Mx, MY, and MZ respectively. The moment release factors will have a value between 0 and 1.

Remarks

This function checks the starting end (end A) of a given member to determine if any degrees of freedom are released. It also returns the stiffness factors for the 6 degrees of freedom, and the partial moment factors for the 3 rotational degrees of freedom.

Note that Starting End and Ending End is based on the Member Incidence specification.

Example

```
Sub GetFullInfo()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare variables for storing the function results.

 Dim pnIsReleased As Integer
 Dim pdSpringStiffnesses(0 To 5) As Double
 Dim pdPartialMomRelFactors(0 To 2) As Double


'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Get release information at starting end (end A) of Member 3.

'Note the use of the VB line continuation character, a space followed by an
' underscore in the following code, allowing a single code statement to
' be written on multiple lines.

    objOpenSTAAD.GetFullMemberReleaseInfoAtStart 3, _
        pnIsReleased, pdSpringStiffnesses(0), pdPartialMomRelFactors(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetFullMemberReleaseInfoAtEnd

GetFullMemberReleaseInfoAtEnd

VB Syntax

integer GetFullMemberReleaseInfoAtEnd (integer nMemberNo, integer pnIsReleased, double pdSpringStiffnesses, double pdPartialMomRelFactors)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to retrieve the release information.

pnIsReleased

An integer variable name for storing the results returned by the function. A value of 0 stored in *pnIsReleased* indicates that none of the degrees of freedom are released at the ending end of the member; a value of 1 indicates that one or more degrees of freedom are released.

pdSpringStiffnesses

An array of 6 double (8-byte floating point) values for the function to use in storing the 6 stiffness factors KFX, KFY, KFZ, KMX, KMY, and KMZ.

pdPartialMomRelFactors

An array of 3 double (8-byte floating point) values for the function to use in storing the partial moment release factors for the rotational degrees of freedom Mx, MY, and MZ respectively. The moment release factors will have a value between 0 and 1.

Remarks

This function checks the ending end (end B) of a given member to determine if any degrees of freedom are released. It also returns the stiffness factors for the 6 degrees of freedom, and the partial moment factors for the 3 rotational degrees of freedom.

Note that Starting End and Ending End is based on the Member Incidence specification.

Example

```
Sub GetFullInfo()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare variables for storing the function results.

 Dim pnIsReleased As Integer
 Dim pdSpringStiffnesses(0 To 5) As Double
 Dim pdPartialMomRelFactors(0 To 2) As Double


'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Get release information at ending end (end B) of Member 3.

'Note the use of the VB line continuation character, a space followed by an
' underscore in the following code, allowing a single code statement to
' be written on multiple lines.


  objOpenSTAAD.GetFullMemberReleaseInfoAtEnd 3, _
        pnIsReleased, pdSpringStiffnesses(0), pdPartialMomRelFactors(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetFullMemberReleaseInfoAtStart

GetMemberBetaAngle

VB Syntax

integer GetMemberBetaAngle (integer nMemberNo, double pdBeta)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to retrieve the beta angle.

pdBeta

The name of a double (8-byte floating point) variable for the function to use in storing the beta angle it retrieves.

Remarks

This function retrieves the beta angle for a given member in the currently open STAAD file. The member number and variable name for storing the results are passed to the function as parameters.

Example

```
Public Sub BetaAngle()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare a double variable for storing the function results.

 Dim pdBeta As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Determine the beta angle of Member 3.

  objOpenSTAAD.GetMemberBetaAngle 3, pdBeta

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberLength
GetMemberWidthAndDepth
GetMemberProperties
GetMemberPropsForPrismatic
GetMemberPropertyShape

GetMemberSteelDesignRatio

VB Syntax

integer GetMemberSteelDesignRatio (integer nMemberNo, double pdRatio)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to retrieve the steel design ratio.

pdRatio

The name of a double (8-byte floating point) variable for the function to use in storing the steel design ratio it retrieves.

Remarks

This function retrieves the steel design ratio for a given member in the currently open STAAD file. The member number and variable name for storing the results are passed to the function as parameters.

Example

```
Public Sub SteelRatio()

'This is a VB for Excel macro.

'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare a double variable for storing the function results.

 Dim pdRatio As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve Steel Design Ratio for Member 3.

  objOpenSTAAD.GetMemberSteelDesignRatio 3, pdRatio

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing
```

```
End Sub
```

See Also

GetMemberDesignProperties

IsMemberACableMember

VB Syntax

IsMemberACableMember (integer nMemberNo, integer pnIsCable, double pdTension)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to determine whether the member is a cable member.

pnIsCable

The name of an integer variable passed to the function for it to use in storing the value it returns. The function will return a value of 0 if the member is not a cable member. It will return a value of 1 if the member is a cable member.

pdTension

The name of a double (8-byte floating point) variable for the function to use in storing the tension force in the member it retrieves from STAAD.*Pro*.

Remarks

This function determines whether a given member in the currently open STAAD file is a cable member. The member number and variable names for storing the results are passed to the function as parameters.

If the member is a cable member, the function returns an integer value of 1 and stores it in the *pnIsCable* integer variable. It also retrieves the value of the tension force applied to the cable member and stores the value in the *pdTension* double variable.

If the member is not a cable member, the function returns a value of 0 and stores it in the *pnIsCable* integer variable. It also stores a value of 0 in the *pdTension* double variable.

All values are given in units of kips and inches

Example

```
Public Sub IsMemCable()

'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare variables for storing the function results.

Dim pnIsCable as Integer
Dim pdTension As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Detemine whether Member 3 is a cable member and return the tension force.

  objOpenSTAAD.IsMemberACableMember 3, pnIsCable, pdTension

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

IsMemberACompressionMember
IsMemberATensionMember
IsMemberATrussMember

IsMemberACompressionMember

VB Syntax

IsMemberACompressionMember (integer nMemberNo, integer pnIsCompression)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to determine whether the member is a compression member.

pnIsCompression

The name of an integer variable passed to the function for it to use in storing the value it returns (either a 0 or a 1).

Remarks

This function determines whether a given member in the currently open STAAD file is specified as a compression-only member. The member number and variable name for storing the results are passed to the function as parameters.

If the member is a compression-only member, the function returns an integer value of 1 and stores it in the *pnIsCompression* integer variable.

If the member is not a compression-only member, the function returns a value of 0 and stores it in the *pnIsCompression* integer variable.

Example

```
Public Sub IsMemCompOnly()

'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare an integer variable for storing the function results.

 Dim pnIsCompression As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Determine whether Member 3 is specified as a compression-only member.
```

```
  objOpenSTAAD.IsMemberACompressionMember 3, pnIsCompression

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

IsMemberACableMember
IsMemberATensionMember
IsMemberATrussMember

IsMemberATensionMember

VB Syntax

IsMemberATensionMember (integer nMemberNo, integer pnIsTension)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to determine whether the member is a tension-only member.

pnIsTension

The name of an integer variable passed to the function for it to use in storing the value it returns (either a 0 or a 1).

Remarks

This function determines whether a given member in the currently open STAAD file is specified as a tension-only member. The member number and variable name for storing the results are passed to the function as parameters.

If the member is a tension-only member, the function returns an integer value of 1 and stores it in the *pnIsTension* integer variable.

If the member is not a tension-only member, the function returns a value of 0 and stores it in the *pnIsTension* integer variable.

Example

```
Public Sub IsMemTensionOnly()

'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare an integer variable for storing the function results.

 Dim pnIsTension As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Determine whether Member 3 is specified as a tension-only member.
```

```
  objOpenSTAAD.IsMemberATensionMember 3, pnIsTension

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

IsMemberACableMember
IsMemberACompressionMember
IsMemberATrussMember

IsMemberATrussMember

VB Syntax

IsMemberATrussMember (integer nMemberNo, integer pnIsTruss)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to determine whether the member is a truss member.

pnIsTruss

The name of an integer variable passed to the function for it to use in storing the value it returns (either a 0 or a 1).

Remarks

This function determines whether a given member in the currently open STAAD file is specified as a truss member. The member number and variable name for storing the results are passed to the function as parameters.

If the member is a truss member, the function returns an integer value of 1 and stores it in the *pnIsTruss* integer variable.

If the member is not a truss member, the function returns a value of 0 and stores it in the *pnIsTruss* integer variable.

Example

```
Sub IsMemTruss

'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare an integer variable for storing the function results.

 Dim pnIsTruss As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Determine whether Member 3 is specified as a truss member.

  objOpenSTAAD.IsMemberATrussMember 3, pnIsTruss
```

```
'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

IsMemberACableMember
IsMemberACompressionMember
IsMemberATensionMember

Properties Functions

GetMemberLength

VB Syntax

integer GetMemberLength (integer nMemberNo, double pdLength)

Parameters

nMemberNo

An integer greater than or equal to one which specifies the member number for which the member length is to be obtained.

pdLength

A double (8-byte floating point) variable name passed to the function for it to use in storing the member length it retrieves.

Remarks

Given the member number, this function retrieves the length of the member.

All values are given in units of kips and inches.

Example

```
Sub HowLong()

'Declare an OpenSTAAD object variable As Output.

 Dim objOpenSTAAD

'Declare a double variable for storing the function results.

 Dim pdLength As Double

'Run an instance of OpenSTAAD and open Example 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\Examp01.std"

'Retrieve the length for Member No. 1 and store the value in the
' pdLength variable.

  objOpenSTAAD.GetMemberLength 1, pdLength
```

```
'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberBetaAngle
GetMemberWidthAndDepth
GetMemberProperties
GetMemberSteelDesignRatio
GetMemberMaterialConstants

GetMemberWidthAndDepth

VB Syntax

integer GetMemberWidthAndDepth (integer nMemberNo, double pdWidth, double pdDepth)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to retrieve the width and depth.

pdWidth

A double (8 byte floating point) variable for the function to use to store the member width.

pdDepth

A double (8 byte floating point) variable for the function to use to store the member depth.

Remarks

This function retrieves the width and depth for a given member in the currently open STAAD model.

All values are given in units of kips and inches.

Example

```
Sub WidthDepth()

 Dim objOpenSTAAD
 Dim pdWidth As Double
 Dim pdDepth As Double

'Run an instance of OpenSTAAD and open Example 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\Examp01.std"

'Retrieve the width and depth for Member No. 3 and store the values in
' the pdWidth and pdDepth variables.

  objOpenSTAAD.GetMemberWidthAndDepth 3, pdWidth, pdDepth

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberBetaAngle
GetMemberLength
GetMemberProperties
GetMemberSteelDesignRatio
GetMemberMaterialConstants

GetMemberProperties

VB Syntax

integer GetMemberProperties (integer nMemberNo, double pdWidth, double pdDepth, double pdAX, double pdAY, double pdAZ, double pdIX, double pdIY, double pdIZ)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to retrieve the properties.

pdWidth

A double (8 byte floating point) variable for the function to use to store the member width.

pdDepth

A double (8 byte floating point) variable for the function to use to store the member depth.

pdAX

A double (8 byte floating point) variable for the function to use to store the cross sectional area of the member.

pdAY

A double (8 byte floating point) variable for the function to use to store the member's effective shear area in the local y axis.

pdAZ

A double (8 byte floating point) variable for the function to use to store the member's effective shear area in the local z axis.

pdIX

A double (8 byte floating point) variable for the function to use to store the member's torsional constant.

pdIY

A double (8 byte floating point) variable for the function to use to store the member's moment of inertia about the local y axis.

pdIZ

A double (8 byte floating point) variable for the function to use to store the member's moment of inertia about the local z axis.

Remarks

This function retrieves the properties of a given member in the currently open STAAD file. The member number is passed to the function, along with 8 variable names for storing the results returned by the function: member width, depth, cross sectional area, effective shear areas in the y and z axes, torsional constant and moments of inertia about the y and z axes.

All values are given in units of kips and inches.

Example

```
Sub MemProps()

 Dim objOpenSTAAD
 Dim pdWidth As Double
 Dim pdDepth As Double
 Dim pdAX As Double
 Dim pdAY As Double
 Dim pdAZ As Double
 Dim pdIX As Double
 Dim pdIY As Double
 Dim pdIZ As Double

'Run an instance of OpenSTAAD and open Example 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\Examp01.std"

'Retrieve the properties for Member No. 3 and store the values in the
' pdWidth, pdDepth, pdAX, pdAY, pdAZ, pdIX, pdIY, pdIZ variables.

'Note the use of the VB line continuation character, a space followed by an
' underscore in the following code, allowing a single statement to be written
' on multiple lines.

  objOpenSTAAD.GetMemberProperties 3, pdWidth, pdDepth, _
                                   pdAX, pdAY, pdAZ, pdIX, pdIY, pdIZ

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing
```

```
End Sub
```

See Also

GetMemberBetaAngle
GetMemberLength
GetMemberWidthAndDepth
GetMemberSteelDesignRatio
GetMemberMaterialConstants
GetMemberPropertiesForPrismatic
GetMemberPropertyShape

GetMemberPropsForPrismatic

VB Syntax

GetMemberPropsForPrismatic (integer nMemberNo, integer pnShape, double pdYD, double pdZD, double pdYB, double pdZB)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to retrieve the member properties.

pnShape

An integer variable name for the function to use in storing the property shape it retrieves from STAAD.*Pro*.

pdYD

A double (8 byte floating point) variable for the function to use to store the value of the depth of the section parallel to the local y-axis it retrieves from STAAD.*Pro*.

pdZD

A double (8 byte floating point) variable for the function to use to store the value of the depth of the section parallel to the local z-axis it retrieves from STAAD.*Pro*.

pdYB

A double (8 byte floating point) variable for the function to use to store the value of the depth of web of T-section it retrieves from STAAD.*Pro*.

pdZB

A double (8 byte floating point) variable for the function to use to store the value of the width of web of T-section or bottom width of trapezoidal section it retrieves from STAAD.*Pro*.

Remarks

This function retrieves the property shape for a given prismatic member in the currently open STAAD file. The member number and variable names for storing the function results are passed to the function as parameters. The function returns an integer number between 0 and 11 corresponding to the shape of the given member and stores it in the *pnShape* variable.

0	Member shape is not defined
1	Shape is from steel table
2	Shape is from user-provided table
3	Rectangle shape
4	Circle shape
5	Tee shape
6	Trapezoid shape
7	General shape
8	Tube shape
9	Pipe shape
10	Tapered section
11	Shape property specified with Assign Profile option

The function also returns the depth of the section parallel to the local y-axis, the depth of the section parallel to the local z-axis, the depth of web if the shape is a T-section, and the width of web if the shape is a T-section, or the bottom width if the shape is a trapezoidal section.

Example

```
Sub Props4Prism()

'Declare OpenSTAAD object as Output.

 Dim objOpenSTAAD As Output

'Declare variables for storing the function results.

 Dim pnShape As Integer
 Dim pdYD As Double
 Dim pdZD As Double
 Dim pdYB As Double
 Dim pdYB As Double

'Run an instance of OpenSTAAD and open Example 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\Examp01.std"

'Retrieve the member properties for Member No. 3.

  objOpenSTAAD.GetMemberPropsForPrismatic 3, pnShape, pdYD, pdZD, pdYB, pdZB

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
```

```
  Set objOpenSTAAD = Nothing
End Sub
```

See Also

GetMemberPropertyShape

GetMemberDesignProperties

VB Syntax

integer GetMemberDesignProperties (integer nMemberNo, string strSectionName, double pdWidth, double pdDepth, double pdAX, double pdAY, double pdAZ, double pdIX, double pdIY, double pdIZ)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to retrieve the properties.

strSectionName

A string variable for the function to use in storing the member property name it retrieves from STAAD.*Pro*, e.g. “W10x68”.

pdWidth

A double (8 byte floating point) variable for the function to use to store the member width.

pdDepth

A double (8 byte floating point) variable for the function to use to store the member depth.

pdAX

A double (8 byte floating point) variable for the function to use to store the cross sectional area of the member.

pdAY

A double (8 byte floating point) variable for the function to use to store the member’s effective shear area in the local y axis.

pdAZ

A double (8 byte floating point) variable for the function to use to store the member's effective shear area in the local z axis.

pdIX

A double (8 byte floating point) variable for the function to use to store the member's torsional constant.

pdIY

A double (8 byte floating point) variable for the function to use to store the member's moment of inertia about the local y axis.

pdIZ

A double (8 byte floating point) variable for the function to use to store the member's moment of inertia about the local z axis.

Remarks

This function retrieves design properties for a given member, including:

- Width and depth
- Cross sectional area
- Effective shear areas in local y and z axes
- Torsional constant
- Moments of inertia about local y and z axes

Example

```
Sub MemProps()

 Dim objOpenSTAAD
 Dim pdWidth As Double
 Dim pdDepth As Double
 Dim pdAX As Double
 Dim pdAY As Double
 Dim pdAZ As Double
 Dim pdIX As Double
 Dim pdIY As Double
 Dim pdIZ As Double
 Dim pstrPropName As String

'Run an instance of OpenSTAAD and open Example 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\Examp01.std"

'Retrieve the properties for Member No. 3 and store the values in the
'pstrPropName, pdWidth, pdDepth, pdAX, pdAY, pdAZ, pdIX, pdIY, pdIZ variables.


'Note the use of the VB line continuation character, a space followed by an
```

```
' underscore in the following code, allowing a single statement to be written
' on multiple lines.

  objOpenSTAAD.GetMemberProperties 3, pstrPropName, pdWidth, pdDepth, _
  pdAX, pdAY, pdAZ, pdIX, pdIY, pdIZ

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberSteelDesignRatio
GetMemberMaterialConstants

GetSteelTableProperties

VB Syntax

GetSteelTableProperties (integer nCountry, string strSectionName, integer pnCrossSection, double pdProperties)

Parameters

nCountry

An integer variable between 1 and 17 passed to the function to specify the steel table from which the properties are to be retrieved:

1	American	10	Indian
2	Australian	11	Japanese
3	British	12	Russian
4	Canadian	13	South African
5	Chinese	14	Spanish
6	Dutch	15	Venezuelan
7	European	16	Korean
8	French	17	Aluminum
9	German		

strSectionName

A string variable passed to the function specifying the section name for which the steel properties are to be retrieved, e.g. “W12x96”. The function will ignore spaces in the string, so, for example, you could also type “W 12 x 96”.

pnCrossSection

An integer variable name for the function to use in storing an integer value corresponding to the cross section type it retrieves from STAAD.Pro. The integer values and cross section types vary, depending on which country’s steel table is being used. The following tables identify the integers and corresponding cross section types for each country’s steel table.

American Sections

1	W Shape	2	M Shape
3	S Shape	4	HP Shape
5	B Shape	6	Channel
7	MCChannel	8	Angle
9	Tube	10	Pipe

Australian Sections

1	UBShape	2	UCShape
3	WBShape	4	WCShape
5	Channel	6	Angle

British Sections

1	UB Shape	2	UC Shape
3	UP Shape	4	JO Shape
5	Channel	6	Angle
7	Tube	8	Pipe

Canadian Sections

1	W Shape	2	M Shape
3	S Shape	4	HP Shape
5	WW Shape	6	Channel
7	MC Channel	8	Angle
9	Tube	10	Pipe

Chinese Sections

1	I Shape	2	Channel
3	Angle	4	Tube
5	Pipe		

Dutch Sections

1	HE Shape	2	IPE Shape
3	INP Shape	4	UNP Channel
5	Angle	6	Tube
7	Pipe	8	Plate Strip
9	Solid Round	10	Solid Square

European Sections

1	IPE Shape	2	HE Shape
3	DIL Shape	4	IPN Shape
5	U Channel	6	UNP Channel
7	Angle	8	Tube
9	Pipe		

French Sections

1	IPE Shape	2	HE Shape
3	IPN Shape	4	Channel
5	Angle	6	Tube
7	Pipe		

German Sections

1	IPE Shape	2	HE Shape
3	I Shape	4	U Channel
5	Angle	6	Tube
7	Pipe		

Indian Sections

1	S Shape	2	I Shape
3	M Shape	4	W Shape
5	TShape	6	Channel
7	Angle	8	Tube
9	Pipe		

Japanese Sections

1	H Shape	2	I Shape
3	T Shape	4	Channel
5	Angle	6	Tube
7	Pipe		

Russian Sections

1	B Shape	2	SH Shape
3	K Shape	4	I Shape
5	Channel	6	Angle
7	Tube	8	Pipe

South African Sections

1	I Shape	2	H Shape
3	PG Shape	4	C Channel
5	Angle	6	Tube
7	Pipe		

Spanish Sections

1	IPE Shape	2	HE Shape
3	IPN Shape	4	Channel
5	Angle	6	Tube
7	Pipe		

Venezuelan Sections

1	Tube	2	Pipe

Korean Sections

1	W Shape	2	WT Shape

3	Channel	4	Angle
5	Pipe		

Aluminum Sections

1	AA Standard I Beams	2	H Beam
3	Army Navy I Beam	4	American Standard I Beam
5	I Beam	6	AA Standard Channel
7	Channel	8	Army Navy Channel
9	Special Channel	10	American Standard Channel
11	Equal Leg Angle	12	Square End Equal Leg Angle
13	Unequal Leg Angle	14	Square End Unequal Leg Angle
15	Square Tube	16	Rectangular Tube
17	RoundTube	18	Pipe
19	Army Navy Tee	20	Tee

pdProperties

An array of 8 double (8-byte floating point) values for the function to use in storing the member properties it retrieves from the STAAD.*Pro* steel tables. The 8 properties that the function retrieves are the cross sectional area (AX), the effective shear area for shear force parallel to the local y-axis (AY), the effective shear area for shear force parallel to the local z-axis (AZ), the member depth, the torsional constant (IX), the moment of inertia about the y-axis (IY), the moment of inertia about the z-axis (IZ) and the member width.

Remarks

This function returns the member type and properties of members in the steel tables. An integer value corresponding to the desired steel table is passed to the function as the first parameter. A string parameter specifying the section name for which the steel properties are to be retrieved, e.g. “W12x96”, is also passed to the function. The function then returns the cross section type and member properties and stores them in the *pnCrossSection* and the *pdProperties* variables respectively.

The member property values will be stored in the *pdProperties* array in the following order:

```
1) pdProperties (0) = AX
2) pdProperties (1) = AY
3) pdProperties (2) = AZ
4) pdProperties (3) = depth
5) pdProperties (4) = IX
6) pdProperties (5) = IY
7) pdProperties (6) = IZ
8) pdProperties (7) = width
```

All values are given in units of kips and inches.

Example

```
Sub SteelTableShape()

'Declare OpenSTAAD object as Output.

 Dim objOpenSTAAD As Output

'Declare variables for storing the function results.

 Dim  As Integer
 Dim pnCrossSection As Integer
 Dim pdProperties(0 To 7) As Double

'Run an instance of OpenSTAAD and open Example 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\Examp01.std"

'Retrieve the member type and properties for a W12x96 section from the
' American steel table.
'Note the use of the VB line continuation character, a space followed by an
' underscore in the following code, allowing a single code statement to
' be written on multiple lines.

  objOpenSTAAD.GetSteelTableProperties 1, string “W12X96”, _
                                       pnCrossSection, pdProperties(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberDesignProperties
GetMemberPropertyShape

GetMemberPropertyShape

VB Syntax

GetMemberPropertyShape (integer nMemberNo, integer pnShape)

Parameters

nMemberNo

An integer greater than or equal to one representing the member number of the member for which the property shape is to be obtained.

pnShape

An integer variable name for the function to use in storing the property shape it retrieves from STAAD.*Pro*.

Remarks

This function retrieves the property shape for a given member in the currently open STAAD file. The member number and variable name for storing the function results are passed to the function as parameters. The function returns an integer number between 0 and 11 corresponding to the shape of the given member.

0	Member shape is not defined
1	Shape is from steel table
2	Shape is from user-provided table
3	Rectangle shape
4	Circle shape
5	Tee shape
6	Trapezoid shape
7	General shape
8	Tube shape
9	Pipe shape
10	Tapered section
11	Shape property specified with Assign Profile option

Example

```
Sub MemPropShape()

'Declare OpenSTAAD object as Output.

 Dim objOpenSTAAD As Output

'Declare an integer variable for storing the function results.

 Dim pnShape As Integer

'Run an instance of OpenSTAAD and open Example 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\Examp01.std"

'Retrieve the member shape for Member No. 3.

  objOpenSTAAD.GetMemberPropertyShape 3, pnShape

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberPropsForPrismatic
GetSteelTableProperties

GetFinalMemberPropertyName

VB Syntax

GetFinalMemberPropertyName (integer nMemberNo, string pstrPropName)

Parameters

nMemberNo

An integer greater than or equal to one representing the member number for which the final member property name is to be obtained.

pstrPropName

A string variable for the function to use in storing the member property name it retrieves from STAAD.*Pro*, e.g. “W10x68”.

Remarks

This function retrieves the final member property name for a given member in the currently open STAAD file. The member number is passed to the function as a parameter. The function then returns a string that represents the member selected for the final design, e.g. “W10x68”. If only an analysis or code check was performed by the STAAD analysis, this function will return the member name specified with STAAD’s Properties commands. If the analysis included member selection based on parameter value restrictions and specified code, or based on STAAD’s member optimization routine, this function will return the final member selected as a result of the member selection.

Example

```
Sub FinalPropName()

'Declare OpenSTAAD object as Output.

 Dim objOpenSTAAD As Output

'Declare a string variable for storing the function results.

 Dim pstrPropName As String

'Run an instance of OpenSTAAD and open Example 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\Examp01.std"

'Retrieve the final member name for Member No. 3.
```

```
  objOpenSTAAD. GetFinalMemberPropertyName 3, pstrPropName

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberPropsForPrismatic
GetSteelTableProperties
GetMemberDesignProperties
GetMemberPropertyShape

GetCompositeSectionParameters

VB Syntax

GetCompositeSectionParameters (long nMemberNo, double pdFc, double pdThickness, double pdWidth)

Parameters

nMemberNo

A long integer greater than or equal to one representing the member number for which the composite section parameters are to be obtained.

pdFc

A double variable for the function to use in storing the strength of concrete.

pdThickness

A double variable for the function to use in storing the thickness of the concrete slab.

pdWidth

A double variable for the function to use in storing the width of the concrete slab.

Remarks

This function retrieves the strength, width and thickness of the conrete slab used as a part of a composite section with a wide flange. The member number is passed to the function as a parameter. If the section is not a composite section, the strength, thickness and width of the slab will be set to zero.

Example

```
Sub CompositeParameters()

'Declare OpenSTAAD object as Output.
 Dim objOpenSTAAD As Output

'Declare variables to store the composite parameters
 Dim Fc As Double
 Dim Thickness As Double
 Dim Width As Double

'Run an instance of OpenSTAAD and open a problem with a composite section.
```

```
  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\My Folder\Examp.std"

'Retrieve the final member name for Member No. 3.

  objOpenSTAAD. GetCompositeSectionParameters 3, Fc, Thickness, Width

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberPropsForPrismatic
GetSteelTableProperties
GetMemberDesignProperties
GetMemberPropertyShape

GetMemberMaterialConstants

VB Syntax

GetMemberMaterialConstants (int nMemberNo, double pdE, double pdPoisson, double pdDensity, double pdAlpha, double pdDampingCoef)

Parameters

nMemberNo

An integer greater than or equal to one representing the member number for which the material constants are to be obtained.

pdE

A double (8 byte floating point) variable for the function to use to store the value of the member's coefficient of elasticity (E).

pdPoisson

A double (8 byte floating point) variable for the function to use to store the value of the member's Poisson's Ratio.

pdDensity

A double (8 byte floating point) variable for the function to use to store the weight density of the member.

pdAlpha

A double (8 byte floating point) variable for the function to use to store the Alpha value of the member.

pdDampingCoef

A double (8 byte floating point) variable for the function to use to store the member's damping coefficient.

Remarks

This function retrieves a given member's material constants including:

- Coefficient of Elasticity, E (a.k.a. Young's Modulus)
- Poisson's Ratio
- Density
- Alpha (Coefficient of Thermal Expansion)
- Damping Coefficient

Example

```
Sub MatConsts()

'Declare an OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare 5 double variables for storing the function results.

 Dim pdE As Double
 Dim pdPoisson As Double
 Dim pdDensity As Double
 Dim pdAlpha As Double
 Dim pdDampingCoef As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve material constants for member no. 3

'Note the use of the VB line continuation character, a space followed by an
' underscore in the following code, allowing a single code statement to
' be written on multiple lines.

objOpenSTAAD.GetMemberMaterialConstants 3, _
                          pdE, pdPoisson, pdDensity, pdAlpha, pdDampingCoef

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberBetaAngle
GetMemberLength
GetMemberWidthAndDepth
GetMemberSteelDesignRatio
GetMemberProperties

GetAreaOfPlate

VB Syntax

integer GetAreaOfPlate (integer nPlateNo, double pdArea)

Parameters

nPlateNo

An integer value greater than zero specifying the number of the plate element for which the area is to be obtained.

pdArea

A double (8-byte floating point) variable name passed to the function for it to use in storing the plate area.

Remarks

This function retrieves the area of a given plate in the currently open STAAD file.

All values are given in units of kips and inches.

Example

```
Sub PlateArea()

'Declare an OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare a double variable for storing the function results.

 Dim pdArea As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 23 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp23.std"

'Retrieve the area of plate 82 and store the value in the pdArea variable.

  objOpenSTAAD.GetAreaOfPlate 82, pdArea

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetAllPlateCenterMoments
GetAllPlateCenterPrincipalStressesAndAngles
GetAllPlateCenterForces
GetPlateCenterNormalPrincipalStresses
GetAllPlateCenterStressesAndMoments
GetPlateThicknesses

GetPlateThicknesses

VB Syntax

GetPlateThicknesses (integer nPlateNo, double pdThick1, double pdThick2, double pdThick3, double pdThick4, integer pnSameThickness)

Parameters

nPlateNo

An integer value greater than zero specifying the number of the plate element for which the plate thickness values are to be retrieved.

pdThick1

A double (8-byte floating point) variable for the function to use in storing the plate thickness at node 1 retrieved from STAAD.*Pro*.

pdThick2

A double (8-byte floating point) variable for the function to use in storing the plate thickness at node 2 retrieved from STAAD.*Pro*.

pdThick3

A double (8-byte floating point) variable for the function to use in storing the plate thickness at node 3 retrieved from STAAD.*Pro*.

pdThick4

A double (8-byte floating point) variable for the function to use in storing the plate thickness at node 4 retrieved from STAAD.*Pro*.

pnSameThickness

An integer variable for the function to use in storing the plate thickness results it retrieves from STAAD.Pro. If the plate is the same thickness at all 4 nodes, this function will return a 1 and store it in the *pnSameThickness* variable. If the plate is not the same thickness at all 4 nodes, this function will return a 0 and store it in the *pnSameThickness* variable.

Remarks

This function retrieves the thicknesses at all 4 nodes of a given plate in the currently open STAAD file. The function also returns an integer value indicating whether the thickness is the same for all 4 nodes.

The plate number and variable names for storing the function results are passed to the function as parameters. The function then returns the thickness at each of the plate's four nodes. In addition, if the plate is the same thickness at all 4 nodes, the function will return a 1 and store it in the *pnSameThickness* variable. If the plate is not the same thickness at all 4 nodes, the function will return a 0 and store it in the *pnSameThickness* variable.

All values are given in units of kips and inches.

Example

```
Sub PlateThick()

'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare variables for storing the function results.

 Dim pdThick1 As Double
 Dim pdThick2 As Double
 Dim pdThick3 As Double
 Dim pdThick4 As Double
 Dim pnSameThickness As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 10 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp10.std"

'Retrieve the plate thicknesses for plate no. 39.
'Note the use of the VB line continuation character, a space followed by an
' underscore in the following code, allowing a single code statement to
' be written on multiple lines.

  objOpenSTAAD.GetPlateThicknesses 39, pdThick1, pdThick2, _
                                   pdThick3, pdThick4, pnSameThickness

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing


End Sub
```

See Also

GetAllPlateCenterMoments
GetAllPlateCenterPrincipalStressesAndAngles
GetAllPlateCenterForces

GetPlateCenterNormalPrincipalStresses
GetAllPlateCenterStressesAndMoments
GetAreaOfPlate

Loads Functions

GetLoadCombinationCaseCount

VB Syntax

integer GetLoadCombinationCaseCount (integer pnCount)

Parameters

pnCount

An integer variable which the function will use to store the number of load combination cases.

Remarks

This function retrieves the number of load combination cases in the currently open STAAD file. It then stores that number in an integer variable passed to it as a parameter.

Example

```
Sub CountLCombs()

'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare an integer variable for storing the function results.

 Dim pnCount As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve the number of load combinations and store that number in the
' pnCount variable.

  objOpenSTAAD.GetLoadCombinationCaseCount pnCount

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing


End Sub
```

See Also

GetPrimaryLoadCaseCount

GetPrimaryLoadCaseCount

VB Syntax

integer GetPrimaryLoadCaseCount (integer pnCount)

Parameters

pnCount

An integer variable which the function will use to store the number of primary load cases.

Remarks

This function retrieves the number of primary load cases in the currently open STAAD file. It then stores that number in an integer variable passed to it as a parameter.

Example

```
Sub CountLCs()

 Dim objOpenSTAAD As Output
 Dim pnCount As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve the number of primary load cases and store that number in the pnCount '
variable.

  objOpenSTAAD.GetPrimaryLoadCaseCount pnCount

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetLoadCombinationCaseCount

GetFirstLoadCase

VB Syntax

GetFirstLoadCase (integer pnLoad, string pstrLoadName)

Parameters

pnLoad

An integer variable for the function to use in storing the load case number it retrieves from STAAD.*Pro*.

pstrLoadName

A string variable for the function to use in storing the load case name it retrieves from STAAD.*Pro*.

Remarks

This function retrieves the first load case number and corresponding load case name for the currently open STAAD file. The function stores the first load case number and corresponding load case name in variables passed to it as parameters. The program stores the first load case number in the *pnLoad* integer variable and the corresponding load case name in the *pstrLoadName* string variable.

This function should always be used before the *GetNextLoadCase* function is called to determine the first load case number (load cases are not necessarily numbered consecutively, and the first load case number may not always be number 1).

Example

```
Sub FirstLC()

'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare two variables for storing the function results.

 Dim pnLoad As Integer
 Dim pstrLoadName As String

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve the first load case number and name.

  objOpenSTAAD.GetFirstLoadCase pnLoad, pstrLoadName
```

```
'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetNextLoadCase

GetNextLoadCase

VB Syntax

GetNextLoadCase (integer nPrevLoadCase, integer pnLoad, string strLoadName)

Parameters

nPrevLoadCase

An integer value passed to the function to specify the load case number which directly precedes in numerical order the load case number and name we wish to retrieve.

pnLoad

An integer variable name passed to the function for it to use in storing the load case number it retrieves from STAAD.*Pro*.

strLoadName

A string variable name passed to the function for it to use in storing the load case name it retrieves from STAAD.*Pro*.

Remarks

This function retrieves the next load case number and load case name when given the previous load case number. Before this function is called, the *GetFirstLoadCase* function should be used to obtain the load case number for the first load case. This function can then be used to obtain the load case number and name for successive load cases.

Example

```
Sub getLCs()

'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare variables for storing the function results.

 Dim pnPriCount As Integer
 Dim pnCombCount As Integer
 Dim pnCount As Integer
 Dim pnLoad1 As Integer
 Dim pnLoad As Integer
 Dim nPrevLoadCase As Integer
```

```
 Dim pstrLoadName As String

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve the number of primary load cases and store that number in the
' pnPriCount variable.

  objOpenSTAAD.GetPrimaryLoadCaseCount pnPriCount

'Retrieve the number of load combinations and store that number in the
' pnCombCount variable.

  objOpenSTAAD.GetLoadCombinationCaseCount pnCombCount

'Calculate the total number of load cases.

 pnCount = pnPriCount + pnCombCount

'Retrieve the first load case number and name.

  objOpenSTAAD.GetFirstLoadCase pnLoad1, pstrLoadName

'Set the previous load case number for the GetNextLoadCase function
' equal to the first case load number.

  nPrevLoadCase = pnLoad1

'Iterate through the load cases and plot their load case numbers and names
' in an Excel worksheet.

 For i = 1 to pnCount

    objOpenSTAAD.GetNextLoadCase nPrevLoadCase, pnLoad, strLoadName

     Cells(31 + i, 11).Value = pnLoad
     Cells(31 + i, 12).Value = strLoadName

  'Set the previous load case number equal to the load case the function
  ' just retrieved, so we can go on to the next load case.

     nPrevLoadCase = pnLoad

 Next i

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetFirstLoadCase

Output Results Functions: Nodes

GetNodeDisplacements

VB Syntax

integer GetNodeDisplacements (integer nNodeNo, integer nLC, double pdDisps)

Parameters

nNodeNo

An integer value greater than 0 specifying the number of the node for which the function is to retrieve the displacement.

nLC

An integer value greater than 0 specifying the load case number for which the function is to retrieve the node displacement.

pdDisps

An array of 6 double (8-byte floating-point) values for the function to use to store the value of the node displacements it retrieves.

Remarks

This function retrieves the node displacement for a given node and load case. It returns 6 values (X, Y, Z, rX, rY and rZ) for the translational and rotational displacement and stores them in an array variable name passed to it as a parameter.

The node displacement values will be stored in the *pdDisps* array in the following order:

pdDisps (0) = X

pdDisps (1) = Y

pdDisps (2) = Z

pdDisps (3) = rX

pdDisps (4) = rY

pdDisps (5) = rZ

All values are given in units of kips and inches.

Example

```
Public Sub NodeDispl()

'This is a VBA for Excel macro.
'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare variables for use in passing the Node Number and
' Load Case Number to the function.

 Dim nNodeNo As Integer
 Dim nLC As Integer

'Declare an array of 6 Double values for storing the function results.

 Dim pdDisps(6) As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Get the Node Number and Load Case Number from the Excel worksheet.

 nNodeNo = Cells(30, 11).Value
 nLC = Cells(31, 11).Value

'Retrieve the displacement values and store them in the array.

 objOpenSTAAD.GetNodeDisplacements nNodeNo, nLC, pdDisps(0)

 'Display the node displacement values in the worksheet.

    For i = 1 To 6
      Cells(31 + i, 11).Value = pdDisps(i - 1)
    Next i

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetNodesCount
GetAllNodesCoordinates

GetNextNodeCoordinate
DoesNodeHaveSupport
GetNumberOfSupportedNodes
GetAllNodesThatAreSupported

GetSupportReactions

VB Syntax

integer GetSupportReactions (long nNodeNo, integer nLC, double pdReactions)

Parameters

nNodeNo

A long value greater than 0 specifying the node number of the node for which the function is to retrieve the support reactions.

nLC

An integer value greater than 0 specifying the load case number for which the function is to retrieve the support reactions.

pdReactions

An array of 6 double (8-byte floating-point) values which the function will use to store the support reactions it retrieves.

Remarks

This function retrieves all six (Fx, Fy, Fz, Mx, My, Mz) support reactions for a given supported node and load case. The support reactions will be stored in the *pdReactions* array in the following order:

pdReactions (0) = Fx

pdReactions (1) = Fy

pdReactions (2) = Fz

pdReactions (3) = Mx

pdReactions (4) = My

pdReactions (5) = Mz

All values are given in units of kips and inches.

Example

```
Public Sub SuppReac()

'This is a VBA for Excel macro.
'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare variables for use in passing the Node Number and Load Case Number to
' the function.

 Dim nNodeNo As Long
 Dim nLC As Integer

'Declare an array of 6 Double values for storing the function results.

 Dim pdReactions(6) As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Get the support's Node Number from Row 30, Column 14 and the Load Case
' Number from Row 31, Column 14 of the Excel worksheet.

 nNodeNo = Cells(30, 14).Value
 nLC = Cells(31, 14).Value

'Retrieve the support reaction values and store them in the array.

 objOpenSTAAD.GetSupportReactions nNodeNo, nLC, pdReactions(0)

 'Display the support reactions in Column 14 of the worksheet, Rows 32-37.

    For i = 1 To 6
      Cells(31 + i, 14).Value = pdReactions(i - 1)
    Next i

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetSupportCondition
GetSupportStiffnesses

GetModeShapeDataAtNode

VB Syntax

GetModeShapeDataAtNode (integer nModeNo, integer nNodeNo, double pdX, double pdY, double pdZ, double pdrX, double pdrY, double pdrZ)

Parameters

nModeNo

An integer variable passed to the function to specify the mode number for which the function is to retrieve the mode shape data.

nNodeNo

An integer variable passed to the function to specify the number of the node at which the function is to retrieve the mode shape data.

pdX

A double (8-byte floating point) variable name passed to the function for it to use in storing the value of the node translational displacement in the X-direction retrieved from STAAD.Pro.

pdY

A double (8-byte floating point) variable name passed to the function for it to use in storing the value of the node translational displacement in the Y-direction retrieved from STAAD.Pro.

pdZ

A double (8-byte floating point) variable name passed to the function for it to use in storing the value of the node translational displacement in the Z-direction retrieved from STAAD.Pro.

pdrX

A double (8-byte floating point) variable name passed to the function for it to use in storing the value of the node rotational displacement in the X-direction retrieved from STAAD.Pro.

pdrY

A double (8-byte floating point) variable name passed to the function for it to use in storing the value of the node rotational displacement in the Y-direction retrieved from STAAD.Pro.

pdrZ

A double (8-byte floating point) variable name passed to the function for it to use in storing the value of the node rotational displacement in the Z-direction retrieved from STAAD.Pro.

Remarks

This function retrieves the translational and rotational node displacement for a given mode number at a given node. The mode number and node number are passed to the function, along with 6 double variable names for storing the function results. The function then returns the six translational and rotational displacements for the given mode number and node number.

Example

```
Public Sub ModeShape()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare 6 Double variables for storing the function results.

 Dim pdX As Double
 Dim pdY Double
 Dim pdZ Double
 Dim pdrX Double
 Dim pdrY Double
 Dim pdrZ Double

'Run an instance of OpenSTAAD and open STAAD Example No. 22 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp22.std"

'Retrieve the mode shape data for Mode 2, Node 11 and store the results
' in the double variables.

 objOpenSTAAD. GetModeShapeDataAtNode 2, 11, pdX, pdY, pdZ, pdrX, pdrY, pdrZ

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetNumberOfModes

GetNumberOfModes

VB Syntax

GetNumberOfModes (integer pnModes)

Parameters

pnModes

An integer variable name for the function to use in storing the number of modes it retrieves from STAAD.*Pro*.

Remarks

This function retrieves the number of mode shapes ultimately considered in the dynamic analysis of the currently open STAAD file.

Prior to the analysis the user may specify a maximum number of mode shapes to consider during dynamic analysis. If the user does not specify a maximum, STAAD.Pro defaults to a maximum of 6 mode shapes. However, if during convergence testing the zero through maximum frequencies are converged, the modal calculation will be completed before the maximum number of mode shapes are calculated.

Example

```
Public Sub ModeShape()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare an integer variable name for storing the function results.

 Dim pnModes As Integer

'Run an instance of OpenSTAAD and open STAAD Example No. 22 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp22.std"

'Retrieve the number of modes considered in the dynamic analysis of Example 22
' and store the results in the pnModes variable.

 objOpenSTAAD.GetNumberOfModes pnModes

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetModeShapeDataAtNode

Output Results Functions: Beams

GetMinBendingMoment

VB Syntax:

integer GetMinBendingMoment (integer nMemberNo, string strDirVector, integer nLC, double pdMinBendingMom)

Parameters

nMemberNo

An integer greater than or equal to one representing the member number for which the minimum bending moment is to be obtained.

strDirVector

A string denoting the direction in which the minimum bending moment is to be obtained. "MY" denotes bending in the y direction and "MZ" denotes bending in the z direction. The string must be enclosed in quotation marks, but it is not case-sensitive.

nLC

An integer greater than or equal to one representing the load case for which the minimum bending moment is to be obtained.

pdMinBendingMom

A double (8-byte floating-point) array for storing the value of the minimum bending moment retrieved by the function.

Remarks

This function will retrieve the minimum bending moment (MY or MZ) for a given member number, direction and load case.

All values are given in units of kips and inches.

Example

```
Public Sub MinBendMom()

'This is a VBA for Excel macro.
'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare a Double variable for storing the function results.

 Dim pdMinBendingMom As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"


'Retrieve the minimum bending moment for member no. 6, Z-direction, load case 1
' and store the results in the double variable.

 objOpenSTAAD.GetMinBendingMoment 6, "MZ", 1, pdMinBendingMom

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMaxBendingMoment

GetMaxBendingMoment

VB Syntax

integer GetMaxBendingMoment (integer nMemberNo, string strDirVector, integer nLC, double pdVal)

Parameters

nMemberNo

An integer greater than or equal to one representing the member number for which the maximum bending moment is to be obtained.

strDirVector

A string denoting the direction in which the maximum bending moment is to be obtained. "MY" denotes bending in the y direction; "MZ" denotes bending in the z direction. The string must be enclosed in quotation marks, but it is not case-sensitive.

nLC

An integer greater than or equal to one representing the load case for which the minimum bending moment is to be obtained.

pdMinBendingMom

A double (8-byte floating-point) array for storing the value of the maximum bending moment retrieved by the function.

Remarks

This function will retrieve the maximum bending moment (MY or MZ) for a given member number, direction and load case.

All values are given in units of kips and inches.

Example

```
Public Sub MaxBendMom()

'This is a VBA for Excel macro.
'Declare OpenSTAAD object variable As Output.
```

```
 Dim objOpenSTAAD As Output

'Declare a double variable for storing the function results.

 Dim pdMaxBendingMom As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"


'Retrieve the maximum bending moment for member no. 6, Z-direction, load case 1
' and store the results in the double variable.

 objOpenSTAAD.GetMaxBendingMoment 6, "MZ", 1, pdMaxBendingMom

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMinBendingMoment

GetMinShearForce

VB Syntax

integer GetMinShearForce (integer nMemberNo, string strDirVector, integer nLC, double pdVal)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to retrieve the minimum shear force.

strDirVector

A string denoting the direction in which the minimum shear force is to be obtained. “Fx” denotes shear in the x direction, “Fy” denotes shear in the y direction and “Fz” denotes shear in the z direction. The string must be enclosed in quotation marks, but it is not case-sensitive.

nLC

An integer greater than 0 representing the load case for which the minimum shear force is to be obtained.

pdVal

A double (8-byte floating point) variable name passed to the function for it to use in storing the value of the minimum shear force it retrieves.

Remarks

This function retrieves the minimum shear force for a given member number, direction and load case.

All values are given in units of kips and inches.

Example

```
Public Sub MinShear()

'This is a VBA for Excel macro.
'Declare OpenSTAAD object variable As Output.
```

```
 Dim objOpenSTAAD As Output

'Declare a Double variable for storing the function results.

 Dim pdVal As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve the minimum shear force for member no. 6, Y-direction, load case 1
' and store the results in the double variable pdVal.

 objOpenSTAAD.GetMinShearForce 6, "Fy", 1, pdVal

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMaxShearForce

GetMaxShearForce

VB Syntax

integer GetMaxShearForce (integer nMemberNo, string strDirVector, integer nLC, double pdVal)

Parameters

nMemberNo

An integer variable greater than 0 specifying the number of the member for which the function is to retrieve the maximum shear force.

strDirVector

A string denoting the direction in which the maximum shear force is to be obtained. “Fx” denotes shear in the x direction, “Fy” denotes shear in the y direction and “Fz” denotes shear in the z direction. The string must be enclosed in quotation marks, but it is not case-sensitive.

nLC

An integer greater than 0 representing the load case for which the maximum shear force is to be obtained.

pdVal

A double (8-byte floating point) variable name passed to the function for it to use in storing the value of the maximum shear force it retrieves.

Remarks

This function retrieves the maximum shear force for a given member number, direction and load case.

All values are given in units of kips and inches.

Example

```
Public Sub MaxShear()

'This is a VBA for Excel macro.
'Declare an OpenSTAAD object variable As Output.
```

```
 Dim objOpenSTAAD As Output

'Declare a double variable for storing the function results.

 Dim pdVal As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve the maximum shear force for member no. 6, Y-direction, load case 1
' and store the results in the double variable pdVal.

 objOpenSTAAD.GetMaxShearForce 6, "Fy", 1, pdVal

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMinShearForce

GetMemberEndForces

VB Syntax

integer GetMemberEndForces (long nMemberNo, integer nEnd, integer nLC, double pdForces)

Parameters

nMemberNo

A long value greater than 0 specifying the member number for which the function is to retrieve the member end forces.

nEnd

Specify a 0 to retrieve the beam end forces from the starting end of the member (starting joint number of member incidence) or a 1 to retrieve the member end forces from the end of the member (end joint number of member incidence).

nLC

An integer greater than 0 representing the load case for which the member end forces are to be obtained.

pdForces

An array of 6 double (8-byte floating-point) values which the function will use to store the member end forces it retrieves.

Remarks

This function will retrieve all six (Fx, Fy, Fz, Mx, My, Mz) member end forces for a particular end of a member for a particular load case. The end forces will be stored in the *pdForces* array in the following order:

pdForces(0) = Fx

pdForces(1) = Fy

pdForces(2) = Fz

pdForces(3) = Mx

pdForces(4) = My

pdForces(5) = Mz

All values are given in units of kips and inches.

Example

```
'This is a VB for Excel macro.

 Dim objOpenSTAAD As Output
 Dim MemberNumber As Long
 Dim EndForces(5) As Double

'Run an instance of OpenSTAAD

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")

'Open a STAAD file.

  objOpenSTAAD.SelectSTAADFile "C:\SPRO2002\STAAD\Examp\US\examp08.std"

'For member ends, 0 = End A (start) and 1 = End B (end),  e.g. for
'MEMBER INCIDENCE 5 1 8, Member 5 starts at Node 1 and Ends at Node 8.
'FX = 0, FY = 1....
'Get the member number from the worksheet cell located at Row 1, Column 2

  MemberNumber = Cells(1, 2).Value

'Write the member end forces for the starting end of the member, load case 1
' into the array called EndForces

  objOpenSTAAD.GetMemberEndForces MemberNumber, 0, 1, EndForces(0)

'Now we can display the values of the array in cells 1,1 through 1,6
' of our worksheet

  For i = 1 To 6
    Cells(i, 1).Value = EndForces(i - 1)
  Next

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing
```

See Also

GetIntermediateMemberForcesAtDistance

GetIntermediateMemberForcesAtDistance

VB Syntax

integer GetIntermediateMemberForcesAtDistance (integer nMemberNo, double dDistanceRatio, integer nLC, double pdForces)

Parameters

nMemberNo

An integer value greater than 0 specifying the member number for which the function is to retrieve the member forces.

dDistanceRatio

A double (8-byte floating point) value between 0 and 1 specifying the distance along the length of the member from the starting end of the member (End A) to the point at which the function is to retrieve the member forces. The distance is expressed as the ratio of the distance to the point of interest divided by the total length of the member. For example, to find the member forces at a point 5 feet from the starting end of a 20 foot member, set *dDistanceRatio* to a value of 0.25.

nLC

An integer greater than 0 representing the load case for which the member forces are to be obtained.

pdForces

An array of 6 double (8-byte floating-point) values which the function will use to store the member forces it retrieves.

Remarks

This function will retrieve all six (Fx, Fy, Fz, Mx, My, Mz) member forces at a given distance from the starting end (End A) of a member for a given load case. The forces will be stored in the *pdForces* array in the following order:

pdForces(0) = Fx

pdForces(1) = Fy

pdForces(2) = Fz

pdForces(3) = Mx

pdForces(4) = My

pdForces(5) = Mz

All values are given in units of kips and inches.

Example

```
Public Sub ForcesAtDist()

'This is a VBA for Excel macro.
'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare a Double variable for storing the function results.

 Dim pdForces(6) As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2002\STAAD\Examp\US\examp01.std"

'Retrieve the forces at the center point of member 6 for load case 1
' and store the results in the pdForces array.

 objOpenSTAAD.GetIntermediateMemberForcesAtDistance 6, 0.5, 1, pdForces(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberEndForces

GetMemberEndDisplacements

VB Syntax

integer GetMemberEndDisplacements (integer nMember, integer nEnd, integer nLC, double pdDispls)

Parameters

nMemberNo

An integer value greater than zero specifying the member number for which the end displacements are to be obtained.

nEnd

Specify a 0 to retrieve the member end displacements from the starting end of the member (starting joint number of member incidence) or a 1 to retrieve the member end displacements from the end of the member (end joint number of member incidence).

nLC

An integer greater than or equal to one representing the load case for which the member end displacements are to be obtained.

pdDispls

An array of 6 double (8-byte floating-point) values which the function will use to store the member end displacements it retrieves from STAAD.Pro.

Remarks

This function will retrieve all six (Δx, Δy, Δz, θx, θy, θz) member end displacements for a particular end of a member for a particular load case. The end displacements will be stored in the *pdDispls* array in the following order:

pdDispls(0) = Δx

pdDispls(1) = Δy

pdDispls(2) = Δz

pdDispls(3) = θx

pdDispls(4) = θy

pdDispls(5) = θz

All values are given in units of kip and inch.

Example

```
'This is a code snippet from a VB for Excel macro.

 Dim objOpenSTAAD As Output
 Dim MemberNumber As Integer
 Dim EndDispls(6) As Double

'Run an instance of OpenSTAAD.

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")

'Open a STAAD file.

  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp08.std"

'Get the member number from the worksheet cell located at Row 1, Column 2.

  MemberNumber = Cells(1, 2).Value

'For member ends, 0 = End A (start) and 1 = End B (end),  e.g. for
' MEMBER INCIDENCE 5 1 8, Member 5 starts at Node 1 and Ends at Node 8.
'End forces will be assigned positions in the array as FX = 0, FY = 1....MZ = 5.

'Write the member end displacements for the starting end of the member,
' load case 1 into the array called EndDispls.

  objOpenSTAAD.GetMemberEndDisplacements MemberNumber, 0, 1, EndDispls(0)

'Now we can display the values of the array in rows 1-6, Column 1 of
' our worksheet.

  For i = 1 To 6
    Cells(i, 1).Value = EndDispls(i - 1)
  Next
```

See Also

GetIntermediateMemberTransDisplacements

GetIntermediateMemberTransDisplacements

VB Syntax

integer GetIntermediateMemberTransDisplacements (integer nMemberNo, double dDistanceRatio, integer nLC, double pdDisps)

Parameters

nMemberNo

An integer greater than 0 representing the member number for which the displacements are to be obtained.

dDistanceRatio

A double (8-byte floating point) value between 0 and 1 specifying the distance along the length of the member from the starting end of the member (End A) to the point at which the function is to retrieve the displacements. The distance is expressed as the ratio of the distance to the point of interest divided by the total length of the member. For example, to find the displacements at a point 5 feet from the starting end of a 20 foot member, set *dDistanceRatio* to a value of 0.25.

nLC

An integer greater than zero representing the load case for which the member end displacements are to be obtained.

pdDisps

An array of 3 double (8-byte floating-point) values which the function will use to store the member displacements it retrieves from STAAD.Pro.

Remarks

This function will retrieve the three member translational displacements (Δx, Δy, Δz) at a given distance from the starting end (End A) of a member for a particular load case. The displacements will be stored in the *pdDisps* array in the following order:

pdDispls(0) = Δx

pdDispls(1) = Δy

pdDispls(2) = Δz

All values are given in units of kip and inch.

Example

```
Public Sub DisplAtDist()

'This is a VBA for Excel macro.
'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare an array of 3 double values for storing the function results.

 Dim pdDisps(0 To 2) As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve the displacements at the center of member 6 for load case 1, and
' store the results in the pdDisps array.

 objOpenSTAAD.GetIntermediateMemberTransDisplacements 6, 0.5, 1, pdDisps(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetMemberEndDisplacements

GetSteelDesignResults

VB Syntax

integer GetSteelDesignResults (integer nMemberNo, integer pnCriticalLC, double pdCritRatio, double pdCritSection, double pdAllowStresses, string lpszCritcCond, integer pnDesignCode, double pdForces, double pdKlr, string lpszStatus)

Parameters

nMemberNo

An integer greater than 0 representing the member number for which the steel design results are to be obtained.

pnCriticalLC

An integer variable passed to the function for it to use in storing the load case number of the critical load case for the member.

pdCritRatio

A double (8-byte floating point) variable passed to the function for it to use in storing the design ratio for the critical loading on the member.

pdCritSection

A double (8-byte floating point) value for the function to use in storing the distance (in inches) along the length of the member from the starting end of the member (End A) to the point at which the critical loading occurs.

pdAllowStresses

An array of 8 double (8-byte floating point) values for the function to use in storing the allowable stresses.

If the AISC ASD design code is used for the analysis, the allowable stresses are stored in the *fAllowStress* array in the following order:

```
1) fAllowStress(0)=Fa
2) fAllowStress(1)=Ft
3) fAllowStress(2)=Fcz
```

```
4) fAllowStress(3)=Fcy
5) fAllowStress(4)=Ftz
6) fAllowStress(5)=Fty
7) fAllowStress(6)=Fv
8) fAllowStress(7)=0
```

If the AISC LRFD design code is used for the analysis, the allowable and actual forces and moments are stored in the *fAllowStress* array as follows:

```
1) fAllowStress(0)=Pnc
2) fAllowStress(1)=pnc
3) fAllowStress(2)=Pnt
4) fAllowStress(3)=pnt
5) fAllowStress(4)=Mnz
6) fAllowStress(5)=mnz
7) fAllowStress(6)=Mny
8) fAllowStress(7)=mny
```

lpszCritCond

A string variable for the function to use to store the Section and Paragraph numbers of the governing clause in the design code.

pnDesignCode

An integer variable, either 0 or 1, for the function to use in storing the design code used for the analysis. The function stores a value of 0 to indicate ASD; a value of 1 indicates LRFD.

pdForces

An array of 6 double double (8-byte floating point) value for the function to use in storing the forces and moments at the critical section. The values are stored in the following order:

```
1) pdForces(0)=Fx
2) pdForces(1)=Fy
3) pdForces(2)=Fz
4) pdForces(3)=Mx
5) pdForces(4)=My
6) pdForces(5)=Mz
```

pdKlr

A double variable passed to the function for it to use in storing the Kl/r ratio (a.k.a. “slenderness ratio”).

lpszStatus

An string variable passed to the function for it to use in storing the status of the beam post analysis, either “PASS” or “FAIL.”

Remarks

This command retrieves the steel design results for a given member in the currently open STAAD file. The member number is passed to the function along with variable names for storing the function results. The function then returns the critical load case number, the distance of the critical section from the starting end of the beam, the allowable stresses, the governing clause of the code, the design code used for the analysis (AISC, ASD or LRFD) the forces and moments at the critical section, the Kl/r ratio and the results of the analysis (Pass or Fail).

All values are given in units of kip and inch.

Example

```
Public Sub ResultsOfSteelDes()

'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare variables for storing the function results.

 Dim pnCriticalLC As Integer
 Dim pdCritRatio As Double
 Dim pdCritSection As Double
 Dim pdAllowStresses(7) As Double
 Dim lpszCritCond As String
 Dim pnDesignCode As Integer
 Dim pdForces(5) As Double
 Dim pdKlr As Double
 Dim plszStatus As String

'Run an instance of OpenSTAAD and open STAAD Example No. 1 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp01.std"

'Retrieve the steel design results for Member 3.
'Note the use of the VB line continuation character, a space followed by an
' underscore in the following code, allowing a single code statement to
' be written on multiple lines.

  objOpenSTAAD.GetSteelDesignResults 3, pnCriticalLC, pdCritRatio, _
       pdCritSection, pdAllowStresses(0), lpszCritCond, pnDesignCode, _
       pdForces(0), pdKlr, plszStatus

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing
```

```
End Sub
```

See Also

GetMemberSteelDesignRatio

Output Results Functions: Plates

GetPlateCenterVonMisesStresses

VB Syntax

integer GetPlateCenterVonMisesStresses (integer nPlateNo, integer nLC, double pdVONT, double pdVONB)

Parameters

nPlateNo

An integer value greater than zero specifying the number of the plate element for which the Von Mises Stresses are to be retrieved.

nLC

An integer greater than zero representing the load case for which the Von Mises Stresses are to be retrieved.

pdVONT

A double (8-byte floating-point) variable name passed to the function for it to use in storing the top Von Mises stress it retrieves from STAAD.Pro.

pdVONB

A double (8-byte floating-point) variable name passed to the function for it to use in storing the bottom Von Mises stress it retrieves from STAAD.Pro.

Remarks

This function retrieves the top and bottom Von Mises stresses for the center of a given plate for a given load case.

Example

```
Sub VonMise()

'Declare OpenSTAAD object variable As Output.

 Dim objOpenSTAAD As Output

'Declare two double variables for storing the function results.

 Dim pdVONT As Double
 Dim pdVONB As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 23 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp23.std"

'Get Von Mises Stresses for center of plate 87, Load Case 1.

  objOpenSTAAD.GetPlateCenterVonMisesStresses 87, 1, pdVONT, pdVONB

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetAllPlateCenterMoments
GetAllPlateCenterPrincipalStressesAndAngles
GetAllPlateCenterForces
GetPlateCenterNormalPrincipalStresses
GetAllPlateCenterStressesAndMoments

GetAllPlateCenterPrincipalStressesAndAngles

VB Syntax

integer GetAllPlateCenterPrincipalStressesAndAngles (integer nPlateNo, integer nLC, double pdStresses)

Parameters

nPlateNo

An integer value greater than zero specifying the number of the plate element for which the center principal stresses and angles are to be obtained.

nLC

An integer greater than zero representing the load case for which the plate center principal stresses and angles are to be obtained.

pdStresses

An array of 8 double (8-byte floating-point) values which the function will use to store the the plate center principal stresses and angles it retrieves from STAAD.Pro (SMAX, SMIN, TMAX, ANGLE: top and bottom).

Remarks

This function retrieves the maximum and minimum principal stresses (SMAX and SMIN), the maximum shear stress (TMAX) and the principal plane angle for the top and bottom of a given plate for a given load case. The plate number and load case number are passed to the function. The function retrieves the principal stresses and angles and stores them in the *pdStresses* array in the following order:

pdStresses (0) = SMAX (top)

pdStresses (1) = SMIN - (top)

pdStresses (2) = TMAX - (top)

pdStresses (3) = ANGLE (top)

pdStresses (4) = SMAX (bottom)

pdStresses (5) = SMIN (bottom)

pdStresses (6) = TMAX (bottom)

pdStresses (7) = ANGLE (bottom)

All values are given in units of kips and inches.

Example

```
Public Sub PlatePrinStres()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare a double 8-value array variable for storing the function results.

 Dim pdStresses(0 To 7) As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 23 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp23.std"

'Get principal stresses for center of plate 87, Load Case 2.

  objOpenSTAAD.GetAllPlateCenterPrincipalStressesAndAngles 87, 2, pdStresses(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetPlateCenterVonMisesStresses
GetAllPlateCenterMoments
GetAllPlateCenterForces
GetPlateCenterNormalPrincipalStresses
GetAllPlateCenterStressesAndMoments

GetAllPlateCenterMoments

VB Syntax

integer GetAllPlateCenterMoments (integer nPlateNo, integer nLC, double pdMoments);

Parameters

nPlateNo

An integer value greater than zero specifying the number of the plate element for which the plate center moments are to be obtained.

nLC

An integer greater than zero representing the load case for which the plate center moments are to be obtained.

pdMoments

An array of 3 double (8-byte floating-point) values which the function will use to store the the plate center moments (MX, MY, MXY) it retrieves from STAAD.Pro.

Remarks

This function retrieves the plate center moments MX, MY, and MXY for a given plate and load case.

All values are given in units of kips and inches.

Example

```
Sub PlateCtrMom()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare an array of 3 double values for storing the function results.

 Dim pdMoments(0 To 2) As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 23 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp23.std"

'Get moments for center of plate 87, Load Case 2.
```

```
  objOpenSTAAD.GetAllPlateCenterMoments 87, 2, pdMoments(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetPlateCenterVonMisesStresses
GetAllPlateCenterPrincipalStressesAndAngles
GetAllPlateCenterForces
GetPlateCenterNormalPrincipalStresses
GetAllPlateCenterStressesAndMoments

GetAllPlateCenterForces

VB Syntax

integer GetAllPlateCenterForces (integer nPlateNo, integer nLC,double pdForces)

Parameters

nPlateNo

An integer value greater than zero specifying the number of the plate element for which the plate center forces are to be obtained.

nLC

An integer greater than zero representing the load case for which the plate center forces are to be obtained.

pdForces

An array of 5 double (8-byte floating-point) values which the function will use to store the the plate center forces (SQX, SQY, SX, SY, and SXY) it retrieves from STAAD.Pro.

Remarks

This function retrieves the plate center forces SQX, SQY, SX, SY, and SXY for a given plate and load case.

All values are given in units of kips and inches.

Example

```
Public Sub PlateCtrForces()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare an array of 5 double values for storing the function results.

 Dim pdForces(0 To 4) As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 23 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp23.std"
```

```
'Get forces for center of plate 87, Load Case 2.

  objOpenSTAAD.GetAllPlateCenterForces 87, 2, pdForces(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetPlateCenterVonMisesStresses
GetAllPlateCenterPrincipalStressesAndAngles
GetAllPlateCenterMoments
GetPlateCenterNormalPrincipalStresses
GetAllPlateCenterStressesAndMoments

GetPlateCenterNormalPrincipalStresses

VB Syntax

integer GetPlateCenterNormalPrincipalStresses (integer nPlateNo, integer nLC, double pdSMAX, double pdSMIN)

Parameters

nPlateNo

An integer value greater than zero specifying the number of the plate element for which the center normal principal stresses are to be obtained.

nLC

An integer greater than zero representing the load case for which the plate center normal principal stresses are to be obtained.

pdSMAX

A double (8-byte floating-point) variable name which the function will use to store the plate top center normal principal stress (SMAX) it retrieves from STAAD.Pro.

pdSMIN

A double (8-byte floating-point) variable name which the function will use to store the plate bottom center normal principal stress (SMIN) it retrieves from STAAD.Pro.

Remarks

This function retrieves the plate center top and bottom normal principal stresses (SMAX and SMIN) for a given plate and load case.

All values are given in units of kips and inches.

Example

```
Public Sub PlCtrNPS()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare 2 double variables for storing the function results.
```

```
 Dim pdSMAX As Double
 Dim pdSMIN As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 23 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp23.std"

'Get normal principal stresses for center of plate 87, Load Case 2.

  objOpenSTAAD.GetPlateCenterNormalPrincipalStresses 87, 2, pdSMAX, pdSMIN

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetPlateCenterVonMisesStresses
GetAllPlateCenterPrincipalStressesAndAngles
GetAllPlateCenterMoments
GetAllPlateCenterForces
GetAllPlateCenterStressesAndMoments

GetAllPlateCenterStressesAndMoments

VB Syntax

integer GetAllPlateCenterStressesAndMoments (integer nPlateNo, integer nLC, double pdStresses)

Parameters

nPlateNo

An integer value greater than zero specifying the number of the plate element for which the center stresses and moments are to be obtained.

nLC

An integer greater than zero representing the load case for which the plate center stresses and moments are to be obtained.

pdStresses

An array of 8 double (8-byte floating-point) values which the function will use to store the plate center stresses and moments (SQX, SQY, MX, MY, MXY, SX, SY, and SXY) it retrieves from STAAD.Pro.

Remarks

This function retrieves the plate center stresses and moments SQX, SQY, MX, MY, MXY, SX, SY, and SXY for a given plate and load case.

All values are given in units of kips and inches.

Example

```
Public Sub PlCtrStrMom()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare an array of 8 double values for storing the function results.

 Dim pdStresses(0 To 7) As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 23 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp23.std"
```

```
'Get principal stresses and moments for center of plate 87, Load Case 2.

  objOpenSTAAD.GetAllPlateCenterStressesAndMoments 87, 2, pdStresses(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetPlateCenterVonMisesStresses
GetAllPlateCenterPrincipalStressesAndAngles
GetAllPlateCenterMoments
GetAllPlateCenterForces
GetAllPlateCenterNormalPrincipalStresses

Output Results Functions: Solids

GetAllSolidPrincipalStresses

VB Syntax

integer GetAllSolidPrincipalStresses (integer nSolidNo, integer nCorner, integer nLC, double pdStresses);

Parameters

nSolidNo

An integer value greater than zero passed to the function to specify the number of the solid element for which the principal stresses are to be obtained.

nCorner

An integer value greater than zero passed to the function to specify the corner (node number) of the solid element at which the principal stresses are to be obtained.

nLC

An integer greater than zero passed to the function to specify the load case number for which the principal stresses are to be obtained.

pdStresses

An array of 3 double (8-byte floating-point) values which the function will use to store the principal stresses (S1, S2, and S3) it retrieves from STAAD.Pro.

Remarks

This function retrieves the principal stresses S1, S2, and S3 for a given corner of a solid element for a given load case.

All values are given in units of kips and inches.

Example

```
Public Sub SolPrinStress()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare an array of 3 double values for storing the function results.

 Dim pdStresses(0 To 2) As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 24 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp24.std"

'Get principal stresses for Node 26 of Solid Element no. 7, Load Case no. 2.

  objOpenSTAAD.GetAllSolidPrincipalStresses 7, 26, 2, pdStresses(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetAllSolidNormalStresses
GetAllSolidShearStresses
GetAllSolidVonMisesStresses

GetAllSolidNormalStresses

VB Syntax

integer GetAllSolidNormalStresses (integer nSolidNo, integer nCorner, integer nLC,double pdStresses)

Parameters

nSolidNo

An integer value greater than zero passed to the function to specify the number of the solid element for which the normal stresses are to be obtained.

nCorner

An integer value greater than zero passed to the function to specify the corner (node number) of the solid element at which the normal stresses are to be obtained.

nLC

An integer greater than zero passed to the function to specify the load case number for which the normal stresses are to be obtained.

pdStresses

An array of 3 double (8-byte floating-point) values which the function will use to store the normal stresses (SXX, SYY, and SZZ) it retrieves from STAAD.Pro.

Remarks

This function retrieves the principal stresses SXX, SYY, and SZZ for a given corner of a solid element for a given load case.

All values are given in units of kips and inches.

Example

```
Public Sub SolNormStress()
'Declare an OpenSTAAD object variable As Output.
 Dim objOpenSTAAD As Output
'Declare an array of 3 double values for storing the function results.
 Dim pdStresses(0 To 2) As Double
```

```
'Run an instance of OpenSTAAD and open STAAD Example No. 24 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp24.std"

'Get normal stresses for Node 26 of Solid Element no. 7, Load Case no. 2.

  objOpenSTAAD.GetAllSolidNormalStresses 7, 26, 2, pdStresses(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetAllSolidPrincipalStresses
GetAllSolidShearStresses
GetAllSolidVonMisesStresses

GetAllSolidShearStresses

VB Syntax

integer GetAllSolidShearStresses (integer nSolidNo, integer nCorner, integer nLC, double pdStresses)

Parameters

nSolidNo

An integer value greater than zero passed to the function to specify the number of the solid element for which the shear stresses are to be obtained.

nCorner

An integer value greater than zero passed to the function to specify the corner (node number) of the solid element at which the shear stresses are to be obtained.

nLC

An integer greater than zero passed to the function to specify the load case number for which the shear stresses are to be obtained.

pdStresses

An array of 3 double (8-byte floating-point) values which the function will use to store the shear stresses (SXY, SYZ, and SZX) it retrieves from STAAD.Pro.

Remarks

This function retrieves the shear stresses SXY, SYZ, and SZX for a given corner of a solid element for a given load case.

All values are given in units of kips and inches.

Example

```
Public Sub SolShearStress()
'Declare OpenSTAAD object variable.
 Dim objOpenSTAAD As Output
'Declare an array of 3 double values for storing the function results.
 Dim pdStresses(0 To 2) As Double
```

```
'Run an instance of OpenSTAAD and open STAAD Example No. 24 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp24.std"

'Get shear stresses for Node 26 of Solid Element no. 7, Load Case no. 2.

  objOpenSTAAD.GetAllSolidShearStresses 7, 26, 2, pdStresses(0)

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetAllSolidPrincipalStresses
GetAllSolidNormalStresses
GetAllSolidVonMisesStresses

GetAllSolidVonMisesStresses

VB Syntax

integer GetAllSolidVonMisesStresses (integer nSolidNo, integer nCorner, integer nLC, double pdStress)

Parameters

nSolidNo

An integer value greater than zero passed to the function to specify the number of the solid element for which the Von Mises Stress is to be obtained.

nCorner

An integer value greater than zero passed to the function to specify the corner (node number) of the solid element at which the Von Mises Stress is to be obtained.

nLC

An integer greater than zero passed to the function to specify the load case number for which the Von Mises Stress is to be obtained.

pdStress

A double (8-byte floating-point) variable name which the function will use to store the Von Mises Stress it retrieves from STAAD.Pro.

Remarks

This function retrieves the Von Mises stress for a given corner of a solid element for a given load case.

All values are given in units of kips and inches.

Example

```
Public Sub SolVolMise()

'Declare OpenSTAAD object variable.

 Dim objOpenSTAAD As Output

'Declare a double variable for storing the function results.

 Dim pdStress As Double

'Run an instance of OpenSTAAD and open STAAD Example No. 24 (US).

  Set objOpenSTAAD = CreateObject("OpenSTAAD.Output.1")
  objOpenSTAAD.SelectSTAADFile "C:\SPRO2004\STAAD\Examp\US\examp24.std"

'Get Von Mises stress for Node 26 of Solid Element no. 7, Load Case no. 2.

  objOpenSTAAD.GetAllSolidVonMisesStresses 7, 26, 2, pdStress

'Close the STAAD file and release the handles to the OpenSTAAD library.

  objOpenSTAAD.CloseSTAADFile
  Set objOpenSTAAD = Nothing

End Sub
```

See Also

GetAllSolidPrincipalStresses
GetAllSolidNormalStresses
GetAllSolidShearStresses

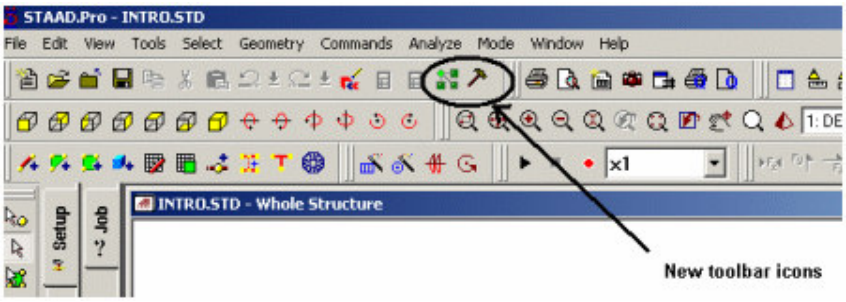
OpenSTAAD Functions – Application Object

Section 3

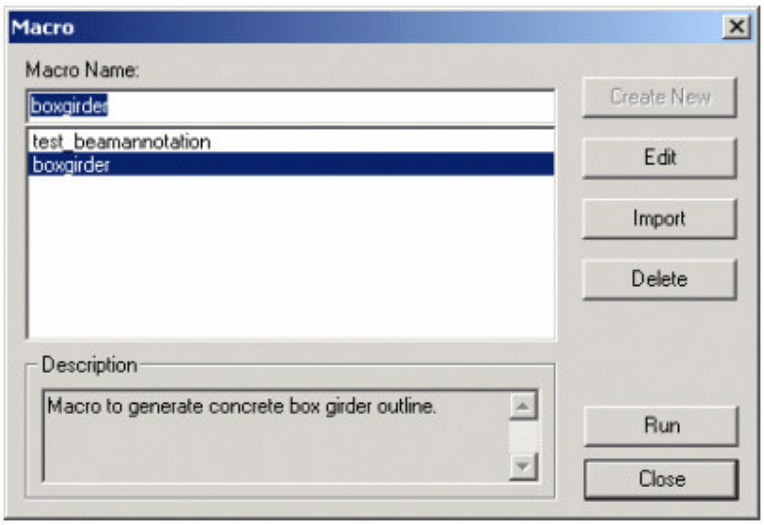
Creating Macros within STAAD.Pro and Adding Them to the STAAD.Pro Menu

The following sub-section describes how to start the VBA editor and create a new macro or load an existing one within the STAAD.Pro environment. It also describes how to run the macros you have created in the STAAD.Pro macro editor by adding a simple menu item.

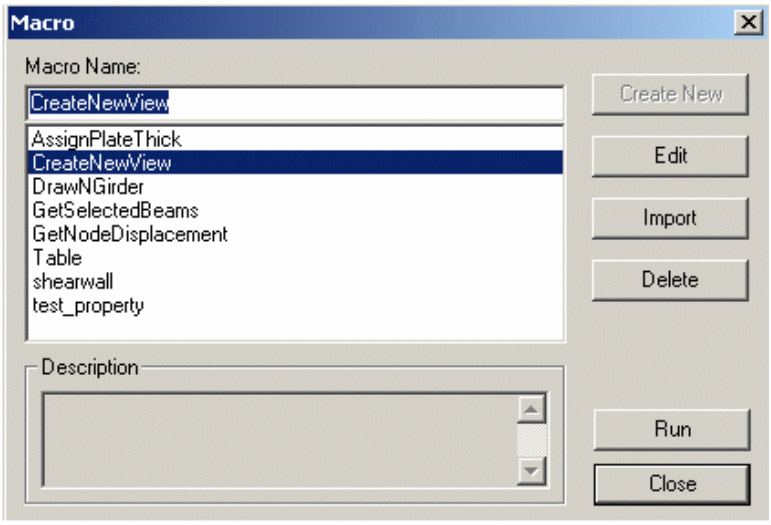
The icons to run and insert macros are shown below.



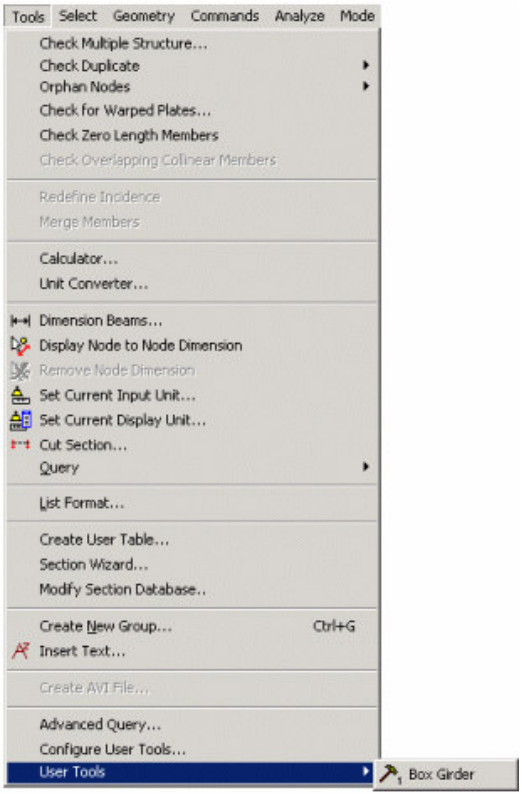
To create or load an existing macro, click on the *Macro* icon  . The following dialog box will pop up in which a new macro can be created or an existing macro can be imported for launching.



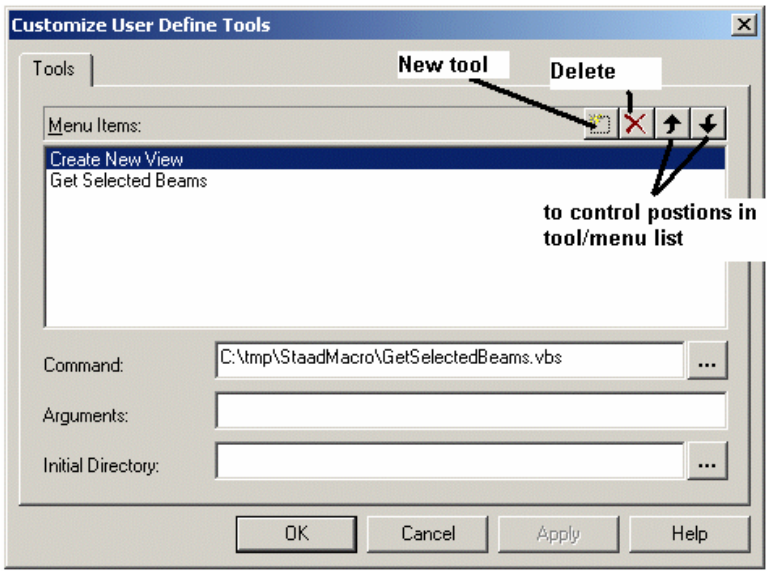
If one of the existing "Macro Names" is selected, the *Edit*, *Delete* and *Run* buttons will be enabled as shown in the following figure. A description to help identify or classify the macro can be provided if a new macro is being created.



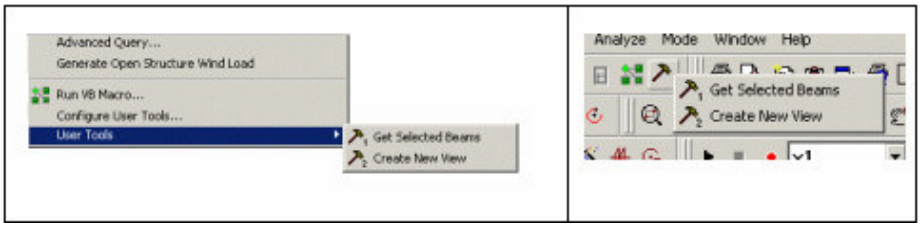
Under the *Tools* menu, the *Configure User Tools* option can be used to directly link a macro to a customized menu item. Once the macro is linked, it can be accessed either from the  icon or from *Tools* | *User Tools* option under the main menu.



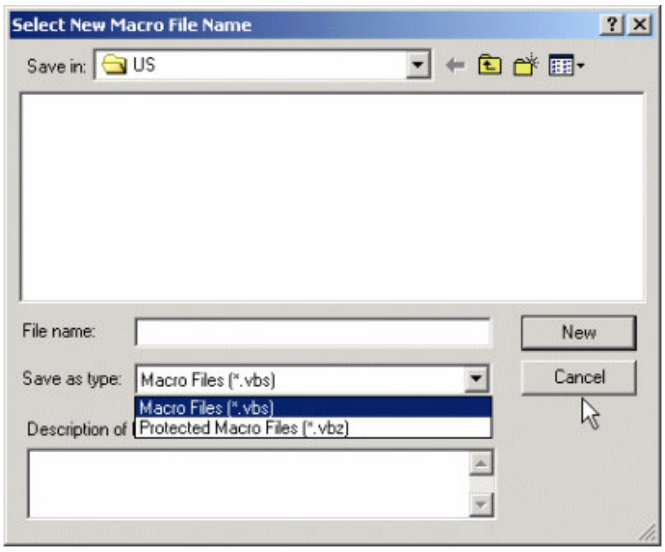
If the *Configure Tools* option is selected, a dialog box will prompt for information about the macro. Click on the *New Tool* icon to create a new macro link. Please note that the *Configure Tools* option is only meant to be used for exisiting macros and not for creating new ones. Once a name has been entered, type in the name (including path) of the macro to be linked in the *Command* box. If there are any additional parameters that are associated with the macro, they can be supplied in the *Arguments* box.



Depending upon how the tools are configured (using the dialog interface above), the macros can now be accessed directly from the  icon or from *Tools* | *User Tools* option under the main menu.



Creating macros is very easy within STAAD.*Pro*. To create a new macro, go to *Edit* | *Create New Macro* from the main menu or click on the  icon and select *New*. The following dialog box will then prompt for the name of the macro file (to be saved under) and the type of macro file. A VBS macro file is a standard macro file whose contents (code) can be viewed by other users. A VBZ macro file is a protected macro whose contents cannot be viewed even in an external editor like *Notepad*. This is useful when the user wants to sell the macro or protect its contents.



Once the *New* button has been clicked, the STAAD VBA Editor will be launched The contents of the macro can be programmed in this editor. Use the functions listed in Sections 2 and 3 to create a customized macro. The STAAD VBA Editor is designed to not only compile a macro, but help debug it as well. A synopsis of the icons in the VBA editor is listed below.

Icon	Function
	Runs or plays a macro. Once the macro has been properly created, it can be executed by clicking on this icon. The macro can also be run from *Edit* | Edit *Existing Macro*.
	*Toggle breakpoint*. This icon is used to set a breakpoint within a macro. This can be used to debug a macro when the contents of individual variables are required during runtime. The macro will physically stop at the line of code within the macro where the breakpoint(s) has/have been set.
	*Watch*. Place your mouse over any variable or constant in the macro. Click on the *Watch* icon to add the variable to the Watch list so its contents can be observed throughout the macro.
	*Step Into*. This function allows for the user to step into a function or subroutine within a macro for futher investigation. Place a breakpoint at the line where you want to *Step Into*. It can also be used to step line by line through the macro.
	*Step Over*. When the user does not want to investigate a certain function or routine but instead wants to continue debugging the next line(s), the *Step Over* function can be used.
	*Step Out*. Used to step out of the current function or subroutine and back into the main part of the code.
	*Edit Dialog*. To create a new dialog box, click on this icon. It will insert a dialog box at the present point in

	the code. To edit an existing dialog box, simply place the mouse within the dialog box code in the macro and then click on the *Edit Dialog* icon.

Once the macro has been completed, it can be exceuted by either clicking on the  icon or clicking on the  icon and selecting *Run*.

Root Applications

GetSTAADFile

VB Syntax

GetSTAADFile (String FileName, Boolean IncludePath)

Parameters

FileName

A string variable that will hold the name of the currently open STD file (without the path name).

IncludePath

A Boolean variable which if true, will write the entire path name of the STD file in the variable FileName.

Remarks

This function retrieves the name of the current STD file.

Example

```
Dim objOpenSTAAD As Object
Dim strFileName As String
Dim bIncludePath As Boolean

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

‘Retrieve the entire path
bIncludePath = true
objOpenSTAAD.GetSTAADFile strFileName, bIncludePath
```

OpenSTAADFile

VB Syntax

OpenSTAADFile (String FileName)

Parameters

FileName

A string variable that will hold the name of the STD file, which needs to be open.

Remarks

This function will open the specified STD file.

Example

```
Dim objOpenSTAAD As Object
Dim strFileName As String

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

‘Open the file
objOpenSTAAD.OpenSTAADFile strFileName
```

CloseSTAADFile

VB Syntax

CloseSTAADFile ()

Remarks

This function closes the currently open STD file.

Example

```
Dim objOpenSTAAD As Object

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

‘Close current STAAD file
objOpenSTAAD.CloseSTAADFile
```

GetSTAADFileFolder

VB Syntax

GetSTAADFileFolder (String FileFolder)

Parameters

FileFolder

A string variable that will hold the path name of folder where the currently open STD file resides. It will not write the name of the STD file into the variable.

Remarks

This function retrieves only the path of the current STD file.

Example

```
Dim objOpenSTAAD As Object
Dim strFileFolder As String

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get the file folder
objOpenSTAAD.GetSTAADFileFolder strFileFolder
```

UpdateStructure

VB Syntax

UpdateStructure ()

Remarks

This function updates the current structure.

Example

```
Dim objOpenSTAAD As Object

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Update structure
objOpenSTAAD.UpdateStructure
```

GetInputUnitForLength

VB Syntax

GetInputUnitForLength (String InputUnitForLength)

Parameters

InputUnitForLength

A string variable that will hold the input unit for length of the currently open STD file. Later the value will be internally converted to an integer ranging from 0 to 7 (0- Inch, 1- Feet, 2- Feet, 3- CentiMeter, 4- Meter, 5- MilliMeter, 6 - DeciMeter, 7 – KiloMeter).

Remarks

This function retrieves the input unit of length of the currently open STD file.

Example

```
Dim objOpenSTAAD As Object
Dim strInputUnitForLength As String

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get Length Unit
objOpenSTAAD.GetInputUnitForLength strInputUnitForLength
```

GetInputUnitForForce

VB Syntax

GetInputUnitForForce (String InputUnitForForce)

Parameters

InputUnitForForce

A string variable that will hold the input unit for force of the currently open STD file. Later the value will be internally converted to an integer ranging from 0 to 7 (0- Kilopound, 1- Pound, 2- Kilogram, 3-Metric Ton, 4- Newton, 5-Kilo Newton, 6- Mega Newton, 7- DecaNewton).

Remarks

This function retrieves the input unit of force of the currently open STD file.

Example

```
Dim objOpenSTAAD As Object
Dim strInputUnitForForce As String

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get Force Unit
objOpenSTAAD.GetInputUnitForForce strInputUnitForForce
```

SetInputUnitForLength

VB Syntax

SetInputUnitForLength (Integer InputUnitForLength)

Parameters

InputUnitForLength

An integer variable that will hold the input unit to be assigned for length of the currently open STD file. Value may vary from 0 to 7 (0- Inch, 1- Feet, 2- Feet, 3- CentiMeter, 4- Meter, 5- MilliMeter, 6 - DeciMeter, 7 – KiloMeter).

Remarks

This function sets the input unit of length of the currently open STD file.

Example

```
Dim objOpenSTAAD As Object
Dim intInputUnitForLength As Integer

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Set Length Unit
objOpenSTAAD.SetInputUnitForLength intInputUnitForLength
```

SetInputUnitForForce

VB Syntax

SetInputUnitForForce (Integer InputUnitForForce)

Parameters

InputUnitForForce

An integer variable that will hold the input unit to be assigned for force of the currently open STD file. Value may vary from 0 to 7 (0- Kilopound, 1- Pound, 2- Kilogram, 3-Metric Ton, 4- Newton, 5-Kilo Newton, 6- Mega Newton, 7- DecaNewton).

Remarks

This function sets the input unit of force of the currently open STD file.

Example

```
Dim objOpenSTAAD As Object
Dim intInputUnitForForce As Integer

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

‘Set Force Unit
objOpenSTAAD.SetInputUnitForForce intInputUnitForForce
```

SetInputUnits

VB Syntax

SetInputUnits (Integer InputUnitForLength, Integer InputUnitForForce)

Parameters

InputUnitForLength

An integer variable that will hold the input unit to be assigned for length of the currently open STD file. Value may vary from 0 to 7 (0- Inch, 1- Feet, 2- Feet, 3- CentiMeter, 4- Meter, 5- MilliMeter, 6 - DeciMeter, 7 – KiloMeter).

InputUnitForForce

An integer variable that will hold the input unit to be assigned for force of the currently open STD file. Value may vary from 0 to 7 (0- Kilopound, 1- Pound, 2- Kilogram, 3-Metric Ton, 4- Newton, 5-Kilo Newton, 6- Mega Newton, 7- DecaNewton).

Remarks

This function sets the input units of length and force of the currently open STD file.

Example

```
Dim objOpenSTAAD As Object
Dim intInputUnitForLength As Integer
Dim intInputUnitForForce As Integer

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

‘Set Input Units
objOpenSTAAD.SetInputUnits intInputUnitForLength, intInputUnitForForce
```

ShowApplication

VB Syntax

ShowApplication ()

Parameters

-none-

Remarks

This function makes the STAAD.Pro application window active.

Example

```
Dim objOpenSTAAD As Object

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Show Application Window
objOpenSTAAD.ShowApplication
```

GetProcessHandle

VB Syntax

GetProcessHandle ()

Remarks

This function retrives the current STAAD.Pro process handle.

Example

```
Dim objOpenSTAAD As Object

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

‘Get Process Handle
objOpenSTAAD.GetProcessHandle
```

GetProcessId

VB Syntax

GetProcessId ()

Remarks

This function retrives the current STAAD.Pro process ID.

Example

```
Dim objOpenSTAAD As Object

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

‘Get Process ID
objOpenSTAAD.GetProcessId
```

GetMainWindowHandle

VB Syntax

GetMainWindowHandle ()

Remarks

This function retrives the main STAAD.Pro window handle.

Example

```
Dim objOpenSTAAD As Object

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get Main Window Handle
objOpenSTAAD.GetMainWindowHandle
```

NewSTAADFile

VB Syntax

NewSTAADFile (String FileName, Integer InputUnitForLength, Integer InputUnitForForce)

Parameters

FileName

A string variable that will hold the name of the STD file, which needs to be created.

InputUnitForLength

An integer variable that will hold the input unit to be assigned for length of the new STD file. Value may vary from 0 to 7 (0- Inch, 1- Feet, 2- Feet, 3- CentiMeter, 4- Meter, 5- MilliMeter, 6 - DeciMeter, 7 – KiloMeter).

InputUnitForForce

An integer variable that will hold the input unit to be assigned for force of the new STD file. Value may vary from 0 to 7 (0- Kilopound, 1- Pound, 2- Kilogram, 3- Metric Ton, 4- Newton, 5-Kilo Newton, 6- Mega Newton, 7- DecaNewton).

Remarks

This function creates a STD file with specified length and force units.

Example

```
Dim objOpenSTAAD As Object
Dim strFileName As String
Dim intInputUnitForLength As Integer
Dim intInputUnitForForce As Integer

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

‘Create New File
objOpenSTAAD.NewSTAADFile strFileName, intInputUnitForLength,
intInputUnitForForce
```

Analyze

VB Syntax

Analyze ()

Parameters

-none-

Remarks

This function anlyzes the current STD file.

Example

```
Dim objOpenSTAAD As Object

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Run Analysis
objOpenSTAAD.Analyze
```

GetShortJobInfo

VB Syntax

GetShortJobInfo (String JobName, String JobClient, String EnggName)

Parameters

JobName

A string variable that will hold the Job Name for the current STD file.

JobClient

A string variable that will hold the Job Client for the current STD file.

EnggName

A string variable that will hold the Engineer's Name for the current STD file.

Remarks

This function retrieves the short job information of the currently open STD file.

Example

```
Dim objOpenSTAAD As Object
Dim strJobName as String
Dim strJobClient as String
Dim strEnggName as String

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get Job Info
objOpenSTAAD.GetShortJobInfo strJobName, strJobClient, strEnggName
```

SetShortJobInfo

VB Syntax

SetShortJobInfo (String JobName, String JobClient, String EnggName)

Parameters

JobName

A string variable that will hold the Job Name for the current STD file.

JobClient

A string variable that will hold the Job Client for the current STD file.

EnggName

A string variable that will hold the Engineer's Name for the current STD file.

Remarks

This function sets the short job information of the currently open STD file.

Example

```
Dim objOpenSTAAD As Object
Dim strJobName as String
Dim strJobClient as String
Dim strEnggName as String

'Set the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")
objOpenSTAAD.SetShortJobInfo strJobName, strJobClient, strEnggName
```

CreateNamedView

VB Syntax

CreateNamedView (String ViewName, Long FlagVal, Long ErrorVal)

Parameters

ViewName

A string variable that will hold the name of the view to be created.

FlagVal

A long variable that will hold the flag value depending upon which the view will be created.

ErrorVal

A long variable that will hold the error number if the view cannot be created.

Remarks

This function creates a view with the specified name.

Example

```
Dim objOpenSTAAD As Object
Dim strViewName As String
Dim lFlagVal as Long
Dim lErrorVal as Long

'Set the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")
objOpenSTAAD.CreateNamedView strViewName, lFlagVal, lErrorVal
```

SaveNamedView

VB Syntax

SaveNamedView (String ViewName, Long ErrorVal)

Parameters

ViewName

A string variable that will hold the name of the view to be saved.

ErrorVal

A long variable that will hold the error number if the view cannot be saved.

Remarks

This function saves the current view with the specified name.

Example

```
Dim strViewName As String
Dim lErrorVal as Long

'Set the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")
objOpenSTAAD.SaveNamedView strViewName, lErrorVal
```

ModifyNamedView

VB Syntax

ModifyNamedView (String ViewName, Integer Entities, Array EntityArray, Long ArrayQualifier, Long ModifyFlag, Long ErrorVal)

Parameters

ViewName

A string variable that will hold the name of the view to be modified.

Entities

An long variable that will hold number of entities.

EntityArray

An long that will hold entity number.

ArrayQualifier

A integer variable that will hold entity qualifier value. Value may vary from 0 to 4(0 - Node, 1 - Beam, 2 - Plate, 3 - Solid, 4 – Surface)

ModifyFlag

A long variable that will hold the flag value depending upon which the view will be modified.

ErrorVal

A long variable that will hold the error number if the view cannot be modified.

Remarks

This function modifies the named views of the currently open STD file.

Example

```
Dim objOpenSTAAD As Object
Dim strViewName As String
Dim intEntities As Integer
Dim lEntityNo as Long
Dim lEntityQualifier as Long
Dim lFlagVal as Long
```

```
Dim lErrorVal as Long

'Set the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")
objOpenSTAAD. ModifyNamedView strViewName, intEntities, lEntityNo,
lEntityQualifier, lFlagVal, lErrorVal
```

GetBaseUnit

VB Syntax

GetBaseUnit ()

Parameters

-none-

Remarks

This function retrives the base unit for the currently open STD file. Value will return 1 for English and 2 for Metric system.

Example

```
Dim objOpenSTAAD As Object

'Set the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")
objOpenSTAAD.GetBaseUnit
```

RemoveNamedView

VB Syntax

RemoveNamedView (String ViewName, Long ErrorVal)

Parameters

ViewName

A string variable that will hold the name of the view to be removed.

ErrorVal

A long variable that will hold the error number if the view cannot be removed.

Remarks

This function removes the current view with the specified name.

Example

```
Dim objOpenSTAAD As Object
Dim strViewName As String
Dim nErrorVal as Long

'Application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")
objOpenSTAAD.RemoveNamedView strViewName, nErrorVal
```

Quit

VB Syntax

Quit ()

Parameters

-none-

Remarks

This function quits STAAD.Pro application environment.

Example

```
Dim objOpenSTAAD As Object

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

‘Quit Application environment


objOpenSTAAD.Quit
```

Geometry Applications

Geometry.GetNodeCount

VB Syntax

Geometry.GetNodeCount ()

Parameters

-none-

Remarks

This function returns the number of nodes in the currently open STAAD file.

Example

```
Dim objOpenSTAAD As Object
Dim lNodeCount as Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get Nodes Count
lNodeCount = objOpenSTAAD.GetNodeCount
```

Geometry.GetNodeList

VB Syntax

long Geometry.GetNodeList (Long NodeNumberArray)

Parameters

NodeNumberArray

Long Array variable in which the node numbers are returned.

Remarks

This function returns the node list of the current STAAD file.

Example

```
Dim objOpenSTAAD As Object
Dim lNodeCnt as Long
Dim NodeNumberArray() As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

‘Get Node Numbers
lNodeCnt = objOpenSTAAD.GetNodeCount

ReDim NodeNumberArray(0 to (lNodeCnt-1)) As Long

'Get node list
objOpenSTAAD.Geometry.GetNodeList (NodeNumberArray)
```

Geometry.AddNode

VB Syntax

Geometry.AddNode (double CoordX, double CoordY, double CoordZ)

Parameters

CoordX, CoordY, CoordZ

Double variables providing the nodal coordinates X, Y and Z of the NodeNo.

Remarks

This function adds a node in the structure.

Example

```
Dim objOpenSTAAD As Object
Dim CoordX As Double
Dim CoordY As Double
Dim CoordZ As Double

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

CoordX = 3.0
CoordY = 2.0
CoordZ = 3.0

'Add a node (3.0, 2.0, 3.0)
objOpenSTAAD.Geometry.AddNode CoordX, CoordY, CoordZ
```

Geometry.CreateNode

VB Syntax

Geometry.CreateNode (long nNodeNo, double CoordX, double CoordY, double CoordZ)

Parameters

nNodeNo

A long variable containing the number to assign the newly created node.

CoordX, CoordY, CoordZ

Double variables providing the nodal coordinates X, Y and Z of the nNodeNo.

Remarks

This function adds a node in the structure with the number specified in nNodeNo. The difference between CreateNode and AddNode is the former has an option to label the node with any user-defined number.

Example

```
Dim objOpenSTAAD As Object
Dim CoordX As Double
Dim CoordY As Double
Dim CoordZ As Double

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

CoordX = 3.0
CoordY = 2.0
CoordZ = 3.0

'Add a node (3.0, 2.0, 3.0) and call it Node # 45
objOpenSTAAD.Geometry.CreateNode 45, CoordX, CoordY, CoordZ
```

Geometry.GetMemberCount

VB Syntax

Geometry.GetMemberCount ()

Parameters

-none-

Remarks

This function returns the number of members in the currently open STAAD file.

Example

```
Dim objOpenSTAAD As Object
Dim lMemberCount As Long

'Get the application object --
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get Member Numbers
lMemberCount = objOpenSTAAD.GetMemberCount
```

Geometry.GetBeamList

VB Syntax

long Geometry.GetBeamList (Long BeamNumberArray)

Parameters

BeamNumberArray

Long Array variable in which the beam numbers are returned.

Remarks

This function returns the member list of the current STAAD file.

Example

```
Dim objOpenSTAAD As Object
Dim lBeamCnt as Long
Dim BeamNumberArray() As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get Beam Numbers
lBeamCnt = objOpenSTAAD.GetMemberCount

ReDim BeamNumberArray(0 to (lBeamCnt-1)) As Long

'Get Beam list
objOpenSTAAD.Geometry.GetBeamList (BeamNumberArray)
```

Geometry.AddBeam

VB Syntax

Geometry.AddBeam (long NodeA, long NodeB)

Parameters

NodeA, NodeB

Long variables, provide member connectivity.

Remarks

This function adds a beam between two specified existing nodes.

Example

```
Dim objOpenSTAAD As Object
Dim NodeA As Long
Dim NodeB As Long

'Get the application object --
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Add a beam connected between nodes 2 and 4
NodeA = 2
NodeB = 4
objOpenSTAAD.Geometry.AddBeam NodeA, NodeB
```

Geometry.CreateBeam

VB Syntax

Geometry.CreateBeam (long nBeamNo, long NodeA, long NodeB)

Parameters

nBeamNo

A long variable containing the number to assign the newly created beam.

NodeA, NodeB

Long variables, provide member connectivity.

Remarks

This function adds a beam in the structure between nodes NodeA and NodeB with the number specified in nBeamNo. The difference between CreateBeam and AddBeam is the former has an option to label the beam with any user-defined number.

Example

```
Dim objOpenSTAAD As Object
Dim NodeA As Long
Dim NodeB As Long

'Get the application object --
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Add a beam connected between nodes 2 and 4. Call it beam # 77
NodeA = 2
NodeB = 4
objOpenSTAAD.Geometry.CreateBeam 77, NodeA, NodeB
```

Geometry.SplitBeam

VB Syntax

Geometry.SplitBeam (long BeamNo, integer Nodes, double DistToNodeArray)

Parameters

BeamNo

A long variable providing the beam member number to split.

Nodes

A long variable providing the number of nodes to be inserted in the beam.

DistToNodeArray

Double array variable containing the distance in length to the nodes.

Remarks

This function splits a beam.

Example

```
Dim objOpenSTAAD As Object
Dim BeamNo As Long
Dim Nodes As Long
Dim DistToNode(4) As Double

'Get the application object --
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Split Beam no 10 (length 5m say) into three unequal parts
BeamNo = 10
Nodes = 2

DistToNode(0) = 1.0
DistToNode(1) = 4.0
objOpenSTAAD.Geometry.SplitBeam BeamNo, Nodes, DistToNode
```

Geometry.SplitBeamInEqlParts

VB Syntax

Geometry.SplitBeamInEqlParts (long BeamNo, integer Parts)

Parameters

BeamNo

A long variable providing the beam member number to split.

Parts

Long variable providing the number of parts into which the beam is to be split.

Remarks

This function splits a beam into equal parts.

Example

```
Dim objOpenSTAAD As Object
Dim BeamNo As Long
Dim Parts As Long

'Get the application object --
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Split Beam no 10 (length 5m say) into three equal parts
BeamNo = 10
Parts = 3

objOpenSTAAD.Geometry.SplitBeamInEqlParts BeamNo, Parts
```

Geometry.GetBeamLength

VB Syntax

double Geometry.GetBeamLength (long BeamNo)

Return Value

A double variable containing the length of the beam.

Parameters

BeamNo

A lfong variable providing the beam member number for which the length is to be retrieved.

Remarks

Returns the length of the beam

Example

```
Dim objOpenSTAAD As Object
Dim BeamNo As Long
Dim BeamLen As Long

'Get the application object --
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get length of the beam 10
BeamNo = 10
BeamLen = objOpenSTAAD.Geometry.GetBeamLength BeamNo
```

Geometry.GetNodeCoordinates

VB Syntax

Geometry.GetNodeCoordinates (long NodeNo, double CoordX, double CoordY, double CoordZ)

Parameters

NodeNo

A long variable providing the node number.

CoordX, CoordY, CoordZ

Double variables in which the nodal coordinates X, Y and Z of the NodeNo are returned.

Remarks

Returns the coordinates of the specified node.

Example

```
Dim objOpenSTAAD As Object
Dim NodeNo As Long
Dim CoordX As Double
Dim CoordY As Double
Dim CoordZ As Double

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get a node
NodeNo = 10
objOpenSTAAD.Geometry.GetNodeCoordinates NodeNo, CoordX, CoordY, CoordZ
```

Geometry.GetNodeNumber

VB Syntax

long Geometry.GetNodeNumber (double CoordX, double CoordY, double CoordZ)

Parameters

CoordX, CoordY, CoordZ

Double variables in which the nodal coordinates X, Y and Z of the NodeNo are returned.

Remarks

Returns the node number at specified coordinates

Example

```
Dim objOpenSTAAD As Object
Dim NodeNo As Long
Dim CoordX As Double
Dim CoordY As Double
Dim CoordZ As Double

'Get the application object

'Get a node
NodeNo = objOpenSTAAD.Geometry.GetNodeNumber CoordX, CoordY, CoordZ
```

Geometry.GetNodeDistance

VB Syntax

double Geometry.GetNodeDistance (long NodeNoA, long NodeNoB)

Return Value

The distance between the nodes.

Parameters

NodeNoA, NodeNoB

Long variables providing the node numbers.

Remarks

Returns the distance between the nodes.

Example

```
Dim objOpenSTAAD As Object
Dim NodeNoA As Long
Dim NodeNoB As Long
Dim NodeDistance As Double

'Get the application object

'Get the distance between node 10 and 11
NodeNoA = 10
NodeNoB = 11
NodeDistance = objOpenSTAAD.Geometry.GetNodeDistance NodeNoA, NodeNoB
```

Geometry.GetPlateCount

VB Syntax

Geometry.GetPlateCount ()

Parameters

-none-

Remarks

This function returns the number of plates in the currently open STAAD file.

Example

```
Dim objOpenSTAAD As Object
Dim lPlateCount as Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get Plates Count
lPlateCount = objOpenSTAAD. GetPlateCount
```

Geometry.GetPlateList

VB Syntax

long Geometry.GetPlateList (Long PlateNumberArray)

Parameters

PlateNumberArray

Long Array variable in which the plate numbers are returned.

Remarks

This function returns the plate list of the current STAAD file.

Example

```
Dim objOpenSTAAD As Object
Dim lPlateCnt as Long
Dim PlateNumberArray() As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get Plate Numbers
lPlateCnt = objOpenSTAAD.GetPlateCount

ReDim PlateNumberArray(0 to (lPlateCnt-1)) As Long

'Get Plate list
objOpenSTAAD.Geometry.GetPlateList (PlateNumberArray)
```

Geometry.AddPlate

VB Syntax

Geometry.AddPlate (long NodeA, long NodeB, long NodeC, long NodeD)

Geometry.AddPlate (long NodeA, long NodeB, long NodeC)

Parameters

NodeA, NodeB, NodeC, NodeD

Long variables, provide element connectivity.

Remarks

This function adds a plate element between existing nodes.

Example

```
Dim objOpenSTAAD As Object
Dim NodeA As Long
Dim NodeB As Long
Dim NodeC As Long
Dim NodeD As Long

'Get the application object --

'Add a plate connected between nodes 2, 4, 5 and 6
NodeA = 2
NodeB = 4
NodeC = 5
NodeD = 6

objOpenSTAAD.Geometry.AddPlate NodeA, NodeB, NodeC, NodeD
```

Geometry.CreatePlate

VB Syntax

Geometry.CreatePlate (long nPlateNo, long NodeA, long NodeB, long NodeC, long NodeD)

Geometry.CreatePlate (long nPlateNo, long NodeA, long NodeB, long NodeC)

Parameters

nPlateNo

A long variable containing the number to assign the newly created plate.

NodeA, NodeB, NodeC, NodeD

Long variables, provide element connectivity. NodeD is not used for triangular (3-noded) elements.

Remarks

This function adds a plate in the structure between exisiting nodes. The difference between CreatePlate and AddPlate is the former has an option to label the plate with any user-defined number.

Example

```
Dim objOpenSTAAD As Object
Dim NodeA As Long
Dim NodeB As Long
Dim NodeC As Long
Dim NodeD As Long

'Get the application object --

'Add a plate connected between nodes 2, 4, 5 and 6, Call it Plate # 22
NodeA = 2
NodeB = 4
NodeC = 5
NodeD = 6

objOpenSTAAD.Geometry.CreatePlate 22, NodeA, NodeB, NodeC, NodeD
```

Geometry.GetSolidCount

VB Syntax

Geometry.GetSolidCount ()

Parameters

-none-

Remarks

This function returns the number of solids in the currently open STAAD file.

Example

```
Dim objOpenSTAAD As Object
Dim lSolidCount as Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get Solids Count
lSolidCount = objOpenSTAAD.GetSolidCount
```

Geometry.GetSolidList

VB Syntax

long Geometry.GetSolidList (Long SolidNumberArray)

Parameters

SolidNumberArray

Long Array variable in which the solid numbers are returned.

Remarks

This function returns the solid list of the current STAAD file.

Example

```
Dim objOpenSTAAD As Object
Dim lSolidCnt as Long
Dim SolidNumberArray() As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get Solid Numbers
lSolidCnt = objOpenSTAAD.GetSolidCount

ReDim SolidNumberArray(0 to (lSolidCnt-1)) As Long

'Get Solid list
objOpenSTAAD.Geometry.GetSolidList (SolidNumberArray)
```

Geometry.AddSolid

VB Syntax

Geometry.AddSolid (long NodeA, long NodeB, long NodeC, long NodeD, long NodeE, long NodeF, long NodeG, long NodeH)

Geometry.AddSolid (long NodeA, long NodeB, long NodeC, long NodeD, long NodeE, long NodeF)

Parameters

NodeA, NodeB, NodeC, NodeD, NodeE, NodeF, NodeG, NodeH

Long variables, provide solid element connectivity.

Remarks

This function adds a solid element between existing nodes.

Example

```
Dim objOpenSTAAD As Object
Dim NodeA As Long
Dim NodeB As Long
Dim NodeC As Long
Dim NodeD As Long
Dim NodeE As Long
Dim NodeF As Long
Dim NodeG As Long
Dim NodeH As Long

'Get the application object --

'Add a solid connected between nodes 2, 4, 5, 6 and 9, 10, 11, 12
NodeA = 2
NodeB = 4
NodeC = 5
NodeD = 6
NodeE = 9
NodeF = 10
NodeG = 11
NodeH = 12

objOpenSTAAD.Geometry.AddSolid NodeA, NodeB, NodeC, NodeD _
                                      NodeE, NodeF, NodeG, NodeH
```

Geometry.CreateSolid

VB Syntax

Geometry.CreateSolid (long nSolidNo, long NodeA, long NodeB, long NodeC, long NodeD, long NodeE, long NodeF, long NodeG, long NodeH)

Geometry.CreateSolid (long nSolidNo, long NodeA, long NodeB, long NodeC, long NodeD, long NodeE, long NodeF)

Parameters

nSolidNo

A long variable containing the number to assign the newly created solid.

NodeA, NodeB, NodeC, NodeD, NodeE, NodeF, NodeG, NodeH

Long variables, provide solid element connectivity.

Remarks

This function adds a plate in the structure between exisiting nodes. The difference between CreateSolid and AddSolid is the former has an option to label the solid with any user-defined number.

Example

```
Dim objOpenSTAAD As Object
Dim NodeA As Long
Dim NodeB As Long
Dim NodeC As Long
Dim NodeD As Long
Dim NodeE As Long
Dim NodeF As Long
Dim NodeG As Long
Dim NodeH As Long

'Get the application object --

'Add a solid connected between nodes 2, 4, 5, 6 and 9, 10, 11, 12. Call it Solid
'# 99
NodeA = 2
NodeB = 4
NodeC = 5
NodeD = 6
NodeE = 9
NodeF = 10
NodeG = 11
NodeH = 12
```

```
objOpenSTAAD.Geometry.CreateSolid 99, NodeA, NodeB, NodeC, NodeD _
                                      NodeE, NodeF, NodeG, NodeH
```

Geometry.GetSurfaceCount

VB Syntax

Geometry.GetSurfaceCount ()

Parameters

-none-

Remarks

This function returns the number of surfaces in the currently open STAAD file.

Example

```
Dim objOpenSTAAD As Object
Dim lSurfaceCount as Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get Surfaces Count
lSurfaceCount = objOpenSTAAD.GetSurfaceCount
```

Geometry.GetSurfaceList

VB Syntax

long Geometry.GetSurfaceList (Long SurfaceNumberArray)

Parameters

SurfaceNumberArray

Long Array variable in which the surface numbers are returned.

Remarks

This function returns the surface list of the current STAAD file.

Example

```
Dim objOpenSTAAD As Object
Dim lSurfaceCnt as Long
Dim SurfaceNumberArray() As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

‘Get Surface Numbers
lSurfaceCnt = objOpenSTAAD.GetSurfaceCount

ReDim SurfaceNumberArray(0 to (lSurfaceCnt-1)) As Long

'Get Surface list
objOpenSTAAD.Geometry.GetSurfaceList (SurfaceNumberArray)
```

Geometry.AddMultipleNodes

VB Syntax

Geometry.AddMultipleNodes (double faCoordinates)

Parameters

faCoordinates

Double array of m x 3 dimension containing X, Y and Z coordinates of nodes.

Remarks

This function adds multiple nodes in the structure.

Example

```
Dim objOpenSTAAD As Object
Dim Coordinates(6,2) As Double

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

For I = 0 To 6
        Coordinates(I, 0) = … 'X coordinate
        Coordinates(I, 1) = … 'Y coordinate
        Coordinates(I, 2) = … 'Z coordinate
Next I
'Add multiple nodes
objOpenSTAAD.Geometry.AddMultipleNodes Coordinates
```

Geometry.AddMultipleBeams

VB Syntax

Geometry.AddMultipleBeams (long naIncidences)

Parameters

naIncidences

Long array of m x 2 dimension containing member connectivity.

Remarks

This function adds multiple beams.

Example

```
Dim objOpenSTAAD As Object
Dim Incidences(6, 1) As Long

'Get the application object --

For I = 0 To 6
         Incidences (I, 0) = … 'ith node
         Incidences (I, 1) = … 'jth node
Next I

'Add multiple beams
objOpenSTAAD.Geometry.AddMultipleBeams Incidences
```

Geometry.AddMultiplePlates

VB Syntax

Geometry.AddMultiplePlates (long naIncidences)

Parameters

naIncidences

Long array of m x 4 dimension containing plate element connectivity.

Remarks

This function adds multiple plate elements.

Example

```
Dim objOpenSTAAD As Object
Dim Incidences(6, 3) As Long

'Get the application object --

For I = 0 To 6
        Incidences (I, 0) = … 'ith node
        Incidences (I, 1) = … 'jth node
        Incidences (I, 2) = … 'kth node
        Incidences (I, 3) = … 'lth node
Next I

'Add multiple plates
objOpenSTAAD.Geometry.AddMultiplePlates Incidences
```

Geometry.AddMultipleSolids

VB Syntax

Geometry.AddMultipleSolids (long naIncidences)

Parameters

naIncidences

Long array of m x 8 dimension containing solid element connectivity.

Remarks

This function adds multiple solid elements.

Example

```
Dim objOpenSTAAD As Object
Dim Incidences(6, 7) As Long

'Get the application object --

For I = 0 To 6
'Front face
         Incidences (I, 0) = … 'ith node
         Incidences (I, 1) = … 'jth node
         Incidences (I, 2) = … 'kth node
         Incidences (I, 3) = … 'lth node
'Back face
         Incidences (I, 4) = … 'mth node
         Incidences (I, 5) = … 'nth node
         Incidences (I, 6) = … 'oth node
         Incidences (I, 7) = … 'pth node
Next I

'Add multiple solids
objOpenSTAAD.Geometry.AddMultipleSolids Incidences
```

Geometry.DeleteNode

VB Syntax

Geometry.DeleteNode (long NodeNo)

Parameters

NodeNo

A Long variable containing the node number to be deleted from the structure.

Remarks

Deletes a node specified by *NodeNo*.

Example

```
Dim objOpenSTAAD As Object
Dim NodeNo As Long

'Get the application object --

'Delete node number 25
NodeNo = 25

objOpenSTAAD.Geometry.DeleteNode NodeNo
```

Geometry.DeleteBeam

VB Syntax

Geometry.DeleteBeam (long BeamNo)

Parameters

BeamNo

A Long variable containing the beam number to be deleted from the structure.

Remarks

Deletes a beam specified by *BeamNo*.

Example

```
Dim objOpenSTAAD As Object
Dim BeamNo As Long

'Get the application object --

'Delete beam number 25
BeamNo = 25

objOpenSTAAD.Geometry.DeleteBeam BeamNo
```

Geometry.DeletePlate

VB Syntax

Geometry.DeletePlate (long PlateNo)

Parameters

PlateNo

A long variable containing the plate number to be deleted from the structure.

Remarks

Deletes a plate specified by *PlateNo*.

Example

```
Dim objOpenSTAAD As Object
Dim PlateNo As Long

'Get the application object --

'Delete plate number 25
PlateNo = 25

objOpenSTAAD.Geometry.DeletePlate PlateNo
```

Geometry.DeleteSolid

VB Syntax

Geometry.DeleteSolid (long SolidNo)

Parameters

SolidNo

A long variable containing the solid number to be deleted from the structure.

Remarks

Deletes a solid specified by *SolidNo*.

Example

```
Dim objOpenSTAAD As Object
Dim SolidNo As Long

'Get the application object --

'Delete solid number 25
SolidNo = 25

objOpenSTAAD.Geometry.DeleteSolid SolidNo
```

Geometry.GetNoOfSelectedNodes

VB Syntax

long Geometry.GetNoOfSelectedNodes ( )

Return Value

A Long variable containing the number of nodes selected.

Remarks

Returns the number of selected nodes.

Example

```
Dim objOpenSTAAD As Object
Dim SelNodesNo As Long

'Get the application object --

'Get no. of selected nodes
SelNodesNo = objOpenSTAAD.Geometry.GetNoOfSelectedNodes
```

Geometry.GetSelectedNodes

VB Syntax

Geometry.GetSelectedNodes (long SelNodeArray, integer Sorted)

Parameters

SelNodeArray

An Long array variable to receive the selected node numbers.

Sorted

Integer variable: 1 = return the selections in sorted order, 0 = to return in the order of selection.

Remarks

Returns the node numbers selected in *SelNodeArray* variable. Call this function after GetNoOfSelectedNodes.

Example

```
Dim objOpenSTAAD As Object
Dim SelNodesNo As Long
Dim SelNodes() As Long

'Get the application object --

'Get no. of selected nodes
SelNodesNo = objOpenSTAAD.Geometry.GetNoOfSelectedNodes

'Reallocate
ReDim SelNodes (SelNodesNo - 1) As Long
objOpenSTAAD.Geometry.GetSelectedNodes SelNodes 1
```

Geometry.GetNoOfSelectedBeams

VB Syntax

long Geometry.GetNoOfSelectedBeams ( )

Return Value

A Long variable containing the number of beams selected.

Remarks

Returns the number of selected beams.

Example

```
Dim objOpenSTAAD As Object
Dim SelBeamsNo As Long

'Get the application object --

'Get no. of selected beams
SelBeamsNo = objOpenSTAAD.Geometry.GetNoOfSelectedBeams
```

Geometry.GetSelectedBeams

VB Syntax

Geometry.GetSelectedBeams (long SelBeamArray, integer Sorted)

Parameter

SelBeamArray

A Long array variable to receive the selected beam numbers.

Sorted

Integer variable: 1 = return the selections in sorted order, 0 = to return in the order of selection.

Remarks

Returns the beam numbers selected in *SelBeamArray* variable. Call this function after GetNoOfSelectedBeams.

Example

```
Dim objOpenSTAAD As Object
Dim SelBeamsNo As Long
Dim SelBeams() As Long

'Get the application object --

'Get no. of selected beams
SelBeamsNo = objOpenSTAAD.Geometry.GetNoOfSelectedBeams

'Reallocate
ReDim SelBeams(SelBeamsNo-1) As Long
objOpenSTAAD.Geometry.GetSelectedBeams SelBeams 1
```

Geometry.GetNoOfSelectedPlates

VB Syntax

long Geometry.GetNoOfSelectedPlates ( )

Return Value

A Long variable containing the number of plates selected.

Remarks

Returns the number of selected plates.

Example

```
Dim objOpenSTAAD As Object
Dim SelPlatesNo As Long

'Get the application object --

'Get no. of selected plates
SelPlatesNo = objOpenSTAAD.Geometry.GetNoOfSelectedPlates
```

Geometry.GetSelectedPlates

VB Syntax

Geometry.GetSelectedPlates (long SelPlateArray, integer Sorted)

Parameter

SelPlateArray

A Long array variable to receive the selected plate numbers.

Sorted

Integer variable: 1 = return the selections in sorted order, 0 = to return in the order of selection.

Remarks

Returns the plate numbers selected in *SelPlateArray* variable. Call this function after GetNoOfSelectedPlates.

Example

```
Dim objOpenSTAAD As Object
Dim SelPlatesNo As Long
Dim SelPlates() As Long

'Get the application object --

'Get no. of selected plates
SelPlatesNo = objOpenSTAAD.Geometry.GetNoOfSelectedPlates

'Reallocate
ReDim SelPlates(SelPlatesNo-1) As Long
objOpenSTAAD.Geometry.GetSelectedPlates SelPlates 1
```

Geometry.GetNoOfSelectedSolids

VB Syntax

long Geometry.GetNoOfSelectedSolids ( )

Return Value

A Long variable containing the number of solids selected.

Remarks

Returns the number of selected solids.

Example

```
Dim objOpenSTAAD As Object
Dim SelSolidsNo As Long

'Get the application object --

'Get no. of selected solids
SelSolidsNo = objOpenSTAAD.Geometry.GetNoOfSelectedSolids
```

Geometry.GetSelectedSolids

VB Syntax

Geometry.GetSelectedSolids (long SelSolidArray, integer Sorted)

Parameter

SelSolidArray

A Long array variable to receive the selected solid numbers.

Sorted

Integer variable: 1 = return the selections in sorted order, 0 = to return in the order of selection.

Remarks

Returns the solid numbers selected in *SelSolidArray* variable. Call this function after GetNoOfSelectedSolids.

Example

```
Dim objOpenSTAAD As Object
Dim SelSolidsNo As Long
Dim SelSolids() As Long

'Get the application object --

'Get no. of selected solids
SelSolidsNo = objOpenSTAAD.Geometry.GetNoOfSelectedSolids

'Reallocate
ReDim SelSolids(SelSolidsNo-1) As Long
objOpenSTAAD.Geometry.GetSelectedSolids SelSolids 1
```

Geometry.GetLastNodeNo

VB Syntax

long Geometry.GetLastNodeNo ( )

Return Value

A Long variable containing the last node number.

Remarks

Returns the last node number.

Example

```
Dim objOpenSTAAD As Object
Dim LastNodeNo As Long

'Get the application object --

'Get last node no.
LastNodeNo = objOpenSTAAD.Geometry.GetLastNodeNo
```

Geometry.GetLastBeamNo

VB Syntax

long Geometry.GetLastBeamNo ( )

Return Value

A Long variable containing the last beam element number.

Remarks

Returns the last beam element number.

Example

```
Dim objOpenSTAAD As Object
Dim LastBeamNo As Long

'Get the application object --

'Get last beam no.
LastBeamNo = objOpenSTAAD.Geometry.GetLastBeamNo
```

Geometry.GetLastPlateNo

VB Syntax

long Geometry.GetLastPlateNo ( )

Return Value

A Long variable containing the last plate element number.

Remarks

Returns the last plate element number.

Example

```
Dim objOpenSTAAD As Object
Dim LastPlateNo As Long

'Get the application object --

'Get last plate no.
LastPlateNo = objOpenSTAAD.Geometry.GetLastPlateNo
```

Geometry.GetLastSolidNo

VB Syntax

long Geometry.GetLastSolidNo ( )

Return Value

A Long variable containing the last solid element number.

Remarks

Returns the last solid element number.

Example

```
Dim objOpenSTAAD As Object
Dim LastSolidNo As Long

'Get the application object --

'Get last solid no.
LastSolidNo = objOpenSTAAD.Geometry.GetLastSolidNo
```

Geometry.SelectNode

VB Syntax

Geometry.SelectNode (long NodeNo)

Parameters

NodeNo

Long variable providing the node number to be selected.

Remarks

This function selects a node in the structure.

Example

```
Dim objOpenSTAAD As Object
Dim NodeNo As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Select node no 2
NodeNo = 2
objOpenSTAAD.Geometry.SelectNode(NodeNo)
```

Geometry.SelectBeam

VB Syntax

Geometry.SelectBeam (long BeamNo)

Parameters

BeamNo

Long variable providing the beam number to be selected.

Remarks

This function selects a beam in the structure.

Example

```
Dim objOpenSTAAD As Object
Dim BeamNo As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Select beam no 2
BeamNo = 2
objOpenSTAAD.Geometry.SelectBeam(BeamNo)
```

Geometry.SelectPlate

VB Syntax

Geometry.SelectPlate (long PlateNo)

Parameters

PlateNo

Long variable providing the plate number to be selected.

Remarks

This function selects a plate in the structure.

Example

```
Dim objOpenSTAAD As Object
Dim PlateNo As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Select plate no 2
PlateNo = 2
objOpenSTAAD.Geometry.SelectPlate(PlateNo)
```

Geometry.SelectSolid

VB Syntax

Geometry.SelectSolid (long SolidNo)

Parameters

SolidNo

Long variable providing the solid number to be selected.

Remarks

This function selects a solid in the structure.

Example

```
Dim objOpenSTAAD As Object
Dim SolidNo As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Select solid no 2
SolidNo = 2
objOpenSTAAD.Geometry.SelectSolid(SolidNo)
```

Geometry.SelectMultipleNodes

VB Syntax

Geometry.SelectMultipleNodes (long aNodeNos)

Parameters

aNodeNos

Long array variable providing the node numbers to be selected.

Remarks

This function selects multiple nodes in the structure.

Example

```
Dim objOpenSTAAD As Object
Dim NodeNos(6) As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Select nodes no 1 to 7
For I = 0 To 6
        NodeNos(I) = I+1
Next I
objOpenSTAAD.Geometry.SelectMultipleNodes(NodeNos)
```

Geometry.SelectMultipleBeams

VB Syntax

Geometry.SelectMultipleBeams (long aBeamNos)

Parameters

aBeamNos

Long array variable providing the beam numbers to be selected.

Remarks

This function selects multiple beams in the structure.

Example

```
Dim objOpenSTAAD As Object
Dim BeamNos(6) As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Select beams no 1 to 7
For I = 0 To 6
        BeamNos(I) = I+1
Next I
objOpenSTAAD.Geometry.SelectMultipleBeams(BeamNos)
```

Geometry.SelectMultiplePlates

VB Syntax

Geometry.SelectMultiplePlates (long aPlateNos)

Parameters

aPlateNos

Long array variable providing the plate numbers to be selected.

Remarks

This function selects multiple plates in the structure.

Example

```
Dim objOpenSTAAD As Object
Dim PlateNos(6) As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Select Plates no 1 to 7
For I = 0 To 6
        PlateNos(I) = I+1
Next I
objOpenSTAAD.Geometry.SelectMultiplePlates(PlateNos)
```

Geometry.SelectMultipleSolids

VB Syntax

Geometry.SelectMultipleSolids (long aSolidNos)

Parameters

aSolidNos

Long array variable providing the solid numbers to be selected.

Remarks

This function selects multiple solids in the structure.

Example

```
Dim objOpenSTAAD As Object
Dim SolidNos(6) As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Select Solids no 1 to 7
For I = 0 To 6
        SolidNos(I) = I+1
Next I
objOpenSTAAD.Geometry.SelectMultipleSolids(SolidNos)
```

Geometry.GetNodeIncidence

VB Syntax

Geometry.GetNodeIncidence (long NodeNo, double CoordX, double CoordY, double CoordZ)

Parameters

NodeNo

A long variable providing the node number.

CoordX, CoordY, CoordZ

Double variables in which the nodal coordinates X, Y and Z of the NodeNo are returned.

Remarks

Returns the coordinates of the specified node.

Example

```
Dim objOpenSTAAD As Object
Dim NodeNo As Long
Dim CoordX As Double
Dim CoordY As Double
Dim CoordZ As Double

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get a node
NodeNo = 10
objOpenSTAAD.Geometry.GetNodeIncidence NodeNo, CoordX, CoordY, CoordZ
```

Geometry.GetMemberIncidence

VB Syntax

Geometry.GetMemberIncidence (long MemberNo, double NodeA, double NodeB)

Parameters

MemberNo

A long variable providing the member number.

NodeA

Double variable in which the starting node number of the member is returned.

NodeB

Double variable in which the ending node number of the member is returned.

Remarks

Returns the connecting joints of the specified member.

Example

```
Dim objOpenSTAAD As Object
Dim MemberNo As Long
Dim NodeA As Double
Dim NodeB As Double

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get a Member
MemberNo = 5
objOpenSTAAD.Geometry.GetMemberIncidence MemberNo, NodeA, NodeB
```

Geometry.GetPlateIncidence

VB Syntax

Geometry.GetPlateIncidence (long PlateNo, double NodeA, double NodeB, double NodeC, double NodeD)

Parameters

PlateNo

A long variable providing the plate number.

NodeA, NodeB, NodeC, NodeD

Double variables in which all the node numbers for the plate element are returned.

Remarks

Returns the connecting joints of the specified plate.

Example

```
Dim objOpenSTAAD As Object
Dim PlateNo As Long
Dim NodeA As Double
Dim NodeB As Double
Dim NodeC As Double
Dim NodeD As Double

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get a Plate
PlateNo = 5
objOpenSTAAD.Geometry.GetPlateIncidence PlateNo, NodeA, NodeB, NodeC, NodeD
```

Geometry.GetSolidIncidence

VB Syntax

Geometry.GetSolidIncidence (long SolidNo, double NodeA, double NodeB, double NodeC, double NodeD, double NodeE, double NodeF, double NodeG, double NodeH)

Parameters

SolidNo

A long variable providing the solid number.

NodeA, NodeB, NodeC, NodeD, NodeE, NodeF, NodeG, NodeH

Double variables in which all the node numbers for the solid element are returned.

Remarks

Returns the connecting joints of the specified solid.

Example

```
Dim objOpenSTAAD As Object
Dim SolidNo As Long
Dim NodeA As Double
Dim NodeB As Double
Dim NodeC As Double
Dim NodeD As Double
Dim NodeE As Double
Dim NodeF As Double
Dim NodeG As Double
Dim NodeH As Double

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Get a Solid
PlateNo = 5
objOpenSTAAD.Geometry.GetSolidIncidence SolidNo, NodeA, NodeB, NodeC, NodeD,
NodeE, NodeF, NodeG, NodeH
```

Geometry.CreateGroup

VB Syntax

Geometry.CreateGroup (long GruopType, string GroupName)

Parameters

GruopType

A long variable providing the group type.

1 = Nodes

2 = Beams/Members

3 = Plates

4 = Solids

5 = Geometry (Beams, Plates and Solids)

6 = Floor (Floor beams)

GroupName

A string variable providing the group name.

Remarks

Creates a group with specified name for the specified type.

Example

```
Dim objOpenSTAAD As Object
Dim lGruopType As Long
Dim strGruopName As String

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Create a Group
objOpenSTAAD.Geometry.CreateGroup lGruopType, strGruopName
```

View Applications

View.ShowFront

VB Syntax

View.ShowFront ( )

Remarks

This function displays the front view of the structure.

Example

```
'Get the application object --
objOpenSTAAD.View.ShowFront
```

View.ShowBack

VB Syntax

View.ShowBack ( )

Remarks

This function displays the back view of the structure.

Example

```
'Get the application object --
objOpenSTAAD.View.ShowBack
```

View.ShowRight

VB Syntax

View.ShowRight ( )

Remarks

This function displays the right view of the structure.

Example

```
'Get the application object --
objOpenSTAAD.View.ShowRight
```

View.ShowLeft

VB Syntax

View.ShowLeft ( )

Remarks

This function displays the left view of the structure.

Example

```
'Get the application object --
objOpenSTAAD.View.ShowLeft
```

View.ShowPlan

VB Syntax

View.ShowPlan ( )

Remarks

This function displays the top (i.e. plan) view of the structure.

Example

```
'Get the application object --
objOpenSTAAD.View.ShowPlan
```

View.ShowBottom

VB Syntax

View.ShowBottom ( )

Remarks

This function displays the bottom view of the structure.

Example

```
'Get the application object --
objOpenSTAAD.View.ShowBottom
```

View.ShowIsometric

VB Syntax

View.ShowIsometric ( )

Remarks

This function displays the isometric view of the structure.

Example

```
'Get the application object --
objOpenSTAAD.View.ShowIsometric
```

View.RotateUp

VB Syntax

View.RotateUp (double Degrees)

Parameters

Degrees

Double variable providing the degree of rotation.

Remarks

Rotates the structure through *Degrees* about the Global X-Axis.

Example

```
'Get the application object --

'Rotate the structure about X Axis through 30 degrees
objOpenSTAAD.View.RotateUp 30.0
```

View.RotateDown

VB Syntax

View.RotateDown (double Degrees)

Parameters

Degrees

Double variable providing the degree of rotation.

Remarks

Rotates the structure through *Degrees* about the Global X-Axis.

Example

```
'Get the application object --

'Rotate the structure about X Axis through 30 degrees
objOpenSTAAD.View.RotateDown 30.0
```

View.RotateLeft

VB Syntax

View.RotateLeft (double Degrees)

Parameters

Degrees

Double variable providing the degree of rotation.

Remarks

Rotates the structure through *Degrees* about the Global Y-Axis.

Example

```
'Get the application object --

'Rotate the structure about Y Axis through 30 degrees
objOpenSTAAD.View.RotateLeft 30.0
```

View.RotateRight

VB Syntax

View.RotateRight (double Degrees)

Parameters

Degrees

Double variable providing the degree of rotation.

Remarks

Rotates the structure through *Degrees* about the Global Y-Axis.

Example

```
'Get the application object --

'Rotate the structure about Y Axis through 30 degrees
objOpenSTAAD.View.RotateRight 30.0
```

View.SpinLeft

VB Syntax

View.SpinLeft (double Degrees)

Parameters

Degrees

Double variable providing the degree of rotation.

Remarks

Rotates the structure through *Degrees* about the Global Z-Axis.

Example

```
'Get the application object --

'Rotate the structure about Z Axis through 30 degrees
objOpenSTAAD.View.SpinLeft 30.0
```

View.SpinRight

VB Syntax

View.SpinRight (double Degrees)

Parameters

Degrees

Double variable providing the degree of rotation.

Remarks

Rotates the structure through *Degrees* about the Global Z-Axis.

Example

```
'Get the application object --

'Rotate the structure about Z Axis through 30 degrees
objOpenSTAAD.View.SpinRight 30.0
```

View.ZoomAll

VB Syntax

View.ZoomAll ( )

Remarks

Display the whole structure.

Example

```
'Get the application object --

'Display the whole structure
objOpenSTAAD.View.ZoomAll
```

View.CreateNewViewForSelections

VB Syntax

View.CreateNewViewForSelections ( )

Remarks

Display a new view for the selected objects.

Example

```
'Get the application object --

'Display a new view for selected objects
objOpenSTAAD.View.CreateNewViewForSelections
```

View.SetLabel

VB Syntax

View.SetLabel (integer LabelID, boolean ShowFlag)

Parameters

LabelID

Integer variable identifying the label type. It may be one of the following values:

0: Node number label

1: Member number label

2: Member property reference label

3: Material property reference label

4: Support label

5: Member release label

6: Member orientation label

7: Member section label

8: Load value label

9: Axes label

10: Node position label

11: Member specification label

12: Member ends

13: Plate element number label

14: Plate element orientation label

15: Solid element number label

16: Dimension label

17: Floor load label

18: Floor load distribution diagram label

19: Wind load label

20: Wind load influence area diagram label

21: Diagram Info

ShowFlag

Boolean variable to set label mode on (True) or off (False).

Remarks

Sets the label on the structure diagram on or off.

Example

```
'Get the application object --

'Label the member numbers
objOpenSTAAD.View.SetLabel(1,True)
```

View.SetSectionView

VB Syntax

View.SetSectionView (integer Plane, float minVal, float maxVal)

Parameters

Plane

Integer variable identifying the section plane. It may be one of the following values:

0: XY Plane

1: YZ Plane

2: XZ Plane

minVal

Minimum range of the cutting plane.

maxVal

Maximum range of the cutting plane.

Remarks

Creates a section view of the structure.

Example

The following call will create a section view in the YZ plane between values X = 0.4 and X = 0.6 in the current view units:

```
Dim fmin As Double
Dim fmax As Double
Dim Plane As Integer

'Get the application object --

'Label the member numbers

Plane = 1 'YZ Plane
fmin = 0.4
fmax = 0.6
objOpenSTAAD.View.SetLabel(plane,fmin,fmax)
```

View.SetDiagramMode

VB Syntax

View.SetDiagramMode (integer diagramID, boolean ShowFlag)

Parameters

diagramID

Integer variable identifying the diagram type. It may be one of the following values:

0: Load

1: Displacement

2: MY

3: MZ

4: FY

5: FZ

6: AX

7: TR

8: Structure

9: Full Section

10: Section Outline

11: Stress

12: Shrink

13: Perspective

14: Hide Structure

15: Fill Plates & Solids

16: Hide Plates & Solids

18: Hide Piping

19: Sort Geometry

20: Sort Nodes

21: Plate Stress

22: Solid Stress

23: Mode Shape

24: Stress Animation

25: Plate Reinforcement

26: Deck Influence Diagram*

27: Deck Carriageways*

28: Deck Triangulation*

29: Deck Loads*

30: Deck Vehicles*

*Need the STAAD.*beava* component

ShowFlag

Boolean variable to set label mode on (True) or off (False).

Remarks

Sets the label on the structure diagram on or off.

Example

```
'Get the application object --
' Turn on the MZ diagram
```

```
objOpenSTAAD.View.SetDiagramMode 1, True
```

View.RefreshView

VB Syntax

View.RefreshView ( )

Parameters

(none)

Remarks

Refreshes the viewing window.

Example

```
'Get the application object --
'Refresh viewing area of STAAD.Pro Window
objOpenSTAAD.View.RefreshView()
```

View.SetNodeAnnotationMode

VB Syntax

View.SetNodeAnnotationMode (integer annotationMode, boolean bRefresh)

Parameters

annotationMode

Integer variable controlling the annotation type. It may be one of the following values:

1 = X Displacement

2 = Y Displacement

3 = Z Displacement

4 = Resultant Displacement

bRefresh

Boolean variable (*True* or *False*). If *True*, STAAD.*Pro* viewing windows refresh with the current annotation mode.

Remarks

Sets the node displacement annotation mode. This function works only in the post-processing mode of STAAD.*Pro*.

Example

```
Dim bRefresh As Boolean
Dim dLabel As Long
Dim bRet As Boolean

'Get the application object --

'Annotate the view with X displacement labels

' Make sure that Staad.Pro is in Postprocessing mode

bRet = staad.View.SetInterfaceMode 1

If bRet Then

        bRefresh = True
```

```
            dLabel = 1 ' disp x
            staad.View.SetNodeAnotationMode(dLabel, bRefresh)
        End If
```

View.SetReactionAnnotationMode

VB Syntax

View.SetReactionAnnotationMode (integer annotationMode, boolean bRefresh)

Parameters

annotationMode

Integer variable controlling the annotation type. It may be one of the following values:

1 = X Reaction

2 = Y Reaction

3 = Z Reaction

4 = X Rotation

5 = Y Rotation

6 = Z Rotation

7 = Reaction Value Only

bRefresh

Boolean variable (*True* or *False*). If *True*, STAAD.*Pro* viewing windows refresh with the current annotation mode.

Remarks

Sets the node displacement annotation mode. This function works only in the post-processing mode of STAAD.*Pro*.

Example

```
Dim bRefresh As Boolean
Dim dLabel As Long
Dim bRet As Boolean

'Get the application object --
```

```
'Annotate the view with X displacement labels
' Make sure that Staad.Pro is in Postprocessing mode
bRet = staad.View.SetInterfaceMode (1)
If bRet Then
        bRefresh = True
        dLabel = 1 ' X Reaction
        staad.View.SetReactionAnotationMode(dLabel, bRefresh)
End If
```

View.GetInterfaceMode

VB Syntax

integer View.GetInterfaceMode ( )

Parameters

(none)

Return

Returns integer value as per the following:

0: Pre-processor or modeling mode

1: Post-processing mode

2: Interactive design mode for STAAD.*etc* interoperability

4: Piping mode

5: Beava mode

Remarks

This function returns the current visual mode in the STAAD.*Pro* environment.

Example

```
Dim bMode As Integer
'Get the application object --
bMode = staad.View.GetInterfaceMode()
```

View.SetInterfaceMode

VB Syntax

boolean View.SetInterfaceMode (integer mode)

Parameters

mode

An integer variable to set the current visual mode in STAAD.*Pro* environment. Followings are the valid values for *mode*:

0: Pre-processor or modeling mode

1: Post-processing mode

2: Interactive design mode for STAAD.*etc* interoperability

4: Piping mode

5: Beava mode

Return

Returns true or false value, signifying the success or failure of the call.

Remarks

This function sets the current visual mode in the STAAD.*Pro* environment.

Example

```
Dim bRefresh As Boolean
Dim dLabel As Integer
Dim bRet As Boolean

'Get the application object --

' Make sure that Staad.Pro is in Postprocessing mode

bRet = staad.View.SetInterfaceMode 1

'Annotate the view with X displacement labels

If bRet Then

         bRefresh = True
```

```
            dLabel = 2 ' Y Reaction
            staad.View.SetReactionAnotationMode dLabel bRefresh
End If
```

View.SetModeSectionPage

VB Syntax

boolean View.SetModeSectionPage (integer modeSection, integer modeMainPage, integer modeSubPage)

Parameters

modeSection

An integer variable to set the current visual mode in the STAAD.*Pro* environment. Valid values for *modeSection* are the following:

0: Pre-processor or modeling mode

1: Post-processing mode

2: Interactive design mode for STAAD.*etc* interoperability

4: Piping mode

5: Beava mode

modeMainPage

An integer variable to set the current main page (the tabs on the left-hand side of the screen) in the STAAD.*Pro* environment. The following are valid values for *modeMainPage*:

0: Setup page

1:Geometry page

2:General page

5:Node Results page

6:Beam Result page

7:Plate Results page

8:Solid Results page

modeSubPage

An integer variable to set the current sub page (within a particular main page - the tabs on the left-hand side of the screen) in the STAAD.*Pro* environment. The following are valid values for *modeSubPage*:

0: Job Info page

1: Beam page

4: Plate page

5: Solid page

6: Property page

7: Constant page

8: Material page

9: Support page

10: Member Specifications page

11: Load page

17: Reaction page

18: Displacement page

19: Failure page

20: Forces page

21: Beam Stress page

22: Plate Stress page

23: Solid Stress page

Return Value

Returns true or false value, signifying the success or failure of the call.

Remarks

This function sets the current page mode in the STAAD.*Pro* environment.

Example

```
Sub Main()

Dim bRet As Boolean
Dim bRefresh As Boolean
Dim staad As Object

         Set staad = GetObject(,"StaadPro.OpenSTAAD")

         ' Test pre-processing modes
         bRet = staad.View.SetInterfaceMode(1)

         If bRet Then
          staad.View.SetModeSectionPage(1,6,20)
                   staad.View.SetBeamAnnotationMode(2,1,False)
         End If

         staad.View.RefreshView()


End Sub
```

View.SetBeamAnnotationMode

VB Syntax

View.SetNodeAnnotationMode (integer annotationType, integer position, boolean bRefresh)

Parameters

annotationType

Integer variable controlling the annotation type. It may be one of the following values:

0: Axial Diagram

1: Torsion Diagram

2: Moment Diagram

3: Shear Diagram

4: Stress Diagram

5: Displacement Diagram

position

Integer variable controlling what values are to be shown for the *annotationType*. It may be one of the following values:

1: End Values

2: Max Absolute Values

4: Mid-span Values

bRefresh

Boolean variable (*True* or *False*). If *True*, STAAD.*Pro* viewing windows refresh with the current annotation mode.

Remarks

Annotates the requested values for beam results in the appropriate view. This function works only in the post-processing mode of STAAD.*Pro*.

Example

```
Dim bRefresh As Boolean
Dim dLabel As Long
Dim dValueLoc As Long
Dim bRet As Boolean

'Get the application object --

'Annotate the view with values of Moments at the end of beams for the active beam
moment diagrams

' Make sure that Staad.Pro is in Postprocessing mode

bRet = staad.View.SetInterfaceMode(spNATypeDX)

If bRet Then

         bRefresh = True

         dLabel = 2 ' Moment diagram
       dValueLoc = 1 ' End values

         staad.View.SetBeamAnotationMode(dLabel, dValueLoc , bRefresh)

End If
```

View.ShowAllMembers

VB Syntax

View.ShowAllMembers ( )

Remarks

Display all members of the current structure.

Example

```
'Get the application object --

'Display the whole structure
objOpenSTAAD.View.ShowAllMembers
```

View.ShowMembers

VB Syntax

View.ShowMembers (long MemberNumber, long MemberNoArray)

Parameters

MemberNumber

A long variable that holds number of members to show.

MemberNoArray

Long array variable holds the member nos, which need to be shown.

Remarks

Show the specified members.

Example

```
Dim objOpenSTAAD As Object
Dim lMemberNumber as Long
Dim MemberNoArray() As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

` Variables
lMemberNumber = 5
ReDim MemberNoArray(0 to 4) As Long
MemberNoArray(0) = 1
MemberNoArray(1) = 2
MemberNoArray(2) = 5
MemberNoArray(3) = 7
MemberNoArray(4) = 10

'Show Member
objOpenSTAAD.View.ShowMembers (lMemberNumber, MemberNoArray)
```

View.HideMembers

VB Syntax

View.HideMembers (long MemberNumber, long MemberNoArray)

Parameters

MemberNumber

A long variable that holds number of members to hide.

MemberNoArray

Long array variable holds the member nos, which need to be hidden.

Remarks

Hide the specified members.

Example

```
Dim objOpenSTAAD As Object
Dim lMemberNumber as Long
Dim MemberNoArray() As Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

' Variables
lMemberNumber = 5
ReDim MemberNoArray(0 to 4) As Long
MemberNoArray(0) = 1
MemberNoArray(1) = 2
MemberNoArray(2) = 5
MemberNoArray(3) = 7
MemberNoArray(4) = 10

'Hide Member
objOpenSTAAD.View.HideMembers (lMemberNumber, MemberNoArray)
```

View.ShowMember

VB Syntax

View.ShowMember (Long MemberNo)

Parameters

MemberNo

A long variable that holds member number to be shown.

Remarks

Show the specified member.

Example

```
Dim objOpenSTAAD As Object
Dim lMemberNo as Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

' Variables
lMemberNo = 5

'Show Member
objOpenSTAAD.View.ShowMember (lMemberNo)
```

View.HideMember

VB Syntax

View.HideMember (Long MemberNo)

Parameters

MemberNo

A long variable that holds member number to be hidden.

Remarks

Hide the specified member.

Example

```
Dim objOpenSTAAD As Object
Dim lMemberNo as Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

' Variables
lMemberNo = 5

'Hide Member
objOpenSTAAD.View.HideMember (lMemberNo)
```

View.HideAllMembers

VB Syntax

View.HideAllMembers ()

Remarks

Hide all the members in the current structure.

Example

```
Dim objOpenSTAAD As Object

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Hide Member
objOpenSTAAD.View.HideAllMembers
```

View.ZoomExtentsMainView

VB Syntax

View.ZoomExtentsMainView ()

Remarks

Adjust viewing scale to zoom main view to the extents.

Example

```
'Get the application object --

'Display the whole structure
objOpenSTAAD.View.ZoomExtentsMainView
```

View.SetUnits

VB Syntax

View.SetUnits (integer UnitType, string UnitForIt)

Parameters

UnitType

An integer variable that holds unit type. Values are as follows:

0 = Dimension

1 = Displacement (Translational Degrees of Freedom)

2 = Sectional Dimension

3 = Sectional Area

4 = Moment of Inertia

5 = Force

6 = Moment

7 = Distributed Force

8 = Distributed Moment

9 = Material Density

10 = Acceleration

11 = Spring Constants (Translational Degrees of Freedom)

12 = Spring Constants (Rotational Degrees of Freedom)

13 = Material Modulus

14 = Stress

15 = Alpha (Coefficient of Thermal Expansion)

16 = Temperature

17 = Mass

18 = Sectional Modulus

19 = Displacement (Rotational Degrees of Freedom)

20 = Soil Subgrade Modulus

-1 = No Unit

UnitForIt

A string variable that holds the unit for the specified type. Like “cm”, “kns”, “feet”, “kn/cm” etc.

Remarks

Set viewing unit for the active view.

Example

```
Dim intUnitType as Integer
Dim strUnitForIt as String

'Get the application object --

'Set View Unit
objOpenSTAAD.View.SetUnits (intUnitType, strUnitForIt)
```

View.HidePlate

VB Syntax

View.HidePlate (Long PlateNo)

Parameters

PlateNo

A long variable that holds plate number to be hidden.

Remarks

Hide the specified plate.

Example

```
Dim objOpenSTAAD As Object
Dim lPlateNo as Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

' Variables
lPlateNo = 102

'Hide Plate
objOpenSTAAD.View.HidePlate (lPlateNo)
```

View.HideSolid

VB Syntax

View.HideSolid (Long SolidNo)

Parameters

SolidNo

A long variable that holds solid number to be hidden.

Remarks

Hide the specified solid.

Example

```
Dim objOpenSTAAD As Object
Dim lSolidNo as Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

‘ Variables
lSolidNo = 35

'Hide Solid
objOpenSTAAD.View.HideSolid (lSolidNo)
```

View.HideSurface

VB Syntax

View.HideSurface (Long SurfaceNo)

Parameters

SurfaceNo

A long variable that holds surface number to be hidden.

Remarks

Hide the specified surface.

Example

```
Dim objOpenSTAAD As Object
Dim lSurfaceNo as Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

‘ Variables
lSurfaceNo = 1005

'Hide Surface
objOpenSTAAD.View.HideSurface (lSurfaceNo)
```

View.HideEntity

VB Syntax

View.HideEntity (Long EntityNo)

Parameters

EntityNo

A long variable that holds entity (ie. Member, Plates etc.) number to be hidden.

Remarks

Hide the specified entity. Entity may be any of Beam, Plate, Solid or Suface.

Example

```
Dim objOpenSTAAD As Object
Dim lEntityNo as Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

' Variables
lEntityNo = 12

'Hide Member
objOpenSTAAD.View.HideEntity (lEntityNo)
```

View.SelectMembersParallelTo

VB Syntax

View.SelectMembersParallelTo (string AxisID)

Parameters

AxisID

A string variable that holds the Axis ID. It may have three values:

X = X-Axis

Y = Y-Axis

Z = Z-Axis

Remarks

Select members parallel to the specified axis.

Example

```
Dim objOpenSTAAD As Object
Dim strAxis as String

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")


strAxis = "X"

'Select Members
objOpenSTAAD.View.SelectMembersParallelTo (strAxis)
```

View.SelectGroup

VB Syntax

View.SelectGroup (string GroupName)

Parameters

GroupName

A string variable that holds the group name.

Remarks

Select the relevant entities of the specified group.

Example

```
Dim objOpenSTAAD As Object
Dim strGroupName as String

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")


'Select Group
objOpenSTAAD.View.SelectGroup (strGroupName)
```

View.SelectInverse

VB Syntax

View.SelectInverse (integer EntityType)

Parameters

EntityType

An integer variable that holds entity type. Values may be as follows:

1 = Node

2 = Beam

3 = Plate

4 = Solid

5 = Suface

Remarks

Inverse geometry selection for the specified entity.

Example

```
Dim objOpenSTAAD As Object
Dim intEntityType as Integer

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Inverse Selection
objOpenSTAAD.View.SelectInverse (intEntityType)
```

View.SelectByItemList

VB Syntax

View.SelectByItemList (integer EntityType, Long ItemNumber, Long ItemListArray)

Parameters

EntityType

An integer variable that holds entity type. Values may be as follows:

1 = Node

2 = Beam

3 = Plate

4 = Solid

5 = Suface

ItemNumber

A long variable that holds total number of entities needs to be selected.

ItemListArray

Long array variable holds the entity nos, which need to be selected.

Remarks

Select entities as specified.

Example

```
Dim objOpenSTAAD As Object
Dim intEntityType as Integer
Dim lItemNumber as Long
Dim lItemListArray() as Array

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

’Variables
lItemNumber = 3
```

```
ReDim lItemListArray(0 to 2) As Long
lItemListArray(0) = 1
lItemListArray(1) = 2
lItemListArray(2) = 5

'Select by list
objOpenSTAAD.View.SelectByItemList (intEntityType, lItemNumber, lItemListArray)
```

View.SelectByMissingAttribute

VB Syntax

View.SelectByMissingAttribute (integer AttributeCode)

Parameters

AttributeCode

An integer variable that holds attribute type. Values may be as follows:

1 = Missing Property

2 = Missing Modulus of Elasticity

3 = Missing Density of Material

4 = Missing Alpha (Coefficient of Thermal Expansion)

5 = Missing Poisson Ratio

Remarks

Select entity list for which specified entity is missing.

Example

```
Dim objOpenSTAAD As Object
Dim intAttributeCode as Integer

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Select by Missing Attribute
objOpenSTAAD.View.SelectByMissingAttribute (intAttributeCode)
```

View.SelectEntitiesConnectedToNode

VB Syntax

View.SelectEntitiesConnectedToNode (integer EntityType, long NodeNo)

Parameters

EntityType

An integer variable that holds entity type. Values may be as follows:

1 = Node

2 = Beam

3 = Plate

4 = Solid

5 = Suface

NodeNo

A long variable that holds node numbers with which connected entities needs to be selected.

Remarks

Select entities as specified in type and connected with the specified node.

Example

```
Dim objOpenSTAAD As Object
Dim intEntityType as Integer
Dim lNodeNo as Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Entities Connected to the Node
objOpenSTAAD.View.SelectEntitiesConnectedToNode (intEntityType, lNodeNo)
```

View.SelectEntitiesConnectedToMember

VB Syntax

View.SelectEntitiesConnectedToMember (integer EntityType, long MemberNo)

Parameters

EntityType

An integer variable that holds entity type. Values may be as follows:

1 = Member

2 = Beam

3 = Plate

4 = Solid

5 = Suface

MemberNo

A long variable that holds Member numbers with which connected entities needs to be selected.

Remarks

Select entities as specified in type and connected with the specified Member.

Example

```
Dim objOpenSTAAD As Object
Dim intEntityType as Integer
Dim lMemberNo as Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Entities Connected to the Member
objOpenSTAAD.View.SelectEntitiesConnectedToMember (intEntityType, lMemberNo)
```

View.SelectEntitiesConnectedToPlate

VB Syntax

View.SelectEntitiesConnectedToPlate (integer EntityType, long PlateNo)

Parameters

EntityType

An integer variable that holds entity type. Values may be as follows:

1 = Plate

2 = Beam

3 = Plate

4 = Solid

5 = Suface

PlateNo

A long variable that holds Plate numbers with which connected entities needs to be selected.

Remarks

Select entities as specified in type and connected with the specified Plate.

Example

```
Dim objOpenSTAAD As Object
Dim intEntityType as Integer
Dim lPlateNo as Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Entities Connected to the Plate
objOpenSTAAD.View.SelectEntitiesConnectedToPlate (intEntityType, lPlateNo)
```

View.SelectEntitiesConnectedToSolid

VB Syntax

View. SelectEntitiesConnectedToSolid (integer EntityType, long SolidNo)

Parameters

EntityType

An integer variable that holds entity type. Values may be as follows:

1 = Solid

2 = Beam

3 = Plate

4 = Solid

5 = Suface

SolidNo

A long variable that holds Solid numbers with which connected entities needs to be selected.

Remarks

Select entities as specified in type and connected with the specified Solid.

Example

```
Dim objOpenSTAAD As Object
Dim intEntityType as Integer
Dim lSolidNo as Long

'Get the application object
Set objOpenSTAAD = GetObject( , "StaadPro.OpenSTAAD")

'Entities Connected to the Solid
objOpenSTAAD.View.SelectEntitiesConnectedToSolid (intEntityType, lSolidNo)
```

Properties Applications

Country Codes

American = 1
Australian = 2
British = 3
Canadian = 4
Chinese = 5
Dutch = 6
European = 7
French = 8
German = 9
Indian = 10
Japanese = 11
Russian = 12
South African = 13
Spanish = 14
Venezuelan = 15
Korean = 16
Aluminum = 17
USColdformed = 18
ISColdformed = 19

Type Specification

ST = 0
RA = 1
D = 2
LD = 3
SD = 4
T = 5
CM = 6
TC = 7
BC = 8
TB = 9
BA = 10
FR = 11
User Defined = -1

Additional Specifications

AddSpec_1	AddSpec_2	AddSpec_3	Used with Type Specification
WP	TH		TC, BC and TB
CT	FC		CM
SP			D, BA and FR
SP			LD and SD
TH	WT	DT	Tube Define
OD	ID		Pipe Define

Note: Additional specifications should be given in current units.

Property.SetMaterialID

VB Syntax

Property.SetMaterialID (long MaterialID)

Parameters

MaterialID

A Long variable providing the material ID (0 = Steel, 1 = Concrete, and 2 = Aluminum).

Remarks

Sets the material for the member property.

Example

```
'Get the application object --

'Set material
objOpenSTAAD.Table.SetMaterialID 0
```

Property.CreateBeamPropertyFromTable

VB Syntax

long Property.CreateBeamPropertyFromTable (long Country, string SectionName, long TypeSpec, double AddSpec_1, double AddSpec_2)

Return Value

A reference number of the property created to be used to access the same.

Parameters

Country

A Long variable providing the country code. Refer to the list of *Country Codes* at the beginning of this section.

SectionName

String variable containing the name of the section.

TypeSpec

A Long variable providing the type specification. Refer to the list of *Type Specifications* at the beginning of this section.

AddSpec_1, AddSpec_2

Double variables providing additional information. Refer to the list of

Additional Specifications at the beginning of this section.

Remarks

Creates beam property from database.

Example

```
'Get the application object --

'Create property ISMB600 from Indian database
PropertyNo = objOpenSTAAD.Property.CreateBeamPropertyFromTable _
                                                    10, "ISMB600", 0, 0.0, 0.0
 '0 = ST, no additional specification required
```

Property.CreateChannelPropertyFromTable

VB Syntax

long Property.CreateChannelPropertyFromTable (long Country, string SectionName, long TypeSpec, double AddSpec_1)

Return Value

A reference number of the property created to be used to access the same.

Parameters

Country

A Long variable providing the country code. Refer to the list of Country Codes provided at the beginning of this section.

SectionName

String variable containing the name of the section.

TypeSpec

A Long variable providing the type specification. Refer to the list of

Type Specifications at the beginning of this section.

AddSpec_1

Double variable providing additional information. Refer to the list of

Additional Specifications at the beginning of this section.

Remarks

Creates channel property from database.

Example

```
'Get the application object --

'Create property ISMC600 from Indian database
PropertyNo = objOpenSTAAD.Property.CreateChannelPropertyFromTable _
                                                        10, "ISMC200", 0, 0.0
```

```
'0 = ST, no additional specification required
```

Property.CreateAnglePropertyFromTable

VB Syntax

long Property.CreateAnglePropertyFromTable (long Country, string SectionName, long TypeSpec, double AddSpec_1)

Return Value

A reference number of the property created to be used to access the same.

Parameters

Country

A Long variable providing the country code. Refer to the list of Country Codes at the beginning of this section.

SectionName

String variable containing the name of the section.

TypeSpec

A Long variable providing the type specification. Refer to the list of

Type Specifications found at the beginning of this section.

AddSpec_1

Double variable providing additional information. Refer to the list of

Additional Specifications found at the beginning of this section.

Remarks

Creates angle property from database.

Example

```
'Get the application object --

'Create property ISA100X100X10 from Indian database
PropertyNo = objOpenSTAAD.Property.CreateAnglePropertyFromTable _
                                                    10, "ISA100X100X10", 0, 0.0
```

```
'0 = ST, no additional specification required
```

Property.CreateTubePropertyFromTable

VB Syntax

long Property.CreateTubePropertyFromTable (long Country, string SectionName, long TypeSpec, double AddSpec_1, double AddSpec_2, double AddSpec_3)

Return Value

A reference number of the property created to be used to access the same.

Parameters

Country

A Long variable providing the country code. Refer to the list of Country Codes found at the beginning of this section.

SectionName

String variable containing the name of the section.

TypeSpec

A Long variable providing the type specification. Refer to the list of

Type Specifications found at the beginning of this section.

AddSpec_1, AddSpec_2, AddSpec_3

Double variables providing additional information. Refer to the list of

Additional Specifications found at the beginning of this section.

Remarks

Creates tube property from database.

Example

```
'Get the application object --

'Create property TUB30302.6 from Indian database
PropertyNo = objOpenSTAAD.Property.CreateTubePropertyFromTable _
```

```
                                                10, " TUB30302.6", 0, 0.0, 0.0, 0.0
 '0 = ST, no additional specification required

'Create user defined tube of 300mm x 200mm x 8mm
PropertyNo = objOpenSTAAD.Property.CreateTubePropertyFromTable _
                                                10, " ", -1, 0.008, 0.3, 0.2
 '-1 = User defined
```

Property.CreatePipePropertyFromTable

VB Syntax

long Property.CreatePipePropertyFromTable (long Country, string SectionName, long TypeSpec, double AddSpec_1, double AddSpec_2)

Return Value

A reference number of the property created to be used to access the same.

Parameters

Country

A Long variable providing the country code. Refer to the list of Country Codes found at the beginning of this section.

SectionName

String variable containing the name of the section.

TypeSpec

A Long variable providing the type specification. Refer to the list of

Type Specifications found at the beginning of this section.

AddSpec_1, AddSpec_2

Double variables providing additional information. Refer to the list of

Additional Specifications found at the beginning of this section.

Remarks

Creates pipe property from database.

Example

```
'Get the application object --

'Create property PIP1270.0M from Indian database
PropertyNo = objOpenSTAAD.Property.CreatePipePropertyFromTable _
```

```
                                                10, " PIP1270.0M", 0, 0.0, 0.0
 '0 = ST, no additional specification required

 'Create user defined tube of OD 300mm and ID 280mm
PropertyNo = objOpenSTAAD.Property.CreatePipePropertyFromTable _
                                                  10, " ", -1, 0.3, 0.28
 '-1 = User defined
```

Property.CreatePrismaticRectangleProperty

VB Syntax

long Property.CreatePrismaticRectangleProperty (double YD, double ZD)

Return Value

A reference number of the property created to be used to access the same.

Parameters

YD

A Double variable providing the depth along the local Y-axis.

ZD

A Double variable providing the width along the local Z-axis.

Remarks

Creates prismatic rectangle property.

Example

```
'Get the application object --

'Create rectangle property of 250x500mm
PropertyNo = objOpenSTAAD.Property.CreatePrismaticRectangleProperty 0.5, 0.25
```

Property.CreatePrismaticCircleProperty

VB Syntax

long Property.CreatePrismaticCircleProperty (double YD)

Return Value

A reference number of the property created to be used to access the same.

Parameters

YD

A Double variable providing the circle diameter.

Remarks

Creates prismatic circle property.

Example

```
'Get the application object --

'Create circle property of 250mm diameter
PropertyNo = objOpenSTAAD.Property.CreatePrismaticCircleProperty 0.25
```

Property.CreatePrismaticTeeProperty

VB Syntax

long Property.CreatePrismaticTeeProperty (double YD, double ZD, double YB, double ZB)

Return Value

A reference number of the property created to be used to access the same.

Parameters

YD

A Double variable providing the overall depth along the local Y-axis.

ZD

A Double variable providing the overall width along the local Z-axis.

YB

A Double variable providing the depth of the web along the local Y-axis.

ZB

A Double variable providing the width of the web along the local Z-axis.

Remarks

Creates prismatic tee property.

Example

```
'Get the application object --

'Create tee property of 250x500mm
PropertyNo = objOpenSTAAD.Property.CreatePrismaticTeeProperty _
                                                    0.5, 0.25, 0.4, 0.1
```

Property.CreatePrismaticTrapezoidalProperty

VB Syntax

long Property.CreatePrismaticTrapezoidalProperty (double YD, double ZD, double ZB)

Return Value

A reference number of the property created to be used to access the same.

Parameters

YD

A Double variable providing the depth along the local Y-axis.

ZD

A Double variable providing the top width along the local Z-axis.

ZB

A Double variable providing the bottom width along the local Z-axis.

Remarks

Creates prismatic trapezoidal property.

Example

```
'Get the application object --

'Create trapezoidal property
PropertyNo = objOpenSTAAD.Property.CreatePrismaticTeeProperty 0.5, 0.25, 0.2
```

Property.CreatePrismaticGeneralProperty

VB Syntax

long Property.CreatePrismaticGeneralProperty (double PropertyArray)

Return Value

A reference number of the property created to be used to access the same.

Parameters

PropertyArray

A Double array variable providing the property values as follows:

Array Index	Property Type
0	AX
1	AY
2	AZ
3	IX
4	IY
5	IZ
6	YD
7	ZD
8	YB
9	ZB

For additional information, please refer to the STAAD.*Pro* Technical Reference manual.

Remarks

Creates prismatic general section property

Example

```
Dim PropArray(0 To 9) As Double

'Get the application object --

'Create general section property
```

```
'fill PropArray here
PropArray(0) =
PropArray(1) =
PropArray(2) =
PropArray(3) =
PropArray(4) =
PropArray(5) =
PropArray(6) =
PropArray(7) =
PropArray(8) =
PropArray(9) =

PropertyNo = objOpenSTAAD.Property.CreatePrismaticGeneralProperty PropArray
```

Property.CreateTaperedIProperty

VB Syntax

long Property.CreateTaperedIProperty (double PropertyArray)

Return Value

A reference number of the property created to be used to access the same.

Parameters

PropertyArray

A Double array variable providing the property values as follows:

Array Index	Property Type
0	F1
1	F2
2	F3
3	F4
4	F5
5	F6
6	F7

For additional information, please refer to the STAAD.*Pro* Technical Reference manual.

Remarks

Creates tapered I property

Example

```
Dim PropArray(0 To 6) As Double

'Get the application object --

'Create tapered I section property
'fill PropArray here
PropArray(0) =
PropArray(1) =
PropArray(2) =
PropArray(3) =
PropArray(4) =
PropArray(5) =
```

```
PropArray(6) =
PropertyNo = objOpenSTAAD.Property.CreateTaperedIProperty PropArray
```

Property.CreateTaperedTubeProperty

VB Syntax

long Property.CreateTaperedTubeProperty (integer TypeOfTube, double d1, double d2, double Th)

Return Value

A reference number of the property created to be used to access the same.

Parameters

TypeOfTube

Integer variable providing the tube type as follows:

Type Of Tube	Value
Round	0
HexDecagonal	1
Dodecagonal	2
Octagonal	3
Hexagonal	4
Square	5

For additional information, please refer to the STAAD.*Pro* Technical Reference manual.

d1, d2

Double variables providing the depth of tube at start and end nodes.

Th

Double variable providing the thickness of tube.

Remarks

Creates tapered tube property.

Example

```
'Get the application object --

'Create tapered tube section property
PropertyNo = objOpenSTAAD.Property.CreateTaperedTubeProperty 0.5, 0.4, 0.01
```

Property.CreateAssignProfileProperty

VB Syntax

long Property.CreateAssignProfileProperty (integer TypeOfProfile)

Return Value

A reference number of the property created to be used to access the same.

Parameters

TypeOfProfile

Integer variable providing the profile type as follows:

Type Of Profile	Value
Angle	0
Double Angle	1
Beam	2
Column	3
Channel	4

For additional information, please refer to the STAAD.Pro Technical Reference manual.

Remarks

Creates "Assign Profile" property.

Example

```
'Get the application object --

'Create Assign Profile property
PropertyNo = objOpenSTAAD.Property.CreateAssignProfileProperty 2
```

Property.CreatePlateThicknessProperty

VB Syntax

long Property.CreatePlateThicknessProperty (double ThicknessArray)

long Property.CreatePlateThicknessProperty (double Thickness)

Return Value

A reference number of the property created to be used to access the same.

Parameters

ThicknessArray

Double array variable providing different plate thickness at all nodes.

Thickness

Double variable providing single plate thickness for all nodes.

Remarks

Creates plate thickness property.

Example

```
'Get the application object --

'Create plate thickness property
PropertyNo = objOpenSTAAD.Property.CreatePlateThicknessProperty 0.2
```

Property.AssignBeamProperty

VB Syntax

bool Property.AssignBeamProperty (integer BeamNo, integer PropertyNo)

bool Property.AssignBeamProperty (integer BeamNoArray, integer PropertyNo)

Return Value

TRUE if successful, else FALSE.

Parameters

BeamNo

Integer variable providing the beam number.

BeamNoArray

Integer variable array providing the beam numbers.

PropertyNo

Integer variable providing the property reference number.

Remarks

Assigns property to beam or beams.

Example

```
'Get the application object --

'assign property no 1 to beam no 3
bResult = objOpenSTAAD.Property.AssignBeamProperty 3, 1
```

Property.AssignPlateThickness

VB Syntax

bool Property.AssignPlateThickness (integer PlateNo, integer PropertyNo)

Return Value

TRUE if successful, else FALSE.

Parameters

PlateNo

Integer variable providing the plate number.

PropertyNo

Integer variable providing the property reference number.

Remarks

Assigns thickness to plate.

Example

```
'Get the application object --

'assign property no 1 to plate no 3
bResult = objOpenSTAAD.Property.AssignPlateThickness 3, 1
```

Property.AssignBetaAngle

VB Syntax

bool Property.AssignBetaAngle (integer BeamNo, double BetaAngle)

bool Property.AssignBetaAngle (integer BeamNoArray, double BetaAngle)

Return Value

TRUE if successful, else FALSE.

Parameters

BeamNo

Integer variable providing the beam number.

BeamNoArray

Integer variable array providing the beam numbers.

BetaAngle

Double variable providing the beta angle in degrees.

Remarks

Assigns Beta Angle to beam or beams.

Example

```
'Get the application object --

'assign beta angle 90.0 to beam no 3
bResult = objOpenSTAAD.Property.AssignBetaAngle 3, 90.0
```

Property.CreateMemberTrussSpec

VB Syntax

long Property.CreateMemberTrussSpec ( )

Return Value

A reference number of the specification created to be used to access the same.

Remarks

Creates MEMBER TRUSS specification.

Example

```
'Get the application object --

'create specification
lSpecNo = objOpenSTAAD.Property.CreateMemberTrussSpec
```

Property.CreateMemberInactiveSpec

VB Syntax

long Property.CreateMemberInactiveSpec ( )

Return Value

A reference number of the specification created to be used to access the same.

Remarks

Creates MEMBER INACTIVE specification.

Example

```
'Get the application object --

'create specification
lSpecNo = objOpenSTAAD.Property.CreateMemberInactiveSpec
```

Property.CreateMemberTensionSpec

VB Syntax

long Property.CreateMemberTensionSpec ( )

Return Value

A reference number of the specification created to be used to access the same.

Remarks

Creates MEMBER TENSION specification.

Example

```
'Get the application object --

'create specification
lSpecNo = objOpenSTAAD.Property.CreateMemberTensionSpec
```

Property.CreateMemberCompressionSpec

VB Syntax

long Property.CreateMemberCompressionSpec ( )

Return Value

A reference number of the specification created to be used to access the same.

Remarks

Creates MEMBER COMPRESSION specification.

Example

```
'Get the application object --

'create specification
lSpecNo = objOpenSTAAD.Property.CreateMemberCompressionSpec
```

Property.CreateMemberIgnoreStiffSpec

VB Syntax

long Property.CreateMemberIgnoreStiffSpec ( )

Return Value

A reference number of the specification created to be used to access the same.

Remarks

Creates MEMBER IGNORE specification.

Example

```
'Get the application object --

'create specification
lSpecNo = objOpenSTAAD.Property.CreateMemberIgnoreStiffSpec
```

Property.CreateMemberCableSpec

VB Syntax

long Property.CreateMemberCableSpec (integer AddCableInfo, double Value)

Return Value

A reference number of the specification created to be used to access the same.

Parameters

AddCableInfo

Integer variable specifying additional information about the cable:

0 = Initial TENSION of *Value* in the cable to be considered

1 = Unstressed LENGTH of *Value* to be considered

Value

Double variable providing Initial TENSION or Unstressed LENGTH.

Remarks

Creates MEMBER CABLE specification.

Example

```
'Get the application object --

'create specification
lSpecNo = objOpenSTAAD.Property.CreateMemberCableSpec 0, 4.5
```

Property.CreateMemberOffsetSpec

VB Syntax

long Property.CreateMemberOffsetSpec (integer StartOrEnd, integer LocalOrGlobal, double OffsetX, double OffsetY, double OffsetZ)

Return Value

A reference number of the specification created to be used to access the same.

Parameters

StartOrEnd

Integer variable specifying whether offsets are at START (= 0) or END (= 1) of the member.

LocalOrGlobal

Integer variable specifying whether the offsets are given with respect to Local (= 1) axis or Global axis (= 0).

OffsetX, OffsetY, OffsetZ

Double variables providing the X, Y and Z offsets in current units.

Remarks

Creates MEMBER OFFSET specification.

Example

```
'Get the application object --

'create specification
lSpecNo = objOpenSTAAD.Property.CreateMemberOffsetSpec 0, 0, 0.5, 0.0, 0.0
```

Property.AssignMemberSpecToBeam

VB Syntax

bool Property.AssignMemberSpecToBeam (integer BeamNo, integer SpecNo)

bool Property.AssignMemberSpecToBeam (integer BeamNoArray, integer SpecNo)

Return Value

TRUE if successful, else FALSE.

Parameters

BeamNo

Integer variable providing the beam number.

BeamNoArray

Integer variable array providing the beam numbers.

SpecNo

Integer variable providing the member specification reference number.

Remarks

Assigns specifications to beam or beams.

Example

```
'Get the application object --

'assign specification no 1 to beam no 3
bResult = objOpenSTAAD.Property.AssignMemberSpecToBeam 3, 1
```

Property.CreateElementPlaneStressSpec

VB Syntax

long Property.CreateElementPlaneStressSpec ( )

Return Value

A reference number of the specification created to be used to access the same.

Remarks

Creates ELEMENT PLANE STRESS specification.

Example

```
'Get the application object --

'create specification
lSpecNo = objOpenSTAAD.Property.CreateElementPlaneStressSpec
```

Property.CreateElementIgnoreInplaneRotnSpec

VB Syntax

long Property.CreateElementIgnoreInplaneRotnSpec ( )

Return Value

A reference number of the specification created to be used to access the same.

Remarks

Creates ELEMENT IGNORE INPLANE ROTATION specification.

Example

```
'Get the application object --

'create specification
lSpecNo = objOpenSTAAD.Property.CreateElementIgnoreInplaneRotnSpec
```

Property.AssignElementSpecToPlate

VB Syntax

bool Property.AssignElementSpecToPlate (integer PlateNo, integer SpecNo)

bool AssignElementSpecToPlate (integer PlateNoArray, integer SpecNo)

Return Value

TRUE if successful, else FALSE.

Parameters

PlateNo

Integer variable providing the plate number.

PlateNoArray

Integer variable array providing the plate numbers.

SpecNo

Integer variable providing the element specification reference number.

Remarks

Assigns specifications to plate or plates.

Example

```
'Get the application object --

'assign specification no 1 to Plate no 3
bResult = objOpenSTAAD.Property.AssignElementSpecToPlate (3, 1)
```

Property.CreateMemberReleaseSpec

VB Syntax

long Property.CreateMemberReleaseSpec (integer StartOrEnd, Integer DOFReleaseArray, Double SpringConstantsArray)

Return Value

A reference number of the specification created to be used to access the same.

Parameters

StartOrEnd

Integer variable specifying whether offsets are at START (= 0) or END (= 1) of the member.

DOFReleaseArray

Integer variable array of size 6 providing releases in different degrees of freedom (0 = No Release, 1 = Release) FX, FY, FZ, MX, MY and MZ.

SpringConstantsArray

Double variable array of size 6 providing the spring constants KFX, KFY, KFZ, KMX, KMY and KMZ.

Remarks

Creates MEMBER RELEASE specification.

Example

```
Dim DOFRelease(0 To 5) As Integer
Dim SpringConst (0 To 5) As Double

 'Get the application object --
 'set the flags for releases
 'set spring constants if any

 'create specification
lSpecNo = objOpenSTAAD.Property.CreateMemberReleaseSpec _
                                        0, DOFRelease, SpringConst
```

Property.CreateMemberPartialReleaseSpec

VB Syntax

long Property.CreateMemberPartialReleaseSpec (integer StartOrEnd, integer PartialRelease, double ReleaseFactor)

long CreateMemberPartialReleaseSpec (integer StartOrEnd, integer PartialReleaseArray, double ReleaseFactorArray)

Return Value

A reference number of the specification created to be used to access the same.

Parameters

StartOrEnd

Integer variable specifying whether offsets are at START (= 0) or END (= 1) of the member.

PartialRelease

Integer variable to set the flag for MP (0 = No Release, 1 = Release).

PartialReleaseArray

Integer variable array of size 3 providing releases in different degrees of freedom (0 = No Release, 1 = Release) MPX, MPY and MPZ.

ReleaseFactor

Double variable providing the partial moment release factor to be applied uniformly to all three DOF's.

ReleaseFactorArray

Double variable array of size 3 providing the partial moment release factors in respective DOF's.

Remarks

Creates MEMBER RELEASE specification.

Example

```
Dim DOFRelease(0 To 2) As Integer
Dim MPFactor (0 To 2) As Double

 'Get the application object --
 'set the flags for releases
 'set moment release factors if any

 'create specification
lSpecNo = objOpenSTAAD.Property.CreateMemberPartialReleaseSpec _
                                                    0, DOFRelease, MPFactor
```

Property.CreateElementNodeReleaseSpec

VB Syntax

long Property.CreateElementNodeReleaseSpec (integer Node, integer DOFReleaseArray)

Return Value

A reference number of the specification created to be used to access the same.

Parameters

Node

Integer variable specifying node (1, 2, 3 or 4) of the element to be released.

DOFReleaseArray

Integer variable array of size 6 providing releases in different degrees of freedom (0 = No Release, 1 = Release) FX, FY, FZ, MX, MY and MZ.

Remarks

Creates ELEMENT RELEASE specification.

Example

```
Dim DOFRelease(0 To 5) As Integer

 'Get the application object --
 'set the flags for releases

 'create specification
lSpecNo = objOpenSTAAD.Property.CreateElementNodeReleaseSpec 1, DOFRelease
```

Property.GetCountryTableNo

VB Syntax

Property.GetCountryTableNo (long BeamNo)

Parameters

BeamNo

Long variable holds the beam no.

Remarks

Returns country table no for the specified member.

Example

```
Dim lCountryTableNo As Long
Dim lBeamNo as Long

'Get the application object -

'Get Property
lCountryTableNo = objOpenSTAAD.Property.GetCountryTableNo (lBeamNo)
```

Property.GetSectionTableNo

VB Syntax

Property.GetSectionTableNo (long BeamNo)

Parameters

BeamNo

Long variable holds the beam no.

Remarks

Returns section table no for the specified member.

Example

```
Dim lSectionTableNo As Long
Dim lBeamNo as Long

'Get the application object -

'Get Property
lSectionTableNo = objOpenSTAAD.Property.GetSectionTableNo (lBeamNo)
```

Property.GetBeamSectionName

VB Syntax

Property.GetBeamSectionName (long BeamNo)

Parameters

BeamNo

Long variable holds the beam no.

Remarks

Returns beam section name for the specified member.

Example

```
Dim lBeamSectionName As Long
Dim lBeamNo as Long

'Get the application object -

'Get Property
lBeamSectionName = objOpenSTAAD.Property.GetBeamSectionName (lBeamNo)
```

Property.GetBeamSectionPropertyTypeNo

VB Syntax

Property.GetBeamSectionPropertyTypeNo (long BeamNo)

Parameters

BeamNo

Long variable holds the beam no.

Remarks

Returns beam section property type no for the specified member.

Example

```
Dim lBSPropertyTypeNo As Long
Dim lBeamNo as Long

'Get the application object -

'Get Property
lBSPropertyTypeNo = objOpenSTAAD.Property.GetBeamSectionPropertyTypeNo (lBeamNo)
```

Property.GetBeamMaterialName

VB Syntax

Property.GetBeamMaterialName (long BeamNo)

Parameters

BeamNo

Long variable holds the beam no.

Remarks

Returns beam material name for the specified member.

Example

```
Dim lBeamMaterialName As Long
Dim lBeamNo as Long

'Get the application object -

'Get Property
lBeamMaterialName = objOpenSTAAD.Property.GetBeamMaterialName (lBeamNo)
```

Property.GetMaterialProperty

VB Syntax

Property.GetMaterialProperty (string MaterialName, double Elasticity, double Poisson, double Density, double Alpha, double Damp)

Parameters

MaterialName

String holds the material name.

Elasticity, Poisson, Density, Alpha, Damp

Double variables return material properties.

Remarks

Retrives material properties of the specified material.

Example

```
 'Get the application object --

 'Get Property
objOpenSTAAD.Property.GetMaterialProperty (strMaterialName, dElasticity,
dPoisson, dDensity, dAlpha, dDamp)
```

Property.GetBeamConstants

VB Syntax

Property.GetBeamConstants (long BeamNo, double Elasticity, double Poisson, double Density, double Alpha, double Damp)

Parameters

BeamNo

Long variable holds the beam number.

Elasticity, Poisson, Density, Alpha, Damp

Double variables return material properties.

Remarks

Retrives material properties of the specified beam member.

Example

```
'Get the application object --

'Get Property
objOpenSTAAD.Property.GetBeamConstants (lBeamNo, dElasticity, dPoisson, dDensity,
dAlpha, dDamp)
```

Property.GetBeamPropertyAll

VB Syntax

Property.GetBeamPropertyAll (long BeamNo, double Width, double Depth, double Ax, double Ay, double Az, double Ix, double Iy, double Iz, double Tf, double Tw)

Parameters

BeamNo

Long variable holds the beam number.

Width, Depth, Ax, Ay, Az, Ix, Iy, Iz, Tf, Tw

Double variables return all member properties.

Remarks

Retrives long member properties of the specified beam member.

Example

```
'Get the application object --

'Get Property
objOpenSTAAD.Property.GetBeamPropertyAll (lBeamNo, dWidth, dDepth, dAx, dAy, dAz,
dIx, dIy, dIz, dTf, dTw)
```

Property.GetBeamProperty

VB Syntax

Property.GetBeamProperty (long BeamNo, double Width, double Depth, double Ax, double Ay, double Az, double Ix, double Iy, double Iz)

Parameters

BeamNo

Long variable holds the beam number.

Width, Depth, Ax, Ay, Az, Ix, Iy, Iz

Double variables return only sectional member properties.

Remarks

Retrives short member properties of the specified beam member.

Example

```
'Get the application object --

'Get Property
objOpenSTAAD.Property.GetBeamProperty (lBeamNo, dWidth, dDepth, dAx, dAy, dAz,
dIx, dIy, dIz)
```

Property.GetBetaAngle

VB Syntax

Property.GetBetaAngle (long BeamNo)

Parameters

BeamNo

Long variable holds the beam number.

Remarks

Retrives beta angle of the specified beam member.

Example

```
'Get the application object --

'Get Property
objOpenSTAAD.Property.GetBetaAngle (lBeamNo)
```

Property.GetSectionPropertyCount

VB Syntax

Property.GetSectionPropertyCount ()

Remarks

Returns total number of different sectional properties exist in the current STAAD file.

Example

```
Dim lSectionPropertyCount as Long

'Get the application object --

'Get Property
lSectionPropertyCount = objOpenSTAAD.Property.GetSectionPropertyCount
```

Property.GetSectionPropertyName

VB Syntax

Property.GetSectionPropertyName (long SecRefNo, string PropertyName)

Parameters

SecRefNo

Long variable holds the section property reference number.

PropertyName

String variable returns the property name.

Remarks

Returns the property name for the specified section property reference number.

Example

```
'Get the application object --

'Get Property
objOpenSTAAD.Property.GetSectionPropertyName (lSecRefNo, strPropertyName)
```

Property.GetSectionPropertyType

VB Syntax

Property.GetSectionPropertyType (long SecRefNo)

Parameters

SecRefNo

Long variable holds the section property reference number.

Remarks

Returns the section property type for the specified section property reference number.

Example

```
'Get the application object --

'Get Property
objOpenSTAAD.Property.GetSectionPropertyType (lSecRefNo)
```

Property.GetSectionPropertyCountry

VB Syntax

Property.GetSectionPropertyCountry (long SecRefNo)

Parameters

SecRefNo

Long variable holds the section property reference number.

Remarks

Returns the country reference number for the section property reference number specified.

Example

```
'Get the application object --

'Get Property
objOpenSTAAD.Property.GetSectionPropertyCountry (lSecRefNo)
```

Property.GetIsotropicMaterialCount

VB Syntax

Property.GetIsotropicMaterialCount ()

Remarks

Returns number of isotropic material present in the current structure.

Example

```
Dim lIsotropicMaterialCount as Long

'Get the application object --

'Get Property
lIsotropicMaterialCount = objOpenSTAAD.Property.GetIsotropicMaterialCount
```

Property.GetIsotropicMaterialProperties

VB Syntax

Property.GetIsotropicMaterialProperties (long MatNo, double E, double Poisson, double G, double Density, double Alpha, double CrDamp)

Parameters

MatNo

Long variable holds the material reference number.

E, Poisson, G, Density, Alpha, CrDamp

Double variables return isotropic material properties.

Remarks

Returns the properties for the specified isotropic material number.

Example

```
'Get the application object --

'Get Property
objOpenSTAAD.Property.GetIsotropicMaterialProperties (lMatNo, dE, dPoisson, dG,
dDensity, dAlpha, dCrDamp)
```

Property.GetOrthotropic2DMaterialCount

VB Syntax

Property.GetOrthotropic2DMaterialCount ()

Remarks

Returns number of 2D orthotropic material present in the current structure.

Example

```
Dim lOrthotropic2DMaterialCount as Long

'Get the application object --

'Get Property
lOrthotropic2DMaterialCount = objOpenSTAAD.Property.GetOrthotropic2DMaterialCount
```

Property.GetOrthotropic2DMaterialProperties

VB Syntax

Property.GetOrthotropic2DMaterialProperties (long MatNo, double EArray, double PoissonArray, double GArray, double DensityArray, double AlphaArray, double CrDampArray)

Parameters

MatNo

Long variable holds the material reference number.

E, Poisson, G, Density, Alpha, CrDamp

Double array (dimension 2) variables return 2D orthotropic material properties.

Remarks

Returns the properties for the specified isotropic material number.

Example

```
'Get the application object --

'Get Property
objOpenSTAAD.Property.GetOrthotropic2DMaterialProperties (lMatNo, dEArray,
dPoissonArray, dGArray, dDensityArray, dAlphaArray, dCrDampArray)
```

Property.GetOrthotropic3DMaterialCount

VB Syntax

Property.GetOrthotropic3DMaterialCount ()

Remarks

Returns number of 3D orthotropic material present in the current structure.

Example

```
Dim lOrthotropic3DMaterialCount as Long

'Get the application object --

'Get Property
lOrthotropic3DMaterialCount = objOpenSTAAD.Property.GetOrthotropic3DMaterialCount
```

Property.GetOrthotropic3DMaterialProperties

VB Syntax

Property.GetOrthotropic3DMaterialProperties (long MatNo, double EArray, double PoissonArray, double GArray, double DensityArray, double AlphaArray, double CrDampArray)

Parameters

MatNo

Long variable holds the material reference number.

E, Poisson, G, Density, Alpha, CrDamp

Double array (dimension 3) variables return 3D orthotropic material properties.

Remarks

Returns the properties for the specified isotropic material number.

Example

```
'Get the application object --

'Get Property
objOpenSTAAD.Property.GetOrthotropic3DMaterialProperties (lMatNo, dEArray,
dPoissonArray, dGArray, dDensityArray, dAlphaArray, dCrDampArray)
```

Property.GetMemberGlobalOffSet

VB Syntax

Property.GetMemberGlobalOffSet (long BeamNo, integer End, double xOffSet, double yOffSet, double zOffSet)

Parameters

BeamNo

Long variable holds the member reference number.

End

Integer value specifies the end of the member.

0 = Start

1 = End

xOffSet, yOffSet, zOffSet

Double variables return offset value of the specified member in all direction at the specified end.

Remarks

Returns beam end offsets in all three global directions.

Example

```
'Get the application object --

'Get Property
objOpenSTAAD.Property.GetMemberGlobalOffSet (lBeamNo, iEnd, dxOffSet, dyOffSet,
dzOffSet)
```

Property.GetMemberLocalOffSet

VB Syntax

Property.GetMemberLocalOffSet (long BeamNo, integer End, double xOffSet, double yOffSet, double zOffSet)

Parameters

BeamNo

Long variable holds the member reference number.

End

Integer value specifies the end of the member.

0 = Start

1 = End

xOffSet, yOffSet, zOffSet

Double variables return offset value of the specified member in all three local directions at the specified end.

Remarks

Returns beam end offsets in all three local directions.

Example

```
'Get the application object --

'Get Property
objOpenSTAAD.Property.GetMemberLocalOffSet (lBeamNo, iEnd, dxOffSet, dyOffSet,
dzOffSet)
```

Property.GetMemberReleaseSpec

VB Syntax

Property.GetMemberReleaseSpec (long BeamNo, integer End, long ReleaseArray, long SpringConstArray)

Parameters

BeamNo

Long variable holds the member reference number.

End

Integer value specifies the end of the member.

0 = Start

1 = End

ReleaseArray, SpringConstArray

Long array returns translational and rotational releases.

Remarks

Returns releases for the specified member at the specified end.

Example

```
'Get the application object --

'Get Property
objOpenSTAAD.Property.GetMemberReleaseSpec (lBeamNo, iEnd, lReleaseArray,
lSpringConstArray
```

Loads Applications

Load.CreateNewPrimaryLoad

VB Syntax

long Load.CreateNewPrimaryLoad (string NewPrimaryLoadTitle)

Return Value

Long integer containing the load number created.

Parameters

NewPrimaryLoadTitle

String variable providing the title of the load case.

Remarks

Creates new PRIMARY load case.

Example

```
'Get the application object --

'Create new load
objOpenSTAAD.Load.CreateNewPrimaryLoad "My Load"
```

Load.SetLoadActive

VB Syntax

bool Load.SetLoadActive (integer LoadNo)

Return Value

TRUE if successful, else FALSE.

Parameters

LoadNo

Integer variable providing the load number which will be made active.

Remarks

Make the specified load number active.

Example

```
'Get the application object --

'make load case 1 active
objOpenSTAAD.Load.SetLoadActive 1
```

Load.AddSelfWeightInXYZ

VB Syntax

bool Load.AddSelfWeightInXYZ (integer Direction, double LoadFactor)

Return Value

TRUE if successful else FALSE.

Parameters

Direction

Integer variable giving the direction of selfweight: X direction = 1, Y direction = 2, Z direction = 3.

LoadFactor

Double variable providing the multiplying factor for selfweight.

Remarks

Adds SELFWEIGHT of the structure in X or Y or Z direction to the active load case.

Example

```
'Get the application object --

'Add selfweight in negative Y direction
objOpenSTAAD.Load.AddSelfWeightInXYZ 2, -1.0
```

Load.AddNodalLoad

VB Syntax

bool Load.AddNodalLoad (integer NodeNo, double ForceX, double ForceY, double ForceZ, double MomentX, double MomentY, double MomentZ)

bool Load.AddNodalLoad (integer NodeNoArray, double ForceX, double ForceY, double ForceZ, double MomentX, double MomentY, double MomentZ)

Return Value

TRUE if successful else FALSE.

Parameters

NodeNo

Integer variable providing the node number.

NodeNoArray

Integer variable array providing the node numbers.

ForceX, ForceY, ForceZ, MomentX, MomentY, MomentZ

Double variables providing the load values in individual directions.

Remarks

Adds JOINT LOAD to the specified node number or numbers.

Example

```
 'Get the application object --

 'Add joint load of 2units in X direction at node number 2
objOpenSTAAD.Load.AddNodalLoad 2, 2.0, 0.0, 0.0, 0.0, 0.0, 0.0
```

Load.AddSupportDisplacement

VB Syntax

bool Load.AddSupportDisplacement (integer NodeNo, integer Direction, double Displacement)

bool Load.AddSupportDisplacement (integer NodeNoArray, integer Direction, double Displacement)

Return Value

TRUE if successful else FALSE.

Parameters

NodeNo

Integer variable providing the node number.

NodeNoArray

Integer variable array providing the node numbers.

Direction

Integer variable giving the direction of displacement: X direction = 1, Y direction = 2, Z direction = 3.

Displacement

Double variable providing the magnitude of the support displacement in current units.

Remarks

Adds SUPPORT DISPLACEMENT to node or nodes.

Example

```
'Get the application object --

'Add joint load of 2mm displacement to node 2 in global X
objOpenSTAAD.Load.AddSupportDisplacement 2, 1, 2.0
```

Load.AddMemberUniformForce

VB Syntax

bool Load.AddMemberUniformForce (integer BeamNo, integer Direction, double ForceValue, double D1Value, double D2Value, double D3Value)

bool Load.AddMemberUniformForce (integer BeamNoArray, integer Direction, double ForceValue, double D1Value, double D2Value, double D3Value)

Return Value

TRUE if successful else FALSE.

Parameters

BeamNo

Integer variable providing the beam number.

BeamNoArray

Integer variable array providing the beam numbers.

Direction

Integer variable giving the direction of load: X direction = 1, Y direction = 2, Z direction = 3, GX direction = 4, GY direction = 5, GZ direction = 6, PX direction = 7, PY direction = 7, PZ direction = 8.

ForceValue

Double variable providing the magnitude of the uniform force in current units.

D1Value, D2Value, D3Value

Double variable providing the value of d1, d2, d3 in current units. For additional information, please refer to the STAAD.*Pro* Technical Reference manual.

Remarks

Adds UNIFORM FORCE to beam or beams.

Example

```
'Get the application object --

'Add member uniform load of 2 units to member 2 in GY direction
objOpenSTAAD.Load.AddMemberUniformForce 2, 5, 2.0, 0.0, 0.0, 0.0
```

Load.AddMemberUniformMoment

VB Syntax

bool Load.AddMemberUniformMoment (integer BeamNo, integer Direction, double MomentValue, double D1Value, double D2Value, double D3Value)

bool Load.AddMemberUniformMoment (integer BeamNoArray, integer Direction, double MomentValue, double D1Value, double D2Value, double D3Value)

Return Value

TRUE if successful else FALSE.

Parameters

BeamNo

Integer variable providing the beam number.

BeamNoArray

Integer variable array providing the beam numbers.

Direction

Integer variable giving the direction of load: X direction = 1, Y direction = 2, Z direction = 3, GX direction = 4, GY direction = 5, GZ direction = 6, PX direction = 7, PY direction = 7, PZ direction = 8.

MomentValue

Double variable providing themagnitude of the uniform moment in current units.

D1Value, D2Value, D3Value

Double variable providing value of d1, d2, d3 in current units. For additional information, please refer to the STAAD.*Pro* Technical Reference manual.

Remarks

Adds UNIFORM MOMENT to beam or beams.

Example

```
'Get the application object --

'Add member uniform moment of 2 units to member 2 in GY direction
objOpenSTAAD.Load.AddMemberUniformMoment 2, 5, 2.0, 0.0, 0.0, 0.0
```

Load.AddMemberConcForce

VB Syntax

bool Load.AddMemberConcForce (integer BeamNo, integer Direction, double ForceValue, double D1Value, double D2Value)

bool Load.AddMemberConcForce (integer BeamNoArray, integer Direction, double ForceValue, double D1Value, double D2Value)

Return Value

TRUE if successful else FALSE.

Parameters

BeamNo

Integer variable providing the beam number.

BeamNoArray

Integer variable array providing the beam numbers.

Direction

Integer variable giving the direction of load: X direction = 1, Y direction = 2, Z direction = 3, GX direction = 4, GY direction = 5, GZ direction = 6, PX direction = 7, PY direction = 7, PZ direction = 8.

ForceValue

Double variable providing the magnitude of the concentrated force in current units.

D1Value, D2Value

Double variables providing values of d1, d2 in current units. For additional information, please refer to the STAAD.*Pro* Technical Reference manual.

Remarks

Adds CONCENTRATED FORCE to beam or beams.

Example

```
'Get the application object --

'Add member concentrated load of 2 units to member 2 in GY direction
objOpenSTAAD.Load.AddMemberConcForce 2, 5, 2.0, 0.0, 0.0, 0.0
```

Load.AddMemberConcMoment

VB Syntax

bool Load.AddMemberConcMoment (integer BeamNo, integer Direction, double MomentValue, double D1Value, double D2Value)

bool Load.AddMemberConcMoment (integer BeamNoArray, integer Direction, double MomentValue, double D1Value, double D2Value)

Return Value

TRUE if successful else FALSE.

Parameters

BeamNo

Integer variable providing the beam number.

BeamNoArray

Integer variable array providing the beam numbers.

Direction

Integer variable giving the direction of load: X direction = 1, Y direction = 2, Z direction = 3, GX direction = 4, GY direction = 5, GZ direction = 6, PX direction = 7, PY direction = 7, PZ direction = 8.

MomentValue

Double variable providing the magnitude of the concentrated moment in current units.

D1Value, D2Value

Double variable providing value of d1, d2 in current units. For additional information, please refer to the STAAD.*Pro* Technical Reference manual.

Remarks

Adds CONCENTRATED MOMENT to beam or beams.

Example

```
'Get the application object --

'Add member concentrated moment of 2 units to member 2 in GY direction
objOpenSTAAD.Load.AddMemberConcMoment 2, 5, 2.0, 0.0, 0.0, 0.0
```

Load.AddMemberLinearVari

VB Syntax

bool Load.AddMemberLinearVari (integer BeamNo, integer Direction, double StartLoad, double EndLoad, double TriLoad)

bool Load.AddMemberLinearVari (integer BeamNoArray, integer Direction, double StartLoad, double EndLoad, double TriLoad)

Return Value

TRUE if successful else FALSE.

Parameters

BeamNo

Integer variable providing the beam number.

BeamNoArray

Integer variable array providing the beam numbers.

Direction

Integer variable giving the direction of load: X direction = 1, Y direction = 2, Z direction = 3.

StartLoad, EndLoad

Double variable providing the load value at the start and end of the member respectively.

TriLoad

Double variable providing the magnitude of the load value for triangular load. If it has a value other than 0, the load is regarded as a triangular load irrespective of values in *StartLoad* and *EndLoad*.

Remarks

Adds LINEARLY VARYING load to beam or beams.

Example

```
'Get the application object --

'Add member linearly varying to member 2 in GY direction
objOpenSTAAD.Load.AddMemberLinearVari 2, 2, 2.0, 0.0, 0.0
```

Load.AddMemberTrapezoidal

VB Syntax

bool Load.AddMemberTrapezoidal (integer BeamNo, integer Direction, double StartLoad, double EndLoad, double StartDistance, double LoadLength)

bool Load.AddMemberTrapezoidal (integer BeamNoArray, integer Direction, double StartLoad, double EndLoad, double StartDistance, double EndDistance)

Return Value

TRUE if successful else FALSE

Parameters

BeamNo

Integer variable providing the beam number.

BeamNoArray

Integer variable array providing the beam numbers.

Direction

Integer variable giving the direction of load: X direction = 1, Y direction = 2, Z direction = 3, GX direction = 4, GY direction = 5, GZ direction = 6, PX direction = 7, PY direction = 7, PZ direction = 8.

StartLoad, EndLoad

Double variable providing the load value at the start and end of the member respectively.

StartDistance, EndDistance

Double variable providing the start and end distance of the load.

Remarks

Adds trapezoidal linearly varying load to beam or beams.

Example

```
'Get the application object --

'Add member linearly varying to member 2 in GY direction
objOpenSTAAD.Load.AddMemberLinearVari 2, 2, 2.0, 0.0, 0.0
```

Load.AddMemberAreaLoad

VB Syntax

bool Load.AddMemberAreaLoad (integer BeamNo, double AreaLoad)

bool Load.AddMemberAreaLoad (integer BeamNoArray, double AreaLoad)

Return Value

TRUE if successful else FALSE.

Parameters

BeamNo

Integer variable providing the beam number.

BeamNoArray

Integer variable array providing the beam numbers.

AreaLoad

Double variable providing the magnitude of the load value.

Remarks

Adds AREA LOAD to beam or beams.

Example

```
'Get the application object --

'Add area load to member 2
objOpenSTAAD.Load.AddMemberAreaLoad 2, 2.0
```

Load.AddMemberFloorLoad

VB Syntax

Load.AddMemberFloorLoad (double Pressure, double YMIN, double YMAX, double ZMIN, double ZMAX, double XMIN, double XMAX)

Parameters

Pressure

A double value to indicate the pressure intensity to be applied as a floor load.

YMIN, YMAX, ZMIN, ZMAX, XMIN, XMAX

Double values to indicate what the boundary of the floor is. The XMIN, XMAX and ZMIN and ZMAX values must lie on the same XZ plane.

Remarks

Automatically finds enclosed panels in the given boundary and adds a FLOOR LOAD command in the format:

YRANGE *ymin ymax* FLOAD *pressure* XRANGE *xmin xmax* ZRANGE *zmin zmax*

Example

```
'Get the application object --

'Add a floor load with intensity of -5
objOpenSTAAD.Load.AddMemberFloorLoad -5, 2.9, 3.1, 0, 80, 0, 200
```

Load.AddMemberFixedEnd

VB Syntax

bool Load.AddMemberFixedEnd (integer BeamNo, double LoadAtStartArray, double LoadAtEndArray)

bool Load.AddMemberFixedEnd (integer BeamNoArray, double LoadAtStartArray, double LoadAtEndArray)

Return Value

TRUE if successful else FALSE.

Parameters

BeamNo

Integer variable providing the beam number.

BeamNoArray

Integer variable array providing the beam numbers.

LoadAtStartArray, LoadAtEndArray

Double variable arrays of dimension 6 providing the fixed end load values at member start and end. The indices are as follows: 0 = Fx_1, 1 = Fy_1, 2 = Fz_1, 3 = Mx_1, 4 = My_1, 5 = Mz_1 for start and 0 = Fx_2, 1 = Fy_2, 2 = Fz_2, 3 = Mx_2, 4 = My_2, 5 = Mz_2 for end.

Remarks

Adds FIXED END LOAD to beam or beams.

Example

```
Dim start(0 To 5) As Double
Dim end(0 To 5) As Double
'Get the application object --

'Fill up the array accordingly

'Add fixed end load to member 2
objOpenSTAAD.Load.AddMemberFixedEnd 2, start, end
```

Load.AddElementPressure

VB Syntax

bool Load.AddElementPressure (integer PlateNo, integer Direction, double Pressure, double X1, double Y1, double X2, double Y2)

bool Load.AddElementPressure (integer PlateNoArray, integer Direction, double Pressure, double X1, double Y1, double X2, double Y2)

Return Value

TRUE if successful else FALSE.

Parameters

PlateNo

Integer variable providing the plate number.

PlateNoArray

Integer variable array providing the plate numbers.

Direction

Integer variable giving the direction of pressure: Local Z direction = 0, GX direction = 1, GY direction = 2, GZ direction = 3.

Pressure

Double variable providing the magnitude of the pressure over the element.

X1, Y1, X2, Y2

Double variable providing the top left coordinate and bottom right coordinate of concentrated load.

Remarks

Adds pressure load to plate elements.

Example

```
'Get the application object --

'Add element pressure
objOpenSTAAD.Load.AddElementPressure 2, 1, 2.0
```

Load.AddElementTrapPressure

VB Syntax

bool Load.AddElementTrapPressure (integer PlateNo, integer Direction, double StartPressure, double EndPressure)

bool Load.AddElementTrapPressure (integer PlateNoArray, integer Direction, double StartPressure, double EndPressure)

Return Value

TRUE if successful else FALSE.

Parameters

PlateNo

Integer variable providing the plate number.

PlateNoArray

Integer variable array providing the plate numbers.

Direction

Integer variable giving the direction of pressure: Local X direction = 1, Local Y direction = 2.

StartPressure, EndPressure

Double variable providing the pressure at start and end.

Remarks

Adds trapezoidal pressure loading to plate elements.

Example

```
'Get the application object --

'Add element pressure
objOpenSTAAD.Load.AddElementTrapPressure 2, 1, 2.0, 3.0
```

Load.AddTemperatureLoad

VB Syntax

bool Load.AddTemperatureLoad (integer ElementNo, double AxialElongation, double DiffTempTopAndBtm, double DiffTempSideToSide)

bool Load.AddTemperatureLoad (integer ElementNoArray, double AxialElongation, double DiffTempTopAndBtm, double DiffTempSideToSide)

Return Value

TRUE if successful else FALSE.

Parameters

ElementNo

Integer variable providing the member/element number.

ElementNoArray

Integer variable array providing the member/element numbers.

AxialElongation

Double variable providing the temperature causing axial elongation.

DiffTempTopAndBtm

Double variable providing the differential temperature between top and bottom surface.

DiffTempSideToSide

Double variable providing the differential temperature from side to side.

Remarks

Adds TEMPERATURE LOAD to beam or plate elements.

Example

```
'Get the application object --

'Add temperature load
objOpenSTAAD.Load.AddTemperatureLoad 2, 10.0, 20.0, 30.0
```

Load.AddStrainLoad

VB Syntax

bool Load.AddStrainLoad (integer ElementNo, double AxialStrain)

bool Load.AddStrainLoad (integer ElementNoArray, double AxialStrain)

Return Value

TRUE if successful else FALSE.

Parameters

ElementNo

Integer variable providing the member/element number.

ElementNoArray

Integer variable array providing the member/element numbers.

AxialStrain

Double variable providing the strain value due to misfit.

Remarks

Adds STRAIN LOAD to beam or plate elements.

Example

```
'Get the application object --

'Add strain load
objOpenSTAAD.Load.AddStrainLoad 2, 0.01
```

Load.GetPrimaryLoadCaseCount

VB Syntax

Load.GetPrimaryLoadCaseCount ()

Remarks

Gets total number of primary load case(s) present in the current structure.

Example

```
Dim lGetPrimaryLoadCaseCount as Long

'Get the application object --

'Get Primary Load Case Count
lGetPrimaryLoadCaseCount = objOpenSTAAD.Load.GetPrimaryLoadCaseCount
```

Load.GetLoadCombinationCaseCount

VB Syntax

Load.GetLoadCombinationCaseCount ()

Remarks

Gets total number of combination load case(s) present in the current structure.

Example

```
Dim lGetLoadCombinationCaseCount as Long

'Get the application object --

'Get Combination Load Case Count
lGetLoadCombinationCaseCount = objOpenSTAAD.Load.GetLoadCombinationCaseCount
```

Load.GetPrimaryLoadCaseNumbers

VB Syntax

Load.GetPrimaryLoadCaseNumbers (long PrimaryLoadCaseNumbersArray)

Parameters

PrimaryLoadCaseNumbersArray

A long array wich stores the load case numbers for all the primary load cases present in the current structure.

Remarks

Gets all primary load case number(s).

Example

```
Dim lGetPrimaryLoadCaseCount as Long
Dim lPrimaryLoadCaseNumbersArray() as Long

'Get the application object --

'Get Primary Load Case Numbers
lGetPrimaryLoadCaseCount = objOpenSTAAD.Load.GetPrimaryLoadCaseCount
 for I = 0 to (lGetPrimaryLoadCaseCount-1)
   lPrimaryLoadCaseNumbersArray (I) = objOpenSTAAD.Load.
GetPrimaryLoadCaseNumbers
 next I
```

Load.GetLoadCombinationCaseNumbers

VB Syntax

Load.GetLoadCombinationCaseNumbers (long CombineLoadCaseNumbersArray)

Parameters

CombineLoadCaseNumbersArray

A long array wich stores the load case numbers for all the primary load cases present in the current structure.

Remarks

Gets all primary load case number(s).

Example

```
Dim lGetLoadCombinationCaseCount as Long
Dim lLoadCombinationCaseNumbersArray() as Long

'Get the application object --

'Get Combination Load Case Numbers
lGetLoadCombinationCaseCount = objOpenSTAAD.Load.GetLoadCombinationCaseCount
 for I = 0 to (lGetLoadCombinationCaseCount-1)
   lLoadCombinationCaseNumbersArray (I) =
objOpenSTAAD.Load.GetLoadCombinationCaseNumbers
 next I
```

Load.CreateNewLoadCombination

VB Syntax

Load.CreateNewLoadCombination (string LoadCombTitle, long LoadCombNo)

Parameters

LoadCombTitle

A string variable, which defines tittle for new load combination.

LoadCombNo

A long variable, which defines number for new load combination.

Remarks

Creates new load combination with the number and title defined.

Example

```
Dim strLoadCombTitle as String
Dim lLoadCombNo as Long

'Get the application object --

'Create New Load Combination
objOpenSTAAD.Load.CreateNewLoadCombination strLoadCombTitle, lLoadCombNo
```

Load.AddLoadAndFactorToCombination

VB Syntax

Load.AddLoadAndFactorToCombination (integer LoadCombNo, long LoadNo, float LoadFactor)

Parameters

LoadCombNo

An integer variable holds the load combination number.

LoadNo

A long variable holds the primary load case number.

LoadFactor

Multiplication factor for the specified primary load case.

Remarks

Adds a primary load case with specified multiplication factor to an existing load combination.

Example

```
'Get the application object --
Dim intLoadCombNo as Integer
Dim lLoadNo as Long
Dim fFactor as Float

'Add Load to Load Combination
objOpenSTAAD.Load.AddLoadAndFactorToCombination intLoadCombNo, lLoadNo, fFactor
```

Load.GetBeamCountAtFloor

VB Syntax

Load.GetBeamCountAtFloor (float MinX, float MaxX, float MinY, float MaxY, float MinZ, float MaxZ, integer Direction)

Parameters

MinX, MaxX, MinY, MaxY, MinZ, MaxZ

Float variables indicate what the boundary of the floor pane.

Direction

Integer variable provide the direction.

1 = X-axis

2 = Y-axis

3 = Z-axis

Remarks

Get total no of beams within the specified boundary.

Example

```
Dim fMinX as Float
Dim fMaxX as Float
Dim fMinY as Float
Dim fMaxY as Float
Dim fMinZ as Float
Dim fMaxZ as Float
Dim intDirection as Integer

'Get the application object --

'Get beam count at floor
objOpenSTAAD.Load.GetBeamCountAtFloor fMinX, fMaxX, fMinY, fMaxY, fMinZ, fMaxZ,
intDirection
```

Load.GetInfluenceArea

VB Syntax

Load.GetInfluenceArea (float MinX, float MaxX, float MinY, float MaxY, float MinZ, float MaxZ, integer Direction, long BeamNosArray, Double AreasArray)

Parameters

MinX, MaxX, MinY, MaxY, MinZ, MaxZ

Float variables indicate what the boundary of the floor pane.

Direction

Integer variable provide the direction.

1 = X-axis

2 = Y-axis

3 = Z-axis

BeamNosArray

Long array returns all the beam nos under consideration.

AreasArray

Double array returns all the influence areas for the beams returned in the lastparameter.

Remarks

Returns beam nos and corresponding influence areas within the specified boundary.

Example

```
'Get the application object --

'Get Influence Area
objOpenSTAAD.Load.GetInfluenceArea (fMinX, fMaxX, fMinY, fMaxY, fMinZ, fMaxZ,
iDirection, lBeamNosArray, dAreasArray)
```

Load.GetActiveLoad

VB Syntax

Load.GetActiveLoad()

Remarks

Returns the current load case number.

Example

```
Dim lActiveLoad as Long

'Get the application object --

'Get active load
lActiveLoad = objOpenSTAAD.GetActiveLoad
```

Load.GetNodalLoadCount

VB Syntax

Load.GetNodalLoadCount (long NodeNo)

Parameters

NodeNo

Long variable provides the node number.

Remarks

Get number of nodal loads present for the specified node.

Example

```
Dim iNodalLoadCount as Integer
Dim lNodeNo as Long

'Get the application object --

'Get Nodal Load Count
iNodalLoadCount = objOpenSTAAD.Load.GetNodalLoadCount ( lNodeNo )
```

Load.GetNodalLoads

VB Syntax

Load.GetNodalLoads (integer NodeNo, double FXArray, double FYArray, double FZArray, double MXArray, double MYArray, double MZArray)

Parameters

NodeNo

Integer variable holds the node number for which nodal loads needs to be retrieved.

FXArray, FYArray, FZArray, MXArray, MYArray, MZArray

Double variables return the load array for all the load cases and all directions.

Remarks

Retruns the array of load values for the specified node. Array will be formed and dimensioned as per defined load counts.

Example

```
'Get the application object --

'Get Nodal Loads
objOpenSTAAD.Load.GetNodalLoads (iNodeNo, dFXArray, dFYArray, dFZArray, dMXArray,
dMYArray, dMZArray)
```

Load.GetUDLLoadCount

VB Syntax

Load.GetUDLLoadCount (long BeamNo)

Parameters

BeamNo

Long variable provides the beam number.

Remarks

Get number of uniformly distributed loads present for the specified beam.

Example

```
'Get the application object --

'Get UDL count
objOpenSTAAD.Load.GetUDLLoadCount (lBeamNo)
```

Load.GetUDLLoads

VB Syntax

Load.GetUDLLoads (long BeamNo, long DirectionArray, double ForceArray, double D1Array, double D2Array, double D3Array)

Parameters

BeamNo

Long variable providing the member number.

DirectionArray, ForceArray, D1Array, D2Array, D3Array

Returns the force value, direction alongwith d1, d2 & d3 parameter for beam UDL.

Remarks

Returns the UDL with all the parameters for the specified member.

Example

```
'Get the application object --

'Get UDL value
objOpenSTAAD.Load.GetUDLLoads (lBeamNo, lDirectionArray, dForceArray, dD1Array,
dD2Array, dD3Array)
```

Load.GetUNIMomentCount

VB Syntax

Load.GetUNIMomentCount (long BeamNo)

Parameters

BeamNo

Long variable providing the member number.

Remarks

Gets the count of uniformly distributed moment for the specified member.

Example

```
'Get the application object --

'Get UMOM count
objOpenSTAAD.Load.GetUNIMomentCount (lBeamNo)
```

Load.GetUNIMoments

VB Syntax

Load.GetUNIMoments (long BeamNo, long DirectionArray, double ForceArray, double D1Array, double D2Array, double D3Array)

Parameters

BeamNo

Long variable providing the member number.

DirectionArray, ForceArray, D1Array, D2Array, D3Array

Returns the value, direction alongwith d1, d2 & d3 parameter for beam uniformly distributed moments.

Remarks

Returns the uni moments with all the parameters for the specified member.

Example

```
'Get the application object --

'Get UMOM value
objOpenSTAAD.Load.GetUNIMoments (lBeamNo, lDirectionArray, dForceArray, dD1Array,
dD2Array, dD3Array)
```

Load.GetTrapLoadCount

VB Syntax

Load.GetTrapLoadCount (long BeamNo)

Parameters

BeamNo

Long variable provides the beam number.

Remarks

Get number of trapezoidal loads present for the specified beam.

Example

```
'Get the application object --

'Get Trapezoidal Load count
objOpenSTAAD.Load.GetTrapLoadCount (lBeamNo)
```

Load.GetTrapLoads

VB Syntax

Load.GetTrapLoads (long BeamNo, long DirectionArray, double W1Array, double W2Array, double D1Array, double D2Array)

Parameters

BeamNo

Long variable providing the member number.

DirectionArray, W1Array, W2Array, D1Array, D2Array

Returns the force value, direction alongwith w1, w2, d1 & d2 parameters for beam trapezoidal load(s).

Remarks

Returns the trapezoidal load(s) with all the parameters for the specified member.

Example

```
'Get the application object --

'Get Trapezoidal Load values
objOpenSTAAD.Load.GetTrapLoads (lBeamNo, lDirectionArray, dW1Array, dW2Array,
dD1Array, dD2Array)
```

Load.GetConcForceCount

VB Syntax

Load.GetConcForceCount (long BeamNo)

Parameters

BeamNo

Long variable provides the beam number.

Remarks

Get number of concentrated loads present for the specified beam.

Example

```
'Get the application object --

'Get Concentrated Load count
objOpenSTAAD.Load.GetConcForceCount (lBeamNo)
```

Load.GetConcForces

VB Syntax

Load.GetConcForces (long BeamNo, long DirectionArray, double ForceArray, double D1Array, double D2Array)

Parameters

BeamNo

Long variable providing the member number.

DirectionArray, ForceArray, D1Array, D2Array

Returns the force value, direction alongwith d1 & d2 parameters for beam trapezoidal load(s).

Remarks

Returns the concentrated forces with all the parameters for the specified member.

Example

```
'Get the application object --

'Get Concentrated Load values
objOpenSTAAD.Load.GetConcForces (lBeamNo, lDirectionArray, dForceArray, dD1Array,
dD2Array)
```

Load.GetConcMomentCount

VB Syntax

Load.GetConcMomentCount (long BeamNo)

Parameters

BeamNo

Long variable provides the beam number.

Remarks

Get number of concentrated moment present for the specified beam.

Example

```
'Get the application object --

'Get Conc Moment Count
objOpenSTAAD.Load.GetConcMomentCount (lBeamNo)
```

Load.GetConcMoments

VB Syntax

Load.GetConcMoments (long BeamNo, long DirectionArray, double MomentArray, double D1Array, double D2Array)

Parameters

BeamNo

Long variable providing the member number.

DirectionArray, MomentArray, D1Array, D2Array

Returns the moment value, direction alongwith d1 & d2 parameters for beam concentrated moment(s).

Remarks

Returns the beam concentrated moment(s) with all the parameters for the specified member.

Example

```
'Get the application object --

'Get Conc Moment Value
objOpenSTAAD.Load.GetConcMoments (lBeamNo, lDirectionArray, dMomentArray,
dDlArray, dD2Array)
```

Supports Applications

Support.CreateSupportFixed

VB Syntax

long Support.CreateSupportFixed ( )

Return Value

Long integer containing the reference number to the support created.

Remarks

Creates FIXED support.

Example

```
'Get the application object --

'Create fixed support
objOpenSTAAD.Support.CreateSupportFixed
```

Support.CreateSupportPinned

VB Syntax

long Support.CreateSupportPinned ( )

Return Value

Long integer containing the reference number to the support created.

Remarks

Creates PINNED support.

Example

```
'Get the application object --

'Create pinned support
objOpenSTAAD.Support.CreateSupportPinned
```

Support.CreateSupportFixedBut

VB Syntax

long Support.CreateSupportFixedBut (double ReleaseCodeArray, double SpringConstantArray)

Return Value

Long integer containing the reference number to the support created.

Parameters

ReleaseCodeArray

Double array of 6 elements specifying releases in FX, FY, FZ, MX, MY, MZ directions. Each element should have a value equal to 0.0 (Fixed) or 1.0 (Released).

SpringConstantArray

Double array of 6 elements specifying spring constants in FX, FY, FZ, MX, MY, MZ directions.

Values should be in the current unit system.

Remarks

Creates FIXED support with releases in specified directions or SPRING supports with spring constants in specified directions.

Example

```
'Get the application object --

'Create fixed support with release in Y direction
release (0) = 0.0 'FX
release (1) = 1.0 'FY
release (2) = 0.0 'FZ
release (3) = 0.0 'MX
release (4) = 0.0 'MY
release (5) = 0.0 'MZ
suppno = staad.support.CreateSupportFixedBut release
```

Support.AssignSupportToNode

VB Syntax

bool Support.AssignSupportToNode (long NodeNo, long SupportNo)

bool Support.AssignSupportToNode (long NodeNoArray, long SupportNo)

Return Value

TRUE if successful, else FALSE..

Parameters

NodeNo

Long variable providing the node number.

NodeNoArray

Long variable array providing the node numbers.

SupportNo

Long variable providing the reference number of the support

Remarks

Assigns the specified support to node or nodes.

Example

```
'Get the application object --

'assign support 2 to node 1
bResult = objOpenSTAAD.Support.AssignSupportToNode 1, 2
```

Support.GetSupportCount

VB Syntax

Support.GetSupportCount ()

Remarks

Returns total number of supported nodes exist in the current structure.

Example

```
'Get the application object --

'Get Support Count
objOpenSTAAD.Support.GetSupportCount
```

Support.GetSupportNodes

VB Syntax

Support.GetSupportNodes (long SupportNodesArray)

Parameters

SupportNodesArray

A long Array stores the numbers of the nodes, which are supported in the current structure.

Remarks

Returns all supported nodes in an array.

Example

```
Dim
Dim lSupportNodesArray() as Long
Dim iSupportCount as Integer
'Get the application object --


iSupportCount = Support.GetSupportCount

ReDim lSupportNodesArray(0 to (iSupportCount-1)) as Long
'Get Support Nodes
objOpenSTAAD.Support.GetSupportNodes(lSupportNodesArray)
```

Support.GetSupportType

VB Syntax

Support.GetSupportType (long SupportNode)

Parameters

SupportNode

Long variable providing the supported node number.

Remarks

Returns back the support type for the specified node (1 = Pinned, 2 = Fixed, 3 = Fixed But).

Example

```
'Get the application object --

'Get Support Type
objOpenSTAAD.Support.GetSupportType (lSupportNode)
```

Support.GetSupportInformation

VB Syntax

Support.GetSupportInformation (long NodeNo, integer ReleaseSpecArray, double SpringSpecArray)

Parameters

NodeNo

Long variable providing the node number.

ReleaseSpecArray

Integer array of dimension 6 retrives. (0 = Released, 1 = Not Released).

SpringSpecArray

Integer array of dimension 6 retrives for the spring stiffness of each dof.

Remarks

Returns support information for the specified node.

Example

```
'Get the application object --

'Get Support Information
objOpenSTAAD.Support. Support.GetSupportInformation (lNodeNo, iReleaseSpecArray,
dSpringSpecArray)
```

Command Applications

Command.PerformAnalysis

VB Syntax

Command.PerformAnalysis (integer PrintOption)

Parameters

PrintOption

An integer variable, which specify the user's choice of print command with Performa Analysis command. The values may vary as:

0 = No Print,

1 = Print Load Data

2 = Print Statics Check

3 = Print Statics Load

4 = Print Mode Shapes

5 = Print Both

6 = Print All

Remarks

Perform first order elastic analysis of the current structure and produce output with specified print option.

Example

```
Dim intPrintOption as Integer

'Get the application object --

'Perform Analysis
objOpenSTAAD.Command.PerformAnalysis intPrintOption
```

Command.PerformPDeltaAnalysisNoConverge

VB Syntax

Command.PerformPDeltaAnalysisNoConverge (integer NumberOfIteration, integer PrintOption)

Parameters

NumberOfIteration

An integer variable, which specify the no of iteration to be performed.

PrintOption

An integer variable, which specify the user's choice of print command with P-Delta analysis command. The values may vary as:

0 = No Print,

1 = Print Load Data

2 = Print Statics Check

3 = Print Statics Load

4 = Print Mode Shapes

5 = Print Both

6 = Print All

Remarks

Perform second order P-Delta analysis of the current structure and produce output with specified print option. Here the program will perform the specified number of iterations whether or not the solution converges.

Example

```
Dim intNumberOfIteration as Integer
Dim intPrintOption as Integer

'Get the application object --
```

```
'P-Delta Analysis no Converge
objOpenSTAAD.Command.PerformPDeltaAnalysisNoConverge intNumberOfIteration,
intPrintOption
```

Command.PerformPDeltaAnalysisConverge

VB Syntax

Command.PerformPDeltaAnalysisConverge (integer NumberOfIteration, integer PrintOption)

Parameters

NumberOfIteration

An integer variable, which specify the no of iteration to be performed.

PrintOption

An integer variable, which specify the user's choice of print command with P-Delta Analysis command. The values may vary as:

0 = No Print,

1 = Print Load Data

2 = Print Statics Check

3 = Print Statics Load

4 = Print Mode Shapes

5 = Print Both

6 = Print All

Remarks

Perform second order P-Delta analysis of the current structure and produce output with specified print option. Here the program checks whether the solution is converging as successive iterations are performed. If the successive iterations result in divergent values of displacements for a particular load case, then the iterations are discontinued for the specific load case.

Example

```
Dim intNumberOfIteration as Integer
```

```
Dim intPrintOption as Integer

'Get the application object --

' P-Delta Analysis with Converge
objOpenSTAAD.Command.PerformPDeltaAnalysisNoConverge intNumberOfIteration,
intPrintOption
```

Command.CreateSteelDesignCommand

VB Syntax

Command.CreateSteelDesignCommand (long DesignCode, long CommandNo, integer IntValuesArray, float FloatValuesArray, string StringValuesArray, long AssignList)

Remarks

Create steel design command.

Output Results Applications

Output.GetOutputUnitForDimension

VB Syntax

Output.GetOutputUnitForDimension (string Unit)

Parameters

Unit

A string containing the output unit for dimension.

Remarks

Returns output unit for dimension.

Example

```
Dim strOutputUnitForDimension as String

'Get the application object --

'Get Output Unit For Dimension
strOutputUnitForDimension = objOpenSTAAD.Output.GetOutputUnitForDimension
```

Output.GetOutputUnitForSectDimension

VB Syntax

Output.GetOutputUnitForSectDimension (string Unit)

Parameters

Unit

A string containing the output unit for sectional diemension(s).

Remarks

Returns output unit for cross sectional diemension(s).

Example

```
Dim strOutputUnitForSectDimension as String

'Get the application object --

'Get Output Unit For Sectional Dimension
strOutputUnitForSectDimension = objOpenSTAAD.Output.GetOutputUnitForSectDimension
```

Output.GetOutputUnitForSectArea

VB Syntax

Output.GetOutputUnitForSectArea (string Unit)

Parameters

Unit

A string containing the output unit for cross sectional area.

Remarks

Returns output unit for cross sectional area.

Example

```
Dim strOutputUnitForSectArea as String

'Get the application object --

'Get Output Unit For Sectional Area
strOutputUnitForSectArea = objOpenSTAAD.Output.GetOutputUnitForSectArea
```

Output.GetOutputUnitForSectInertia

VB Syntax

Output.GetOutputUnitForSectInertia (string Unit)

Parameters

Unit

A string containing the output unit for sectional inertia.

Remarks

Returns output unit for sectional intertia.

Example

```
Dim strOutputUnitForSectInertia as String

'Get the application object --

'Get Output Unit For Sectional Inertia
strOutputUnitForSectInertia = objOpenSTAAD.Output.GetOutputUnitForSectInertia
```

Output.GetOutputUnitForSectModulus

VB Syntax

Output.GetOutputUnitForSectModulus (string Unit)

Parameters

Unit

A string containing the output unit for sectional modulus.

Remarks

Returns output unit for sectional modulus.

Example

```
Dim strOutputUnitForSectModulus as String

'Get the application object --

'Get Output Unit For Sectional Modulus
strOutputUnitForSectModulus = objOpenSTAAD.Output.GetOutputUnitForSectModulus
```

Output.GetOutputUnitForDensity

VB Syntax

Output.GetOutputUnitForDensity (string Unit)

Parameters

Unit

A string containing the output unit for material density.

Remarks

Returns output unit for material density.

Example

```
Dim strOutputUnitForDensity as String

'Get the application object --

'Get Output Unit For Material Density
strOutputUnitForDensity = objOpenSTAAD.Output.GetOutputUnitForDensity
```

Output.GetOutputUnitForDisplacement

VB Syntax

Output.GetOutputUnitForDisplacement (string Unit)

Parameters

Unit

A string containing the output unit for translational displacements.

Remarks

Returns output unit for translational displacements..

Example

```
Dim strOutputUnitForDisplacement as String

'Get the application object --

'Get Output Unit For Displacement
strOutputUnitForDisplacement = objOpenSTAAD.Output.GetOutputUnitForDisplacement
```

Output.GetOutputUnitForRotation

VB Syntax

Output.GetOutputUnitForRotation (string Unit)

Parameters

Unit

A string containing the output unit for rotational displacement.

Remarks

Returns output unit for rotational displacement.

Example

```
Dim strOutputUnitForRotation as String

'Get the application object --

'Get Output Unit For Rotation
strOutputUnitForRotation = objOpenSTAAD.Output.GetOutputUnitForRotation
```

Output.GetOutputUnitForForce

VB Syntax

Output.GetOutputUnitForForce (string Unit)

Parameters

Unit

A string containing the output unit for forces (FX, FY, FZ).

Remarks

Returns output unit for forces (FX, FY, FZ).

Example

```
Dim strOutputUnitForForce as String

'Get the application object --

'Get Output Unit For Force
strOutputUnitForForce = objOpenSTAAD.Output.GetOutputUnitForForce
```

Output.GetOutputUnitForMoment

VB Syntax

Output.GetOutputUnitForMoment (string Unit)

Parameters

Unit

A string containing the output unit for moments (MX, MY, FZ).

Remarks

Returns output unit for moments (MX, MY, MZ).

Example

```
Dim strOutputUnitForMoment as String

'Get the application object --

'Get Output Unit For Moment
strOutputUnitForMoment = objOpenSTAAD.Output.GetOutputUnitForMoment
```

Output.GetOutputUnitForDistForce

VB Syntax

Output.GetOutputUnitForDistForce (string Unit)

Parameters

Unit

A string containing the output unit for distributed forces.

Remarks

Returns output unit for distributed forces.

Example

```
Dim strOutputUnitForDistForce as String

'Get the application object --

'Get Output Unit For Distributed Force
strOutputUnitForDistForce = objOpenSTAAD.Output.GetOutputUnitForDistForce
```

Output.GetOutputUnitForDistMoment

VB Syntax

Output.GetOutputUnitForDistMoment (string Unit)

Parameters

Unit

A string containing the output unit for distributed moments.

Remarks

Returns output unit for distributed moments.

Example

```
Dim strOutputUnitForDistMoment as String

'Get the application object --

'Get Output Unit For Distributed Moment
strOutputUnitForDistMoment = objOpenSTAAD.Output.GetOutputUnitForDistMoment
```

Output.GetOutputUnitForStress

VB Syntax

Output.GetOutputUnitForStress (string Unit)

Parameters

Unit

A string containing the output unit for stress.

Remarks

Returns output unit for stress.

Example

```
Dim strOutputUnitForStress as String

'Get the application object --

'Get Output Unit For Stress
strOutputUnitForStress = objOpenSTAAD.Output.GetOutputUnitForStress
```

Output.GetNodeDisplacements

VB Syntax

Output.GetNodeDisplacements (long NodeNo, long LoadCase, long DisplacementArray)

Parameters

NodeNo

Long variable contains the node number.

LoadCase

Long variable contains the load case number.

DisplacementArray

A long array of dimension 6, which returns nodal displacements.

Remarks

Returns nodal displacements for the node number and load case specified.

Example

```
'Get the application object --

'Get Nodal Displacement
objOpenSTAAD.Output.GetNodeDisplacements (lNodeNo, lLoadCase, lDisplacementArray)
```

Output.GetSupportReactions

VB Syntax

Output.GetSupportReactions (long NodeNo, long LoadCase, long ReactionArray)

Parameters

NodeNo

Long variable contains the node number, which is supported.

LoadCase

Long variable contains the load case number.

ReactionArray

A long array of dimension 6, which returns support reactions.

Remarks

Returns support reactions for the node number and load case specified.

Example

```
'Get the application object --

'Get Support Reaction
objOpenSTAAD.Output.GetSupportReactions (lNodeNo, lLoadCase, lReactionArray)
```

Output.GetMemberEndDisplacements

VB Syntax

Output.GetMemberEndDisplacements (long MemberNo, long End, long LoadCase, long DisplacementArray)

Parameters

MemberNo

Long variable contains the member number.

End

Long variable contains member end.

0 = Start

1 = End

LoadCase

Long variable contains the load case number.

DisplacementArray

A long array of dimension 6, which returns member end displacements.

Remarks

Returns member end displacements for specified member number, member end and load case.

Example

```
'Get the application object --

'Get Member End Displacements
objOpenSTAAD.Output.GetMemberEndDisplacements (lMemberNo, lEnd, lLoadCase,
lDisplacementArray)
```

Output.GetIntermediateMemberTransDisplacements

VB Syntax

Output.GetIntermediateMemberTransDisplacements (long MemberNo, double Distance, long LoadCase, long DisplacementArray)

Parameters

MemberNo

Long variable contains the member number.

Distance

Double variable contains distance from starting end in terms of member length.

LoadCase

Long variable contains the load case number.

DisplacementArray

A long array of dimension 6, which returns member sectional displacements.

Remarks

Returns sectional displacements for specified member number, distance and load case.

Example

```
'Get the application object --

'Get Member Intermediate Displacements
objOpenSTAAD.Output.GetIntermediateMemberTransDisplacements (lMemberNo,
dDistance, lLoadCase, lDisplacementArray)
```

Output.GetMemberEndForces

VB Syntax

Output.GetMemberEndForces (long MemberNo, long End, long LoadCase, long ForceArray)

Parameters

MemberNo

Long variable contains the member number.

End

Long variable contains member end.

0 = Start

1 = End

LoadCase

Long variable contains the load case number.

ForceArray

A long array of dimension 6, which returns member end forces.

Remarks

Returns member end forces for specified member number, member end and load case.

Example

```
'Get the application object --

'Get Member End Forces
objOpenSTAAD.Output.GetMemberEndForces (lMemberNo, lEnd, lLoadCase, lForceArray)
```

Output.GetAllPlateCenterStressesAndMoments

VB Syntax

Output.GetAllPlateCenterStressesAndMoments (long PlateNo, long LoadCase, long CenterStressesArray)

Parameters

PlateNo

A long variable contains plate element no.

LoadCase

A long variable contains load case no.

CenterStressesArray

A long array of dimension 8, which returns plate center stresses and moments.

Remarks

Returns plate center stresses and moments for the specified plate for specified load case.

Example

```
'Get the application object --

'Get All Plate Center Stresses And Moments
objOpenSTAAD.Output.GetAllPlateCenterStressesAndMoments (lPlateNo, lLoadCase,
lCenterStressesArray)
```

Output.GetPlateCenterNormalPrincipalStresses

VB Syntax

Output.GetPlateCenterNormalPrincipalStresses (long PlateNo, long LoadCase, double SMAXTop, double SMINTop, double SMAXBottom, double SMINBottom)

Parameters

PlateNo

A long variable contains plate element no.

LoadCase

A long variable contains load case no.

SMAXTop, SMINTop, SMAXBottom, SMINBottom

Double variables, which return the corresponding values for the center of the specified plate.

Remarks

Returns Smax and Smin for Top and Bottom of the specified plate.

Example

```
'Get the application object --

'Get Plate Center Normal Principal Stresses
objOpenSTAAD.Output.GetPlateCenterNormalPrincipalStresses (lPlateNo, lLoadCase,
dSMAXTop, dSMINTop, dSMAXBottom, dSMINBottom)
```

Output.GetAllPlateCenterForces

VB Syntax

Output.GetAllPlateCenterForces (long PlateNo, long LoadCase, long CenterForcesArray)

Parameters

PlateNo

A long variable contains plate element no.

LoadCase

A long variable contains load case no.

CenterForcesArray

A long array of dimension 5, which returns plate center forces.

Remarks

Returns plate center forces for the specified plate and load case.

Example

```
'Get the application object --

'Get Plate Center Forces
objOpenSTAAD.Output.GetAllPlateCenterForces (lPlateNo, lLoadCase,
lCenterForcesArray)
```

Output.GetAllPlateCenterMoments

VB Syntax

Output.GetAllPlateCenterMoments (long PlateNo, long LoadCase, long CenterMomentsArray)

Parameters

PlateNo

A long variable contains plate element no.

LoadCase

A long variable contains load case no.

CenterMomentsArray

A long array of dimension 3, which returns plate center moments.

Remarks

Returns plate center moments for the specified plate and load case.

Example

```
'Get the application object --

'Get Plate Center Moments
objOpenSTAAD.Output.GetAllPlateCenterMoments (lPlateNo, lLoadCase,
lCenterMomentsArray)
```

Output.GetAllPlateCenterPrincipalStressesAndAngles

VB Syntax

Output.GetAllPlateCenterPrincipalStressesAndAngles (long PlateNo, long LoadCase, long StressesAndAnglesArray)

Parameters

PlateNo

A long variable contains plate element no.

LoadCase

A long variable contains load case no.

StressesAndAnglesArray

A long array of dimension 8, which returns plate center stresses and angles.

Remarks

Returns plate center stresses and angles for the specified plate and load case.

Example

```
'Get the application object --

'Get Plate Center Principal Stresses and Angles
objOpenSTAAD.Output.GetAllPlateCenterPrincipalStressesAndAngles (lPlateNo,
lLoadCase, lStressesAndAnglesArray)
```

Output.GetPlateCenterVonMisesStresses

VB Syntax

Output.GetPlateCenterVonMisesStresses (long PlateNo, long LoadCase, double VONT, double VONB)

Parameters

PlateNo

A long variable contains plate element no.

LoadCase

A long variable contains load case no.

VONT, VONB

Double variables return the value of Von-Mises Top and Bottom for the specified plate center point.

Remarks

Returns plate center Von-Mises Top and Bottom for the specified plate and load case.

Example

```
'Get the application object --

'Get Plate Center Center Von-Mises Stresses
objOpenSTAAD.Output.GetPlateCenterVonMisesStresses (lPlateNo, lLoadCase, dVONT,
dVONB)
```

Output.GetAllSolidNormalStresses

VB Syntax

Output.GetAllSolidNormalStresses (long SolidNo, long Corner, long LoadCase, long StressesArray)

Parameters

SolidNo

A long variable contains solid element no.

Corner

A long variable specifies the corner of that solid element.

LoadCase

A long variable contains load case no.

StressesArray

A long array of dimension 3, which returns solid corner stresses.

Remarks

Returns normal stresses for the specified solid element at the specified corner.

Example

```
'Get the application object --

'Get Solid Corner Normal Stresses
objOpenSTAAD.Output.GetAllSolidNormalStresses (lSolidNo, lCorner, lLoadCase,
lStressesArray)
```

Output.GetAllSolidShearStresses

VB Syntax

Output.GetAllSolidShearStresses (long SolidNo, long Corner, long LoadCase, long ShearStressesArray)

Parameters

SolidNo

A long variable contains solid element no.

Corner

A long variable specifies the corner of that solid element. (-1 = Center, 1 to 8 all corner nodes).

LoadCase

A long variable contains load case no.

ShearStressesArray

A long array of dimension 3, which returns solid corner shear stresses.

Remarks

Returns shear stresses for the specified solid element at the specified corner.

Example

```
'Get the application object --

'Get Solid Corner Shear Stresses
objOpenSTAAD.Output.GetAllSolidShearStresses (lSolidNo, lCorner, lLoadCase,
lShearStressesArray)
```

Output.GetAllSolidPrincipalStresses

VB Syntax

Output.GetAllSolidPrincipalStresses (long SolidNo, long Corner, long LoadCase, long PrincipalStressesArray)

Parameters

SolidNo

A long variable contains solid element no.

Corner

A long variable specifies the corner of that solid element. (-1 = Center, 1 to 8 all corner nodes).

LoadCase

A long variable contains load case no.

PrincipalStressesArray

A long array of dimension 3, which returns solid corner principal stresses.

Remarks

Returns principal stresses for the specified solid element at the specified corner.

Example

```
'Get the application object --

'Get Solid Corner Principal Stresses
objOpenSTAAD.Output.GetAllSolidPrincipalStresses (lSolidNo, lCorner, lLoadCase,
lPrincipalStressesArray)
```

Output.GetAllSolidVonMisesStresses

VB Syntax

Output.GetAllSolidVonMisesStresses (long SolidNo, long Corner, long LoadCase, long VonMisesStressesArray)

Parameters

SolidNo

A long variable contains solid element no.

Corner

A long variable specifies the corner of that solid element. (-1 = Center, 1 to 8 all corner nodes).

LoadCase

A long variable contains load case no.

VonMisesStressesArray

A long array of dimension 3, which returns solid corner Von mises stresses.

Remarks

Returns normal stresses for the specified solid element at the specified corner.

Example

```
'Get the application object --

'Get Solid Corner Von Stresses
objOpenSTAAD.Output.GetAllSolidVonMisesStresses (lSolidNo, lCorner, lLoadCase,
lVonMisesStressesArray)
```

Results Tables Applications

Table.CreateReport

VB Syntax

long Table.CreateReport (string szReportTitle)

Return Value

A reference number for the report created to be used to access the report.

Parameters

szReportTitle

A null terminated string containing the title of the report.

Remarks

Creates a report with the specified title.

Example

```
'Get the application object --

'Create report
ReportNo = objOpenSTAAD.Table.CreateReport "My Report"
```

Table.SaveReport

VB Syntax

Table.SaveReport (long ReportNo)

Parameters

ReportNo

A long variable providing the report number identifying the STAAD report which is to be saved.

Remarks

Saves the specified report along with all its tables.

Example

```
'Get the application object --

'Save report
objOpenSTAAD.Table.SaveReport ReportNo
```

Table.SaveReportAll

VB Syntax

Table.SaveReportAll ( )

Remarks

Saves all the reports created.

Example

```
'Get the application object --

'Save all reports
objOpenSTAAD.Table.SaveReportAll
```

Table.GetReportCount

VB Syntax

long Table.GetReportCount ( )

Return Value

Number of reports created.

Remarks

Returns the number of reports created.

Example

```
'Get the application object --

'get number of reports
ReportNos = objOpenSTAAD.Table.GetReportCount
```

Table.AddTable

VB Syntax

long Table.AddTable (long ReportNo, string szTableName, long NumRows, long NumCols)

Return Value

A reference number for the table created to be used to access the table.

Parameters

ReportNo

Long variable containing the report number to which this table will be added.

szTableName

A null terminated string containing the name of the table.

NumRows, NumCols

Long variables providing the number of rows and columns of the table.

Remarks

Adds a table to the specified *ReportNo*.

Example

```
'Get the application object --

'Add table to report no 1 with 10 rows and 5 columns
NumRows = 10
NumCols = 5
TableNo = objOpenSTAAD.Table.AddTable 1, "My Table", NumRows, NumCols
```

Table.RenameTable

VB Syntax

Table.RenameTable (long ReportNo, long TableNo, string szNewTableName)

Parameters

ReportNo

Long variable containing the report number whose table will be renamed.

TableNo

Long variable containing the table number to be renamed.

szNewTableName

A null terminated string containing the new name for the table.

Remarks

Renames a table in a report specified by *TableNo*.

Example

```
'Get the application object --

'Rename Table
objOpenSTAAD.Table.RenameTable ReportNo, TableNo, "My New Table"
```

Table.DeleteTable

VB Syntax

Table.DeleteTable (long ReportNo, long TableNo)

Parameters

ReportNo

Long variable containing the report number from which a table is to be deleted.

TableNo

Long variable containing the table number to be deleted.

Remarks

Deletes a table specified by *TableNo* in a report specified by *ReportNo*.

Example

```
'Get the application object --

'Delete Table
objOpenSTAAD.Table.DeleteTable ReportNo, TableNo
```

Table.ResizeTable

VB Syntax

Table.ResizeTable (long ReportNo, long TableNo, long NumRows, long NumCols)

Parameters

ReportNo

Long variable containing the report number whose table will be resized.

TableNo

Long variable containing the table number to be resized.

NumRows, NumCols

Long variables providing the new number of rows and columns of the table.

Remarks

Resizes a table specified by *TableNo* in a report specified by *ReportNo*.

Example

```
'Get the application object --

'Resize Table
NumRows = 10
NumCols = 5
objOpenSTAAD.Table.ResizeTable ReportNo, TableNo, NumRows, NumCols
```

Table.SaveTable

VB Syntax

Table.SaveTable (long ReportNo, long TableNo)

Parameters

ReportNo

Long variable containing the report number whose table will be saved.

TableNo

Long variable containing the table number to be saved.

Remarks

Saves a table in a report specified by *TableNo*.

Example

```
'Get the application object --

'Save Table
objOpenSTAAD.Table.SaveTable ReportNo, TableNo
```

Table.GetTableCount

VB Syntax

long Table.GetTableCount (long ReportNo)

Return Value

Number of tables in the report.

Parameters

ReportNo

Long variable containing the report number.

Remarks

Returns the number of tables created.

Example

```
'Get the application object --

'get number of reports
TableNos = objOpenSTAAD.Table.GetTableCount ReportNo
```

Table.SetCellValue

VB Syntax

Table.SetCellValue (long ReportNo, long TableNo, long RowNo, long ColNo, Data_Type Value)

Parameters

ReportNo

Long variable containing the report number.

TableNo

Long variable containing the table number.

RowNo, ColNo

Long variables containing the row and column number of the cell.

Value

A variable of *Data_Type* (Integer, Long, Double, String) containing the value to be inserted in the cell.

Remarks

Puts a value into the cell of table at the specified row and column in a report.

Example

```
'Get the application object --

'Set value to cell
RowNo = 2
ColNo = 5
Value = 5.25     'declared as Double
objOpenSTAAD.Table.SetCellValue ReportNo, TableNo, Rowno, ColNo, Value
```

Table.GetCellValue

VB Syntax

Table.GetCellValue (long ReportNo, long TableNo, long RowNo, long ColNo, string szValue)

Return Value

Copies the data in the specified cell into the string provided.

Parameters

ReportNo

Long variable containing the report number.

TableNo

Long variable containing the table number.

RowNo, ColNo

Long variables containing the row and column number of the cell.

szValue

String variable in which the value in the cell will be copied .

Remarks

Gets a value in the cell of table at the specified row and column in a report.

Example

```
Dim szValue As String
 'Get the application object --

 'Set value to cell
RowNo = 2
ColNo = 5
objOpenSTAAD.Table.GetCellValue ReportNo, TableNo, Rowno, ColNo, szValue
```

Table.SetColumnHeader

VB Syntax

Table.SetColumnHeader (long ReportNo, long TableNo, long ColNo, string szHeader)

Parameters

ReportNo

Long variable containing the report number.

TableNo

Long variable containing the table number.

ColNo

Long variable containing the column number.

szHeader

String variable providing the column header.

Remarks

Sets the heading of a column.

Example

```
'Get the application object --

'Set header for the column
Header = "Column 5"
ColNo = 5
objOpenSTAAD.Table.SetColumnHeader ReportNo, TableNo, ColNo, szHeader
```

Table.SetColumnUnitString

VB Syntax

Table.SetColumnUnitString (long ReportNo, long TableNo, long ColNo, string szUnitString)

Parameters

ReportNo

Long variable containing the report number.

TableNo

Long variable containing the table number.

ColNo

Long variable containing the column number.

szUnitString

String variable providing the column unit string.

Remarks

Sets the unit string of a column.

Example

```
'Get the application object --

'Set unit for the column
szUnit = "kN/m^2"
ColNo = 5
objOpenSTAAD.Table.SetColumnUnitString ReportNo, TableNo, ColNo, szUnit
```

Table.SetRowHeader

VB Syntax

Table.SetRowHeader (long ReportNo, long TableNo, long RowNo, string szHeader)

Parameters

ReportNo

Long variable containing the report number.

TableNo

Long variable containing the table number.

RowNo

Long variable containing the row number.

szHeader

String variable providing the row header.

Remarks

Sets the heading of a row.

Example

```
'Get the application object --

'Set header for the row
Header = "Row 5"
RowNo = 5
objOpenSTAAD.Table.SetRowHeader ReportNo, TableNo, RowNo, szHeader
```

Table.SetCellTextColor

VB Syntax

Table.SetCellTextColor (long ReportNo, long TableNo, long RowNo, long ColNo, integer RedVal, integer GreenVal, integer BlueVal)

Parameters

ReportNo

Long variable containing the report number.

TableNo

Long variable containing the table number.

RowNo

Long variable containing the row number.

ColNo

Long variable containing the column number.

RedVal, GreenVal, BlueVal

An integer between 0 and 255 that represents the intensity of red, green or blue in the color for the text. A value of 0 for a particualr color indicates that no shade of that color will be mixed into the final color. To have the text written in yellow, the values would be RedVal = 255, GreenVal = 255 and BlueVal = 0.

Remarks

Sets the color of the text to be displayed in a particular cell. By default, the color is always set to black.

Example

```
'Get the application object --

Dim RedVal As Integer
Dim GreenVal As Integer
Dim BlueVal As Integer

'Set the text to yellow
```

```
RedVal = 255
GreenVal = 255
BlueVal = 0

objOpenSTAAD.Table.SetCellTextColor ReportNo, TableNo, RowNo, ColNo, RedVal,
GreenVal, BlueVal
```

Table.SetCellTextBold

VB Syntax

Table.SetCellTextBold (long ReportNo, long TableNo, long RowNo, long ColNo)

Parameters

ReportNo

Long variable containing the report number.

TableNo

Long variable containing the table number.

RowNo

Long variable containing the row number.

ColNo

Long variable containing the column number.

Remarks

Sets the text in a given row and column to bold.

Example

```
'Get the application object --
‘Set the text in row 9 and column 4 to bold
objOpenSTAAD.Table.SetCellTextBold ReportNo, TableNo, 9, 4
```

Table.SetCellTextItalic

VB Syntax

Table.SetCellTextItalic (long ReportNo, long TableNo, long RowNo, long ColNo)

Parameters

ReportNo

Long variable containing the report number.

TableNo

Long variable containing the table number.

RowNo

Long variable containing the row number.

ColNo

Long variable containing the column number.

Remarks

Italicizes the text in a given row and column.

Example

```
'Get the application object --
‘Italicize the text in row 9 and column 4
objOpenSTAAD.Table.SetCellTextItalic ReportNo, TableNo, 9, 4
```

Table.SetCellTextUnderline

VB Syntax

Table.SetCellTextUnderline (long ReportNo, long TableNo, long RowNo, long ColNo)

Parameters

ReportNo

Long variable containing the report number.

TableNo

Long variable containing the table number.

RowNo

Long variable containing the row number.

ColNo

Long variable containing the column number.

Remarks

Underlines the text in a given row and column.

Example

```
'Get the application object --
‘Underline the text in row 9 and column 4
objOpenSTAAD.Table.SetCellTextUnderline ReportNo, TableNo, 9, 4
```

Table.SetCellTextSize

VB Syntax

Table.SetCellTextSize (long ReportNo, long TableNo, long RowNo, long ColNo, double FontSize)

Parameters

ReportNo

Long variable containing the report number.

TableNo

Long variable containing the table number.

RowNo

Long variable containing the row number.

ColNo

Long variable containing the column number.

FontSize

Double variable containing the size of the font to set the text to.

Remarks

Sets the text in a particular row and column to a certain font size. The font sizes are equivalent to the ones used in Microsoft Word.

Example

```
'Get the application object --
‘Set the font size for the text in row 10 and column 6 to 18 pt
objOpenSTAAD.Table.SetCellTextSize ReportNo, TableNo, 10, 6, 16.0
```

Table.SetCellTextSizeAll

VB Syntax

Table.SetCellTextSizeAll (long ReportNo, long TableNo, double FontSize)

Parameters

ReportNo

Long variable containing the report number.

TableNo

Long variable containing the table number.

FontSize

Double variable containing the size of the font to set the entire table to.

Remarks

Sets the text in the entire table to FontSize. The font sizes are equivalent to the ones used in Microsoft Word.

Example

```
'Get the application object --
‘Set the font size for the text in the table to 16 pt
objOpenSTAAD.Table.SetCellTextSizeAll ReportNo, TableNo, 16.0
```

Table.SetCellTextHorzAlignment

VB Syntax

Table.SetCellTextHorzAlignment (long ReportNo, long TableNo, long RowNo, long ColNo, integer Alignment)

Parameters

ReportNo

Long variable containing the report number.

TableNo

Long variable containing the table number.

RowNo

Long variable containing the row number.

ColNo

Long variable containing the column number.

Alignment

An integer variable containing the type of horizontal alignment to apply. The possible values are:

0 = left

1 = center

2 = right

Remarks

Sets the text in a particular row and column to a specified horizontal alignment. By default, all the text is right justified.

Example

```
'Get the application object --

‘Set the horizontal alignment for the text in row 10 and column 6 to center-     ‘
justified.

objOpenSTAAD.Table.SetCellTextHorzAlignment ReportNo, TableNo, 10, 6, 1
```

Table.SetCellTextVertAlignment

VB Syntax

Table.SetCellTextVertAlignment (long ReportNo, long TableNo, long RowNo, long ColNo, integer Alignment)

Parameters

ReportNo

Long variable containing the report number.

TableNo

Long variable containing the table number.

RowNo

Long variable containing the row number.

ColNo

Long variable containing the column number.

Alignment

An integer variable containing the type of vertical alignment to apply. The possible values are:

0 = top

4 = center

8 = bottom

Remarks

Sets the text in a particular row and column to a specified vertical alignment. By default, all the text is top justified.

Example

```
'Get the application object --

‘Set the horizontal alignment for the text in row 10 and column 6 to center-     ‘
justified.
```

OpenSTAAD Functions – Troubleshooting

Section 4

Errors and Error Messages

Error message: "Method 'OpenSTAADFunctionName' of object IOutput failed.", where OpenSTAADFunctionName is the name of an OpenSTAAD function, e.g., "Method 'GetMemberBetaAngle' of object IOutput failed."

Check to be sure that you are passing all parameters required by the function. If a parameter is missing, or if the parameter being passed is not valid, the 'Method …of object IOutput failed' message may appear. For example, the message may appear if a member number is passed to the function, but that member number does not exist in the currently open STAAD file.

You may also receive similar messages if your version of STAAD.Pro is not compatible with OpenSTAAD. OpenSTAAD is compatible with STAAD.Pro 2001 Build 1006.US.REL and higher. You will need to reanalyze your model in that build. This is because we switched the way the data is recorded in Build 1006 to save more space. Build 1006 US.REL was released in November 2001.

UK Builds prior to STAAD.Pro 2002 are not compatible with OpenSTAAD.

Function is not retrieving correct values.

Check to be sure that you have saved the STAAD file after making changes to the input before you run any OpenSTAAD functions that retrieve input information. If you are running STAAD functions that retrieve analysis results, check to be sure that you have saved the STAAD file and re-run the analysis after making any changes to the input.

Error message: "Type mismatch"

Check to make sure that you have declared all variables using the DIM statement at the beginning of your program or macro, e.g.:

```
Dim pnIsReleased As Integer

Dim pdSpringStiffnesses(0 To 5) As Double

Dim pdPartialMomRelFactors(0 To 2) As Double
```

Confirm that when you pass array variables to the function, you have specified the starting position in the array for the function to use in filling up the array, e.g.:

```
objOpenSTAAD.GetFullMemberReleaseInfoAtStar  3,  pnIsReleased,  _
                                             pdSpringStiffnesses(0),  pdPartialMomRelFactors(0)
```

Error message: “Object doesn’t support this property or method”

Check to make sure you have typed the function name correctly.

Error message: "ActiveX component can't create object" when attempting to run an OpenSTAAD macro in Microsoft Excel.

1. Make sure that you have a directory called "OpenSTAAD" inside of your Spro2004 folder (i.e. C:\Spro2004\ OpenSTAAD).

2. In that folder, make sure you have the files openstaad.dll, staadread.tbl, staadread.dll and steeltable.dll. If not, please go to http://www.reiworld.com/Product/Pro/OpenSTAAD.asp and click on the "OpenSTAAD Setup Program" link near the bottom of the page. This will reinstall the OpenSTAAD component.

3. When you open the Excel file, is it asking you to *Enable Macros*? If so, make sure you press *Yes*. If not, you need to change the security settings in Excel. Go to Tools->Macro->Security... from the Excel menu. Make sure the *Security Level* is set to *Medium*. Close Excel down (completely, not just the file) and reopen the file again. This time it should ask you if you want your Macros enabled or disabled. Choose *Enabled*.

4. When you run your macro, if it still gives you the error ("ActiveX component can't create object"), it might be because the OpenSTAAD library was not registered properly when the program was installed. To register the

OpenSTAAD dll, close down Excel, go to your Windows *Start* menu and select the *Run* command. Type in:

regsvr32 c:\spro2004\openstaad\openstaad.dll

and click the *OK* button. (Please note that the path to the openstaad.dll file may be different on your computer if you did not use the default installation path provided by the STAAD.*Pro* Setup program when you installed your software – if so, you should modify the above command to correspond to the actual location on your computer.) A dialog box should display the message, "DllRegisterServer in c:\spro2004\openstaad\openstaad.dll succeeded." If the registration did not succeed, please contact our technical support staff for further instructions.

5. Try opening and running the Excel beam example file provided with your STAAD.Pro software (C:\Spro2004\OpenSTAAD\Rectangle-beam .xls).

Error message: "User Defined type not defined"

This message "User Defined type not defined" may appear on the line which declares the OpenSTAAD object variable if the variable is declared "As Output", e.g. "Dim objOpenSTAAD As Output". The message may be appearing because the OpenSTAAD library references have not been included. The VBA compiler therefore does not know which functions are associated with the OpenSTAAD object.

There are two ways to eliminate this error message:

1. Declare the OpenSTAAD object As Object, instead of As Output, e.g.

Dim objOpenSTAAD As Object

2. Include the OpenSTAAD library reference in your VBA editor. This second option has the added benefit that once you do it, the compiler will now recognize the OpenSTAAD object and will pop up a list of functions whenever you refer to the object in your VBA editor.

To include the OpenSTAAD library reference, pull down the *Tools* menu in your VBA editor and select the *References* command. A dialog box with title "References – Normal" will open. You will see a scroll box inside the dialog box labeled "Available References." Scroll down through the list of references until you

find one called “OpenSTAAD 1.0 Type Library”. Toggle on the corresponding check box, then click the *OK* button.

Now re-run your macro to see if the problem with the “User Defined type not defined” error message has been solved.

Error message: "One or more results files are not compatible with the current model and cannot be loaded..."

You need to have a successful analysis before you can run any OpenSTAAD macro, including such simple macros as one that only asks for the number of nodes in the model. The program is not smart enough to know which commands will work and which will not when no results are present. It first checks to see if valid results are available. If they are not, it displays the error message and terminates.

You can use the OpenSTAAD command AreResultsAvailable in your macro to check to see if results are available. In some cases, this function also will result in the error message "One or more results files are not compatible with the current model and cannot be loaded...". That means that even though the STAAD results file with an ANL extension has been created, the file contains only error messages and no meaningful results.

You can also try inserting a FINISH command in your input file preceding the location in the input that is causing the analysis errors. Depending on whether the errors take place before any analysis is performed, you might get rid of the original error message, but in its place you may see the message, "No Results Available." However, the best way to eliminate this problem is to modify your input file so that you obtain a successful analyis before you try to run your OpenSTAAD macro.

Under certain conditions, you might get this “…results files are not compatible…” error message, even though the analysis runs successfully. OpenSTAAD currently will not work with the Moving Load Generator. The reason is that the results from the Moving Load Generator are not kept in the same database as the other STAAD results. Therefore, when OpenSTAAD tries to read the Output File, it is not able to find those missing load results, so it displays an error message and stops processing the macro. This same situation is in effect for any other loads that you are unable to see until you perform the analysis, like UBC loads, for example. If the loads are defined in the input file, OpenSTAAD will work fine, but not with loads that are only generated when the analysis is run, because these results are not kept in a location where OpenSTAAD can find them.

If you remove your moving load generation command from your input file, and run the analysis, you should then be able to run your OpenSTAAD macro.

The capability to handle projects with program-generated loads such as Moving Loads and UBC Loads will be added to a future release of OpenSTAAD.

List of Acronyms

API	Application Program Interface
ATL	Active Template Library
COM	Component Object Model
FTP	File Transfer Protocol
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
I/O	Input/Output
URL	Uniform Resource Locator
VB	Visual Basic
VBA	Visual Basic for Applications

Index of Functions