Improvements from SM Toolbox 2.5.13 to SM Toolbox 2.0.0

A toolbox for connecting SAP2000(CSiBRiDGE) and MATLAB applications

R.Javanmardi

21-Oct-2020

Chapter 1. ConstraintDef

1.1. ChangeName

1.1.1. Remarks

1.1.2. Syntax

[ret]=SM.ConstraintDef.ChangeName(Name,NewName)

1.2. Count

1.2.1. Remarks

1.2.2. Syntax

[ret]=SM.ConstraintDef.Count(...)

1.3. Delete

1.3.1. Remarks

The function deletes the specified constraint. All constraint assignments for that constraint are also deleted.

1.3.2. Syntax

[ret]=SM.ConstraintDef.Delete(Name)

1.4. GetBeam

1.4.1. Remarks

1.4.2. Syntax

[ret,CSys]=SM.ConstraintDef.GetBeam(Name)

1.5. GetBody

1.5.1. Remarks

1.5.2. Syntax

[ret,Value,CSys]=SM.ConstraintDef.GetBody(Name)

1.6. GetConstraintType

1.6.1. Remarks

1.6.2. Syntax

[ret,ConstraintType]=SM.ConstraintDef.GetConstraintType(Name)

1.7. GetDiaphragm

1.7.1. Remarks

1.7.2. Syntax

[ret,Axis,CSys]=SM.ConstraintDef.GetDiaphragm(Name)

1.8. GetEqual

1.8.1. Remarks

1.8.2. Syntax

[ret,Value,CSys]=SM.ConstraintDef.GetEqual(Name)

1.9. GetLine

1.9.1. Remarks

1.9.2. Syntax

[ret,Value,CSys]=SM.ConstraintDef.GetLine(Name)

1.10. GetLocal

1.10.1. Remarks

1.10.2. Syntax

[ret,Value]=SM.ConstraintDef.GetLocal(Name)

1.11. GetNameList

1.11.1. Remarks

This function retrieves the names of all defined joint constraints.

1.11.2. Syntax

[ret,NumberNames,MyName]=SM.ConstraintDef.GetNameList()

1.12. GetPlate

1.12.1. Remarks

1.12.2. Syntax

[ret,Axis,CSys]=SM.ConstraintDef.GetPlate(Name)

1.13. GetRod

1.13.1. Remarks

1.13.2. Syntax

[ret,Axis,CSys]=SM.ConstraintDef.GetRod(Name)

1.14. GetSpecialRigidDiaphragmList

1.14.1. Remarks

This function retrieves the list of the names of each special rigid diaphragm constraint. A special rigid diaphragm constraint is required for assignment of auto seismic load diaphragm eccentricity overwrites. It is also required for calculation of auto wind loads whose exposure widths are determined from the extents of rigid diaphragms.

A special rigid diaphragm constraint is a constraint with the following features:

1. The constraint type is Diaphragm

2. The constraint coordinate system is Global.

3. The constraint axis is Z.

1.14.2. Syntax

[ret,Num,Diaph]=SM.ConstraintDef.GetSpecialRigidDiaphragmList()

1.15. GetWeld

1.15.1. Remarks

1.15.2. Syntax

[ret,Value,Tolerance,CSys]=SM.ConstraintDef.GetWeld(Name)

1.16. SetBeam

1.16.1. Remarks

This function defines a Beam constraint. If the specified name is not used for a constraint, a new constraint is defined using the specified name. If the specified name is already used for another Beam constraint, the definition of that constraint is modified. If the specified name is already used for some constraint that is not a Beam constraint, an error is returned.

1.16.2. Syntax

[ret]=SM.ConstraintDef.SetBeam(Name,...)

1.17. SetBody

1.17.1. Remarks

This function defines a Body constraint. If the specified name is not used for a constraint, a new constraint is defined using the specified name. If the specified name is already used for another Body constraint, the definition of that constraint is modified. If the specified name is already used for some constraint that is not a Body constraint, an error is returned.

1.17.2. Syntax

[ret]=SM.ConstraintDef.SetBody(Name,Value,...)

1.18. SetDiaphragm

1.18.1. Remarks

This function defines a Diaphragm constraint. If the specified name is not used for a constraint, a new constraint is defined using the specified name. If the specified name is already used for another Diaphragm constraint, the definition of that constraint is modified. If the specified name is already used for some constraint that is not a Diaphragm constraint, an error is returned.

1.18.2. Syntax

[ret]=SM.ConstraintDef.SetDiaphragm(Name,...)

1.19. SetEqual

1.19.1. Remarks

This function defines an Equal constraint. If the specified name is not used for a constraint, a new constraint is defined using the specified name. If the specified name is already used for another Equal constraint, the definition of that constraint is modified. If the specified name is already used for some constraint that is not an Equal constraint, an error is returned.

1.19.2. Syntax

[ret]=SM.ConstraintDef.SetEqual(Name,Value,...)

1.20. SetLine

1.20.1. Remarks

This function defines a Line constraint. If the specified name is not used for a constraint, a new constraint is defined using the specified name. If the specified name is already used for another Line constraint, the definition of that constraint is modified. If the specified name is already used for some constraint that is not a Line constraint, an error is returned.

1.20.2. Syntax

[ret]=SM.ConstraintDef.SetLine(Name,Value,...)

1.21. SetLocal

1.21.1. Remarks

This function defines a Local constraint. If the specified name is not used for a constraint, a new constraint is defined using the specified name. If the specified name is already used for another Local constraint, the definition of that constraint is modified. If the specified name is already used for some constraint that is not a Local constraint, an error is returned.

1.21.2. Syntax

[ret]=SM.ConstraintDef.SetLocal(Name,Value)

1.22. SetPlate

1.22.1. Remarks

This function defines a Plate constraint. If the specified name is not used for a constraint, a new constraint is defined using the specified name. If the specified name is already used for another Plate constraint, the definition of that constraint is modified. If the specified name is already used for some constraint that is not a Plate constraint, an error is returned.

1.22.2. Syntax

[ret]=SM.ConstraintDef.SetPlate(Name,...)

1.23. SetRod

1.23.1. Remarks

This function defines a Rod constraint. If the specified name is not used for a constraint, a new constraint is defined using the specified name. If the specified name is already used for another Rod constraint, the definition of that constraint is modified. If the specified name is already used for some constraint that is not a Rod constraint, an error is returned.

1.23.2. Syntax

[ret]=SM.ConstraintDef.SetRod(Name,...)

1.24. SetWeld

1.24.1. Remarks

This function defines a Weld constraint. If the specified name is not used for a constraint, a new constraint is defined using the specified name. If the specified name is already used for another Weld constraint, the definition of that constraint is modified. If the specified name is already used for some constraint that is not a Weld constraint, an error is returned.

1.24.2. Syntax

[ret]=SM.ConstraintDef.SetWeld(Name,Value,Tolerance,...)

Chapter 2. GDispl

2.1. Add

2.1.1. Remarks

This function adds a new generalized displacement with the specified name and type.

The new generalized displacement must have a different name from all other generalized displacements. If the name is not unique, an error will be returned.

2.1.2. Syntax

[ret]=SM.GDispl.Add(Name,MyType)

2.2. ChangeName

2.2.1. Remarks

The new generalized displacement name must be different from all other generalized displacement names. If the name is not unique, an error will be returned.

2.2.2. Syntax

[ret]=SM.GDispl.ChangeName(Name,NewName)

2.3. CountPoint

2.3.1. Remarks

This function retrieves the total number of point objects included in a specified generalized displacement.

2.3.2. Syntax

[ret,Count]=SM.GDispl.CountPoint(Name)

2.4. Delete

2.4.1. Remarks

This function deletes the specified generalized displacement.

2.4.2. Syntax

[ret]=SM.GDispl.Delete(Name)

2.5. DeletePoint

2.5.1. Remarks

This function deletes one point object from a generalized displacement definition.

2.5.2. Syntax

[ret]=SM.GDispl.DeletePoint(Name,PointName)

2.6. GetNameList

2.6.1. Remarks

This function retrieves the names of all defined generalized displacements.

2.6.2. Syntax

[ret,NumberNames,MyName]=SM.GDispl.GetNameList()

2.7. Count

2.7.1. Remarks

2.7.2. Syntax

[ret]=SM.GDispl.Count()

2.8. GetTypeOAPI

2.8.1. Remarks

This function retrieves the generalized displacement type.

2.8.2. Syntax

[ret,MyType]=SM.GDispl.GetTypeOAPI(Name)

2.9. SetPoint

2.9.1. Remarks

This function adds a point object and its scale factors to a generalized displacement definition, or, if the point object already exists in the generalized displacement definition, it modifies the scale factors.

2.9.2. Syntax

[ret]=SM.GDispl.SetPoint(Name,PointName,SF)

2.10. GetPoint

2.10.1. Remarks

This function retrieves the point objects and their scale factors from a generalized displacement definition.

2.10.2. Syntax

[ret,NumberItems,PointName,U1,U2,U3,R1,R2,R3]=SM.GDispl.GetPoint(Name)

2.11. SetTypeOAPI

2.11.1. Remarks

This function sets the generalized displacement type.

2.11.2. Syntax

[ret]=SM.GDispl.SetTypeOAPI(Name,MyType)

Chapter 3. GroupDef

3.1. ChangeName

3.1.1. Remarks

Changing the name of group ALL will fail and return an error.

3.1.2. Syntax

[ret]=SM.GroupDef.ChangeName(Name,NewName)

3.2. Clear

3.2.1. Remarks

This function clears (removes) all assignments from the specified group.

3.2.2. Syntax

[ret]=SM.GroupDef.Clear(Name)

3.3. Count

3.3.1. Remarks

3.3.2. Syntax

[ret]=SM.GroupDef.Count()

3.4. Delete

3.4.1. Remarks

The function deletes the specified group. It will return an error if an attempt is made to delete the Group named ALL.

3.4.2. Syntax

[ret]=SM.GroupDef.Delete(Name)

3.5. GetAssignments

3.5.1. Remarks

This function retrieves the assignments to a specified group.

3.5.2. Syntax

[ret,NumberItems,ObjectType,ObjectName]=SM.GroupDef.GetAssignments(Name)

3.6. GetGroup

3.6.1. Remarks

3.6.2. Syntax

[ret,color,SpecifiedForSelection,SpecifiedForSectionCutDefinition,...

SpecifiedForSteelDesign,SpecifiedForConcreteDesign,SpecifiedForAluminumDesign,...

SpecifiedForColdFormedDesign,SpecifiedForStaticNLActiveStage,...

SpecifiedForBridgeResponseOutput,SpecifiedForAutoSeismicOutput,...

SpecifiedForAutoWindOutput,SpecifiedForMassAndWeight]=SM.GroupDef.GetGroup(Name)

3.7. GetNameList

3.7.1. Remarks

This function retrieves the names of all defined groups.

3.7.2. Syntax

[ret,NumberNames,MyName]=SM.GroupDef.GetNameList()

3.8. SetGroup

3.8.1. Remarks

3.8.2. Syntax

[ret]=SM.GroupDef.SetGroup(Name,...)

Chapter 4. Helper

4.1. CreateObject

4.1.1. Remarks

Starts SAP2000 at the given path and returns an instance of SapObject if successful, nothing otherwise.

4.1.2. Syntax

[Sobj]=SM.Helper.CreateObject(ProgramPath,APIDLLPath)

4.2. CreateObjectProgID

4.2.1. Remarks

Starts SAP2000 and returns an instance of SapObject if successful, nothing otherwise.

The function searches the registry for the newest installed version of SAP2000 unless overridden with an environment variable containing the full path to SAP2000.exe. The overriding environment variable is "CSI_SAP2000_API_SapObject_PATH" for SAP2000 and "CSI_CSiBridge_API_SapObject_PATH" for CSiBridge.

4.2.2. Syntax

[Sobj]=SM.Helper.CreateObjectProgID(ProgID)

4.3. GetOAPIVersionNumber

4.3.1. Remarks

Retrieves the API version. The returned API version can be compared to SapObject.GetOAPIVersionNumber to determine API compatibility with the started/ attached instance of SAP2000.

4.3.2. Syntax

[API_version]=SM.Helper.GetOAPIVersionNumber()

Chapter 5. SectCut

5.1. AddQuad

5.1.1. Remarks

This function adds a new quadrilateral to a section cut defined by quadrilaterals.

5.1.2. Syntax

[ret]=SM.SectCut.AddQuad(Name,X,Y,Z)

5.2. GetCutInfo

5.2.1. Remarks

This function gets basic information about an existing section cut.

5.2.2. Syntax

[ret,GroupName,MyType,Num]=SM.SectCut.GetCutInfo(Name)

5.3. GetLocalAxesAdvancedAnalysis

5.3.1. Remarks

This function gets the advanced local axes data for an existing section cut whose result type is Analysis.

5.3.2. Syntax

[ret,Active,AxVectOpt,AxCSys,axdir,AxPt,AxVect,Plane2,PlVectOpt,...

PlCSys,PlDir,PlPt,PlVect]=SM.SectCut.GetLocalAxesAdvancedAnalysis(Name)

5.4. GetLocalAxesAnalysis

5.4.1. Remarks

This function gets the local axes angles for an existing section cut whose result type is Analysis.

5.4.2. Syntax

[ret,Z,Y,X,IsAdvanced]=SM.SectCut.GetLocalAxesAnalysis(Name)

5.5. GetLocalAxesAngleDesign

5.5.1. Remarks

This function gets the local axes angle for section cuts whose result type is Design (Wall, Spandrel or Slab).

5.5.2. Syntax

[ret,Angle]=SM.SectCut.GetLocalAxesAngleDesign(Name)

5.6. GetNameList

5.6.1. Remarks

This function retrieves the names of all defined section cuts.

5.6.2. Syntax

[ret,NumberNames,MyName]=SM.SectCut.GetNameList()

5.7. GetQuad

5.7.1. Remarks

5.7.2. Syntax

[ret,X,Y,Z]=SM.SectCut.GetQuad(Name,Num)

5.8. GetResultLocation

5.8.1. Remarks

This function gets the results location for an existing section cut.

5.8.2. Syntax

[ret,IsDefault,X,Y,Z]=SM.SectCut.GetResultLocation(Name)

5.9. GetResultsSide

5.9.1. Remarks

This function gets the side of the elements from which results are obtained.

5.9.2. Syntax

[ret,Side]=SM.SectCut.GetResultsSide(Name)

5.10. SetByGroup

5.10.1. Remarks

This function adds a new section cut defined by a group to the model or reinitializes an existing section cut to be defined by a group.

5.10.2. Syntax

[ret]=SM.SectCut.SetByGroup(Name,GroupName,MyType)

5.11. SetByQuad

5.11.1. Remarks

This function adds a new section cut defined by a quadrilateral to the model or reinitializes an existing section cut to be defined by a quadrilateral.

5.11.2. Syntax

[ret]=SM.SectCut.SetByQuad(Name,GroupName,MyType,X,Y,Z)

5.12. SetLocalAxesAdvancedAnalysis

5.12.1. Remarks

This function sets the advanced local axes data for an existing section cut whose result type is Analysis.

5.12.2. Syntax

[ret]=SM.SectCut.SetLocalAxesAdvancedAnalysis(Name,Active,AxVectOpt,...

AxCSys,axdir,AxPt,AxVect,Plane2,PlVectOpt,PlCSys,PlDir,PlPt,PlVect)

5.13. SetLocalAxesAnalysis

5.13.1. Remarks

This function sets the local axes angles for an existing section cut whose result type is Analysis.

5.13.2. Syntax

[ret]=SM.SectCut.SetLocalAxesAnalysis(Name,Z,Y,X)

5.14. SetLocalAxesAngleDesign

5.14.1. Remarks

This function sets the local axes angle for section cuts whose result type is Design (Wall, Spandrel or Slab).

5.14.2. Syntax

[ret]=SM.SectCut.SetLocalAxesAngleDesign(Name,Angle)

5.15. SetResultLocation

5.15.1. Remarks

This function sets the results location for an existing section cut.

5.15.2. Syntax

[ret]=SM.SectCut.SetResultLocation(Name,IsDefault,...)

5.16. SetResultsSide

5.16.1. Remarks

This function sets the side of the elements from which results are obtained.

5.16.2. Syntax

[ret]=SM.SectCut.SetResultsSide(Name,Side)