

```

1 Imports System.Collections.Generic
2
3 Imports Grasshopper.Kernel
4 Imports Rhino.Geometry
5 Imports Interop.gsa_8_7
6 Imports UsageDataCollection
7
8
9 Public Class SetElement
10     Inherits GH_Component
11     ''' <summary>
12     ''' Set an element in GSA with new or modified characteristics
13     ''' </summary>
14     Public Sub New()
15         MyBase.New("Set Element", "Set Element", _
16             "Set an element in GSA with new or modified characteristics",
17             _
18             "GsaComTools", "Elements")
19     End Sub
20     ''' <summary>
21     ''' Registers all the input parameters for this component. SET, EL, 11, ,
22     NO_RGB, BEAM, 1, 101, 12, 22, 3
23     ''' </summary>
24     Protected Overrides Sub RegisterInputParams(pManager As
25         GH_Component.GH_InputParamManager)
26         pManager.AddGenericParameter("Gsa", "Gsa", "Gsa COM Object",
27             GH_ParamAccess.item)
28         pManager.AddIntegerParameter("Element Number", "Record", "Element
29             Number. Leave blank to add element with arbitrary element number",
30             GH_ParamAccess.item)
31         pManager.AddIntegerParameter("Element Type", "Type", "Element Type: "
32             & vbCrLf & _
33                 "1 - Bar"
34             & vbCrLf & _
35                 "2 - Beam"
36             & vbCrLf & _
37                 "3 -
38                 Spring" & vbCrLf & _
39                 "4 - Mass"
40             & vbCrLf & _
41                 "5 - Quad
42             & vbCrLf & _
43                 "6 - Quad
44             & vbCrLf & _
45                 "7 -
46                 Triangle 3" & vbCrLf & _
47                 "8 -
48                 Triangle 6" & vbCrLf & _
49                 "9 - Link"
50             & vbCrLf & _
51                 "10 -
52                 Cable" & vbCrLf & _
53                 "18 -
54                 Ground Spring" & vbCrLf & _
55                 "19 -

```

```

        Spacer" & vbCrLf & _
39                                     "20 -  ↗
        Strut" & vbCrLf & _
40                                     "21 - Tie" ↗
        & vbCrLf, _
41                                     ↗
    GH_ParamAccess.item)
42    pManager.AddIntegerParameter("Element Property", "Prop", "Element  ↗
        Property Number", GH_ParamAccess.item)
43    pManager.AddIntegerParameter("Element Group", "Group", "Element Group  ↗
        Number", GH_ParamAccess.item)
44    pManager.AddIntegerParameter("Element Ends", "Ends", "Element Ends",  ↗
        GH_ParamAccess.list)
45    pManager.AddIntegerParameter("Element Orient. Node", "OrientNode",  ↗
        "Element Orientation Node (Optional)", GH_ParamAccess.item)
46    pManager.AddNumberParameter("Element Orient. Angle", "OrientAngle",  ↗
        "Element Orientation Angle (Optional)", GH_ParamAccess.item)
47    pManager.AddBooleanParameter("Dummy", "Dummy", "Dummy (Default -  ↗
        False)", GH_ParamAccess.item)
48
49    pManager(1).Optional = True
50    pManager(2).Optional = True
51    pManager(3).Optional = True
52    pManager(4).Optional = True
53    pManager(5).Optional = True
54    pManager(6).Optional = True
55    pManager(7).Optional = True
56    pManager(8).Optional = True
57
58
59    End Sub
60
61    ''' <summary>
62    ''' Registers all the output parameters for this component.
63    ''' </summary>
64    Protected Overrides Sub RegisterOutputParams(pManager As  ↗
        GH_Component.GH_OutputParamManager)
65    pManager.AddGenericParameter("Gsa", "Gsa", "Gsa COM Object",  ↗
        GH_ParamAccess.item)
66    End Sub
67
68    ''' <summary>
69    ''' This is the method that actually does the work.
70    ''' </summary>
71    ''' <param name="DA">The DA object is used to retrieve from inputs and  ↗
        store in outputs.</param>
72    Protected Overrides Sub SolveInstance(DA As IGH_DataAccess)
73
74        'Declare Variables
75        Dim FileName As String = ""
76        Dim s As Short
77        Dim myDataCollector As New DataCollector
78        Dim myObjectWrapper As Types.GH_ObjectWrapper = Nothing
79        Dim ElNum As String
80        Dim ElNumInt As Integer
81        Dim ElType As Integer

```

```

82     Dim ElTypeString As String
83     Dim ElProp As Integer
84     Dim ElGroup As Integer
85     Dim ElEnds As New List(Of Integer)
86     Dim ElOrientationNode As Integer
87     Dim ElOrientationNodeString As String
88     Dim ElOrientationAngle As Double
89     Dim ElDummy As String
90     Dim ElDummyBoolean As Boolean
91     Dim BlankInput As Boolean = False
92     Dim myCommand As String = ""
93     Dim ElemRefs As New List(Of Integer)
94     Dim Elements() As GsaElement = Nothing
95     Dim EndString As String
96     Dim CommandString As String
97
98     'Get Input Values
99     If (Not DA.GetData(0, myObjectWrapper)) Then Return
100    If (Not DA.GetData(1, ElNumInt)) Then myCommand = "ADD" Else myCommand =
        = "SET"
101    If (Not DA.GetData(2, ElType)) Then ElType = -1
102    If (Not DA.GetData(3, ElProp)) Then ElProp = Nothing
103    If (Not DA.GetData(4, ElGroup)) Then ElGroup = Nothing
104    If (Not DA.GetDataList(5, ElEnds)) Then ElEnds = Nothing
105    If (Not DA.GetData(6, ElOrientationNode)) Then ElOrientationNode =
        Nothing
106    If (Not DA.GetData(7, ElOrientationAngle)) Then ElOrientationAngle =
        Nothing
107    If (Not DA.GetData(8, ElDummyBoolean)) Then ElDummy = "BLANK"
108
109
110    'Get GSA Object
111    Dim GsaObj As ComAuto = TryCast(myObjectWrapper.Value, ComAuto)
112
113    'Get existing values if in "SET" mode
114    If myCommand = "SET" Then
115
116        'Element Number
117        ElNum = CStr(ElNumInt)
118
119        'Get existing element data
120        ElemRefs.Add(CInt(ElNum))
121        s = GsaObj.Elements(ElemRefs.ToArray, Elements)
122
123        'Element Type
124        If ElType = -1 Then
125            If UBound(Elements) = -1 Then
126                AddRuntimeMessage(GH_RuntimeMessageLevel.Error, "Could not
                    determine element type. Ensure element exists in GSA or
                    specify explicit value.")
127                Return
128            Else
129                ElType = Elements(0).eType
130            End If
131        End If
132

```

```
133 'Element Property
134 If ElProp = Nothing Then
135     If UBound(Elements) = -1 Then
136         AddRuntimeMessage(GH_RuntimeMessageLevel.Error, "Could not
determine element property. Ensure element exists in GSA or
specify explicit value.")
137         Return
138     Else
139         ElProp = Elements(0).Property
140     End If
141 End If
142
143 'Element Group
144 If ElGroup = Nothing Then
145     If UBound(Elements) = -1 Then
146         AddRuntimeMessage(GH_RuntimeMessageLevel.Error, "Could not
determine element group. Ensure element exists in GSA or
specify explicit value.")
147         Return
148     Else
149         ElGroup = Elements(0).Group
150     End If
151 End If
152
153 'Element Ends
154 If ElEnds Is Nothing Then
155     If UBound(Elements) = -1 Then
156         AddRuntimeMessage(GH_RuntimeMessageLevel.Error, "Could not
determine element ends. Ensure element exists in GSA or
specify explicit value.")
157         Return
158     Else
159         EndString = CStr(Elements(0).Topo(0))
160         For i = 1 To UBound(Elements(0).Topo)
161             EndString = EndString & ", " & Elements(0).Topo(i)
162         Next
163     End If
164 Else
165     EndString = CStr(ElEnds(0))
166     For i = 1 To ElEnds.Count - 1
167         EndString = EndString & ", " & ElEnds(i)
168     Next
169 End If
170
171 'Element Orientation Node
172 If ElOrientationNode = Nothing Then
173     If UBound(Elements) = -1 Then
174         ElOrientationNodeString = ""
175     Else
176         ElOrientationNodeString = CStr(Elements(0).OrientNode)
177     End If
178 Else
179     ElOrientationNodeString = CStr(ElOrientationNode)
180 End If
181
182 'Element Orientation Angle
```

```

183         If ElOrientationAngle = Nothing Then
184             If UBound(Elements) = -1 Then
185                 ElOrientationAngle = 0
186             Else
187                 ElOrientationAngle = CDb1(GsaObj.Arg(10, CStr
188                     (GsaObj.GwaCommand("GET, EL, " & ElNum))))
189             End If
190         End If
191
192         'Element Dummy
193         If ElDummy = "BLANK" Then
194             If UBound(Elements) = -1 Then
195                 ElDummy = ""
196             Else
197                 ElDummy = GsaObj.Arg(GsaObj.NumArg(CStr(GsaObj.GwaCommand
198                     ("GET, EL, " & ElNum))) - 1, CStr(GsaObj.GwaCommand("GET,
199                     EL, " & ElNum)))
200             End If
201         Else
202             If ElDummyBoolean = True Then ElDummy = "DUMMY" Else ElDummy =
203                 ""
204         End If
205
206         'Prep data if in "ADD" mode
207     Else
208         'Element Number
209         ElNumInt = CInt(GsaObj.GwaCommand("HIGHEST, EL")) + 1
210         ElNum = CStr(ElNumInt)
211
212         'Element Type
213         If ElType = -1 Then
214             AddRuntimeMessage(GH_RuntimeMessageLevel.Error, "Element type
215                 not collected. Specify value.")
216             Return
217         End If
218
219         'Element Property
220         If ElProp = Nothing Then
221             AddRuntimeMessage(GH_RuntimeMessageLevel.Error, "Element
222                 property not collected. Specify value.")
223             Return
224         End If
225
226         'Element Group
227         If ElGroup = Nothing Then
228             AddRuntimeMessage(GH_RuntimeMessageLevel.Error, "Element group
229                 not collected. Specify value.")
230             Return
231         End If
232
233         'Element Ends
234         If ElEnds Is Nothing Then
235             AddRuntimeMessage(GH_RuntimeMessageLevel.Error, "Element ends
236                 not collected. Specify value(s).")
237             Return
238         Else

```

```

231         EndString = CStr(ElEnds(0))
232         For i = 1 To ElEnds.Count - 1
233             EndString = EndString & ", " & ElEnds(i)
234         Next
235     End If
236
237     'Element Orientation Node
238     If ElOrientationNode = Nothing Then
239         ElOrientationNodeString = ""
240     Else
241         ElOrientationNodeString = CStr(ElOrientationNode)
242     End If
243
244     'Element Orientation Angle
245     If ElOrientationAngle = Nothing Then
246         ElOrientationAngle = 0
247     End If
248
249     'Element Dummy
250     If ElDummyBoolean = Nothing Then
251         ElDummy = ""
252     Else
253         If ElDummyBoolean = True Then ElDummy = "DUMMY" Else ElDummy =
254             ""
255     End If
256 End If
257
258
259 'Build Command String
260 Select Case ElType
261     Case 1
262         CommandString = "SET, EL_BAR, " & ElNum & ", " & ElProp & ", " &
263             & ElGroup & ", " & EndString & ", " & ElOrientationNode &
264             ", " & ElOrientationAngle & ", " & ElDummy
265     Case 2
266         CommandString = "SET, EL_BEAM, " & ElNum & ", " & ElProp & ",
267             " & ElGroup & ", " & EndString & ", " & ElOrientationNode &
268             ", " & ElOrientationAngle & ", " & ElDummy
269     Case 3
270         CommandString = "SET, EL_SPRING, " & ElNum & ", " & ElProp &
271             ", " & ElGroup & ", " & EndString & ", " & ElOrientationNode
272             & ", " & ElOrientationAngle & ", " & ElDummy
273     Case 4
274         CommandString = "SET, EL_MASS, " & ElNum & ", " & ElProp & ",
275             " & ElGroup & ", " & EndString & ", " & ElDummy
276     Case 5
277         CommandString = "SET, EL_QUAD4, " & ElNum & ", " & ElProp & ",
278             " & ElGroup & ", " & EndString & ", " & ElOrientationAngle
279             & ", " & ElDummy
280     Case 6
281         CommandString = "SET, EL_QUAD8, " & ElNum & ", " & ElProp & ",
282             " & ElGroup & ", " & EndString & ", " & ElOrientationAngle
283             & ", " & ElDummy
284     Case 7
285         CommandString = "SET, EL_TRI3, " & ElNum & ", " & ElProp & ",

```

```

        " & ElGroup & ", " & EndString & ", " & ElOrientationAngle & "
        ", " & ElDummy

275     Case 8
276         CommandString = "SET, EL_TRI6, " & ElNum & ", " & ElProp & ", "
        " & ElGroup & ", " & EndString & ", " & ElOrientationAngle & "
        ", " & ElDummy

277     Case 9
278         CommandString = "SET, EL_LINK, " & ElNum & ", " & ElProp & ", "
        " & ElGroup & ", " & EndString & ", " & ElDummy

279     Case 10
280         CommandString = "SET, EL_CABLE, " & ElNum & ", " & ElProp & ", "
        " & ElGroup & ", " & EndString & ", " & ElDummy

281     Case 18
282         CommandString = "SET, EL_GROUND, " & ElNum & ", " & ElProp & "
        ", " & ElGroup & ", " & EndString & ", " & ElDummy

283     Case 19
284         CommandString = "SET, EL_SPACER, " & ElNum & ", " & ElProp & "
        ", " & ElGroup & ", " & EndString & ", " & ElDummy

285     Case 20
286         CommandString = "SET, EL_STRUT, " & ElNum & ", " & ElProp & ", "
        " & ElGroup & ", " & EndString & ", " & ElOrientationNode & "
        ", " & ElOrientationAngle & ", " & ElDummy

287     Case 21
288         CommandString = "SET, EL_TIE, " & ElNum & ", " & ElProp & ", "
        " & ElGroup & ", " & EndString & ", " & ElOrientationNode & "
        ", " & ElOrientationAngle & ", " & ElDummy

289 End Select
290
291 'Execute GwaCommand
292 GsaObj.GwaCommand(CommandString)
293
294 ' Assign the data to the output parameter.
295 DA.SetData(0, GsaObj)
296
297 'Send Usage Data
298 myDataCollector.UsageStatistics("SetElement")
299 End Sub
300
301 ''' <summary>
302 ''' Provides an Icon for every component that will be visible in the User
    Interface.
303 ''' Icons need to be 24x24 pixels.
304 ''' </summary>
305 Protected Overrides ReadOnly Property Icon() As System.Drawing.Bitmap
306     Get
307         'You can add image files to your project resources and access them
            like this:
308         ' return Resources.IconForThisComponent;
309         Return My.Resources.Element_Set_Add
310     End Get
311 End Property
312
313 ''' <summary>
314 ''' Gets the unique ID for this component. Do not change this ID after
    release.
315 ''' </summary>

```

```
316     Public Overrides ReadOnly Property ComponentGuid() As Guid
317         Get
318             Return New Guid("{ccf86cd9-e9e1-43c3-a3d0-b4500ed2979a}")
319         End Get
320     End Property
321 End Class
```