



```
1 Imports SldWorks
2 Imports SwCommands
3 Imports SwConst
4 Imports System.IO
5 Imports System.IO.Compression
6 Imports System.IO.Compression.ZipFile
7 Imports Microsoft.Office.Interop.Excel
8
9 Public Class SWFunctions
10
11     Public Shared swApp As SldWorks.SldWorks
12
13
14     Public Shared CusProperties_Part As CustomPropertyManager
15     Public Shared CusProperties_Assy As CustomPropertyManager
16
17     Public Shared swModelDocExt_Part As ModelDocExtension
18     Public Shared swModelDocExt_Assy As ModelDocExtension
19
20     Public Shared Add_BOM As Boolean = False
21     Public Shared swAssy_Docs As New List(Of Assy_Docs)
22     Public Shared swPart_Docs As New List(Of Part_Docs)
23     Public Shared swDwg_Docs As New List(Of Drawing_Docs)
24     Shared swComp_Assy As String
25
26
27     Class Assy_Docs
28         Public Comp As String
29         Public subcomp As String
30         Public instance_ID As String
31         Public Part_Number As String = "Null"
32         Public Nomenclature As String = "Null"
33         Public Spec As String = "Null"
34         Public Description As String = "Null"
35         Public Material As String = "Null"
36         Public Weight As String = "Null"
37         Public Name As String
38         Public Counter As Integer
39         Public Used As Boolean
40
41     Public Sub New(s1 As String, s2 As String, s3 As String, s4 As String, s5 As String, s6 As String, s7 As String, s8 As String, s9 As String)
42         Comp = s1
43         subcomp = s2
44         instance_ID = s3
45         Part_Number = s4
46         Nomenclature = s5
47         Description = s6
48         Spec = s7
49         Material = s8
50         Weight = s9
51
```

```
52         End Sub
53
54     End Class
55
56     Class Part_Docs
57         Public Comp As String
58         Public subcomp As String
59         Public instance_ID As String
60         Public Part_Number As String = "Null"
61         Public Nomenclature As String = "Null"
62         Public Spec As String = "Null"
63         Public Description As String = "Null"
64         Public Material As String = "Null"
65         Public Weight As String = "Null"
66         Public Name As String
67         Public Counter As Integer
68         Public Used As Boolean
69
70         Public Sub New(s1 As String, s2 As String, s3 As String, s4 As String, 
71             s5 As String, s6 As String, s7 As String, s8 As String, s9 As String)
72             Comp = s1
73             subcomp = s2
74             instance_ID = s3
75             Part_Number = s4
76             Nomenclature = s5
77             Description = s6
78             Spec = s7
79             Material = s8
80             Weight = s9
81
82         End Sub
83
84     End Class
85
86     Class Drawing_Docs
87         Public Comp As String
88         Public subcomp As String
89         Public instance_ID As String
90         Public Part_Number As String = "Null"
91         Public Nomenclature As String = "Null"
92         Public Spec As String = "Null"
93         Public Description As String = "Null"
94         Public Material As String = "Null"
95         Public Name As String
96         Public Counter As Integer
97         Public Used As Boolean
98
99         Public Sub New(s1 As String, s2 As String, s3 As String, s4 As String, 
100             s5 As String, s6 As String, s7 As String, s8 As String)
101             Comp = s1
102             subcomp = s2
103             instance_ID = s3
```

```

102         Part_Number = s4
103         Nomenclature = s5
104         Spec = s6
105         Description = s7
106         Material = s8
107
108     End Sub
109
110 End Class
111
112
113 Shared Sub Connect_SW()
114     Dim SWProcess() As Process
115     Dim SWProcess2() As Process
116
117     Dim app1 As Object = Nothing
118     Dim SW_Path = String.Empty
119     Dim aProcess As System.Diagnostics.Process
120     If swApp Is Nothing Then
121
122
123         SWProcess = Process.GetProcessesByName("SLDWORKS")
124
125         'MsgBox(SWProcess.Count) '& " --- " & SWProcess2.Count)
126         If SWProcess.Count > 1 Then
127             For k = 0 To SWProcess.Count - 1
128
129                 'SWProcess(k).CloseMainWindow()
130                 aProcess = System.Diagnostics.Process.GetProcessById
131                 (SWProcess(k).Id)
132                 aProcess.Kill() 'kills all .exe
133                 'aProcess.Close() 'doesn't work at all
134                 'aProcess.CloseMainWindow() 'closes windows that are opened
135                 and visible
136
137                 Threading.Thread.Sleep(500)
138             Next
139         End If
140         SWProcess = Process.GetProcessesByName("SLDWORKS")
141         If SWProcess.Count = 1 Then
142             'app1 = TryCast(CreateObject("SldWorks.Application"),
143             SolidWorks.Interop.sldworks.ISldWorks)
144             app1 = CreateObject("SldWorks.Application")
145             Threading.Thread.Sleep(1000)
146         Else
147             For i = 10 To 0 Step -1
148                 If i = 0 Then
149                     SW_Path = "C:\Program Files\SOLIDWORKS Corp\SOLIDWORKS
150                     \SLDWORKS.exe"
151                     Exit For
152                 End If
153             Next
154         End If
155     End Sub
156 End Sub

```

```
150         SW_Path = "C:\Program Files\SOLIDWORKS Corp\SOLIDWORKS (" & ↗
151             i & ")"
152         'MsgBox(SW_Path & " ---- " & Directory.Exists(SW_Path))
153         If Directory.Exists(SW_Path) Then
154             SW_Path = SW_Path & "\SLDWORKS.exe"
155             Exit For
156         End If
157     Next
158     Process.Start(SW_Path)
159     SWProcess2 = Process.GetProcessesByName("SLDWORKS")
160     Threading.Thread.Sleep(1000)
161     'MsgBox(SWProcess2.Count)
162     If SWProcess2.Count = 1 Then
163         'app1 = TryCast(CreateObject("SldWorks.Application"), ↗
164             SolidWorks.Interop.sldworks.ISldWorks)
165         app1 = CreateObject("SldWorks.Application")
166         'MsgBox("1")
167     End If
168 End If
169
170
171     app1.Visible = True
172     app1.FrameState = 1
173     swApp = app1
174 End If
175
176
177 End Sub
178
179 Shared Sub Remove_SW()
180     If swApp IsNot Nothing Then
181
182         Threading.Thread.Sleep(250)
183         System.Runtime.InteropServices.Marshal.ReleaseComObject(swApp)
184         swApp = Nothing
185
186     End If
187
188 End Sub
189
190
191 Shared Function Opened_Docs()
192     Dim swApp As SldWorks.SldWorks
193     Dim Open_Docs As Object
194     Dim count As Integer
195
196     swApp = CreateObject("SldWorks.Application")
197
198     count = swApp.GetDocumentCount
199     Open_Docs = swApp.GetDocuments
```

```
200
201         Return Open_Docs
202     End Function
203
204
205     Shared Function Sheet_Rename()
206
207         Dim swApp As SldWorks.SldWorks
208         Dim swDoc As ModelDoc2
209         Dim swDraw As DrawingDoc
210         Dim SWSheet As Sheet
211         Dim swModelDocExt As ModelDocExtension
212
213         swApp = CreateObject("SldWorks.Application")
214         swDoc = swApp.ActiveDoc()
215
216         If swDoc Is Nothing Then
217             Functions.Error_Form()
218         Else
219
220             If swDoc.GetType <> 3 Then
221                 Functions.Error_Form(, "this is not a drawing file")
222             Else
223                 swDraw = swDoc
224                 If swDraw Is Nothing Then
225                     Functions.Error_Form(, "No Drawing sheet loaded")
226                 Else
227                     Dim sheetnames() As String
228                     Dim Sheetnumbers As Integer
229                     Start_Window.SW_Doc.Text = "Sheet Renaming in Proocess"
230
231                     swApp.Visible = True
232                     Sheetnumbers = swDraw.GetSheetCount
233                     sheetnames = swDraw.GetSheetNames
234
235                     For i = 1 To (Sheetnumbers)
236                         swDraw.ActivateSheet(sheetnames(i - 1))
237                         SWSheet = swDraw.GetCurrentSheet
238                         SWSheet.SetName("Renaming" & i)
239                     Next
240
241                     sheetnames = swDraw.GetSheetNames
242                     For i = 1 To (Sheetnumbers)
243                         swDraw.ActivateSheet(sheetnames(i - 1))
244                         SWSheet = swDraw.GetCurrentSheet
245                         SWSheet.SetName("Sheet" & i)
246                         swModelDocExt = swDoc.Extension
247                         swModelDocExt.ViewZoomToSheet()
248                     Next
249
250                 End If
251             End If
```

```
252         Return True
253     End If
254
255     Return False
256 End Function
257
258
259 Shared Function PDF()
260
261     Dim swApp As SldWorks.SldWorks
262     Dim swDoc As ModelDoc2
263
264     Dim CusProperties As CustomPropertyManager
265
266     Dim bool As Boolean
267
268     swApp = CreateObject("SldWorks.Application")
269     swDoc = swApp.ActiveDoc
270
271     If swDoc Is Nothing Then
272         Functions.Error_Form()
273     Else
274         If swDoc.GetType = 3 Then
275
276             Dim Title = String.Empty
277             Dim REV = String.Empty
278             Dim REV_1 = String.Empty
279             Dim Valout = String.Empty
280             Dim Res_Valout = String.Empty
281             Dim Resolved As Boolean
282             Dim Cus_Prop_val As Integer
283             Dim FileName = String.Empty
284             Dim PDF_Path = String.Empty
285             Dim PathName = String.Empty
286             Dim Structural = String.Empty
287             Dim Drawings_Major = String.Empty
288             Dim Drawings_Minor = String.Empty
289             Dim Plane_Folder = String.Empty
290             Dim Released_to_Inspection = "(4) Released to Inspection"
291             Dim Released_As = String.Empty
292
293             Dim swLayer As Layer
294             Dim swLayerMgr = swDoc.GetLayerManager
295
296             PathName = swDoc.GetPathName()
297             Debug.Print(PathName)
298
299             FileName = System.IO.Path.GetFileNameWithoutExtension(PathName)
300             Debug.Print(FileName)
301
302             PathName = System.IO.Path.GetDirectoryName(PathName)
303             Debug.Print(PathName)
```

```
304
305         swDoc.ClearSelection2(True)
306
307         Structural = Directory.GetParent(PathName).FullName
308         Debug.Print(Structural)
309
310         Drawings_Major = Directory.GetParent(Structural).FullName
311         Debug.Print(Drawings_Major)
312
313         Dim length = Drawings_Major.Length - 18
314         Debug.Print(length)
315
316         Dim Major = Drawings_Major.Substring(length, 3)
317         Debug.Print(Major)
318
319         Plane_Folder = Directory.GetParent(Drawings_Major).FullName
320         Debug.Print(Plane_Folder)
321
322         'If Major = "(2)" Then
323         '    'Drawing is a major
324         '    Released_As = "Major"
325         '    PDF_Path = Path.Combine(Plane_Folder,
326         '        Released_to_Inspection)
327         '    PDF_Path = Path.Combine(PDF_Path, Released_As)
328         '    System.IO.Directory.CreateDirectory(PDF_Path)
329
330         'ElseIf Major = "(3)" Then
331
332         '    'Drawing is a minor
333         '    Released_As = "Minor"
334
335         '    PDF_Path = Path.Combine(Drawings_Major,
336         '        Released_to_Inspection)
337         '    PDF_Path = Path.Combine(PDF_Path, Released_As)
338         '    System.IO.Directory.CreateDirectory(PDF_Path)
339
340         'Else
341         Released_As = "PDF"
342         PDF_Path = Path.Combine(PathName, "PDF RELEASES")
343
344         'End If
345
346         CusProperties = swDoc.Extension.CustomPropertyManager("")
347
348         Title = CusProperties.Get("TITLE")
349         REV_1 = CusProperties.Get("REV 1")
350
351         If REV_1.StartsWith("P") Then
352             REV = CusProperties.Get("REV 1")
353         Else
```

```
354         REV = CusProperties.Get("CURRENT REV")
355     End If
356
357     Try
358         Cus_Prop_val = CusProperties.Get6("CURRENT REV", False, Valout, REV, Resolved, False)
359
360         Select Case REV
361             Case "P1"
362                 PDF_Path = Path.Combine(PDF_Path, "Preliminaries")
363
364             Case "P2"
365                 PDF_Path = Path.Combine(PDF_Path, "Preliminaries")
366             Case "P3"
367                 PDF_Path = Path.Combine(PDF_Path, "Preliminaries")
368             Case "P4"
369                 PDF_Path = Path.Combine(PDF_Path, "Preliminaries")
370             Case "P5"
371                 PDF_Path = Path.Combine(PDF_Path, "Preliminaries")
372             Case "P6"
373                 PDF_Path = Path.Combine(PDF_Path, "Preliminaries")
374             Case "P7"
375                 PDF_Path = Path.Combine(PDF_Path, "Preliminaries")
376             Case "I/R"
377                 PDF_Path = Path.Combine(PDF_Path, "IR")
378                 REV = "IR"
379                 swLayer = swLayerMgr.GetLayer("SIGBLK")
380                 swLayer.Visible = False
381             Case "A"
382                 PDF_Path = Path.Combine(PDF_Path, "A")
383                 swLayer = swLayerMgr.GetLayer("SIGBLK")
384                 swLayer.Visible = False
385             Case "B"
386                 PDF_Path = Path.Combine(PDF_Path, "B")
387                 swLayer = swLayerMgr.GetLayer("SIGBLK")
388                 swLayer.Visible = False
389             Case "C"
390                 PDF_Path = Path.Combine(PDF_Path, "C")
391                 swLayer = swLayerMgr.GetLayer("SIGBLK")
392                 swLayer.Visible = False
393             Case "D"
394                 PDF_Path = Path.Combine(PDF_Path, "D")
395                 swLayer = swLayerMgr.GetLayer("SIGBLK")
396                 swLayer.Visible = False
397             Case Else
398                 REV = ""
399         End Select
400
401         Catch 'nl As System.NullReferenceException
402             Functions.Error_Form("MISSING LAYER", "SIGBLK")
403
404     End Try
```



```
405
406         System.IO.Directory.CreateDirectory(PDF_Path)
407
408         'If REV = "I/R" Then
409         '     REV = "IR"
410         '     swLayer = swLayerMgr.GetLayer("SIGBLK")
411         '     swLayer.Visible = False
412         'End If
413
414         PDF_Path = Path.Combine(PDF_Path, FileName)
415         Title = PDF_Path + " " + REV + " (" + Title + ")"
416
417         bool = swDoc.SaveAs4(Title + ".PDF",                                ↗
                               swSaveAsVersion_e.swSaveAsCurrentVersion,    ↗
                               swSaveAsOptions_e.swSaveAsOptions_Silent, 0, 0)
418         If bool = False Then
419             Functions.Error_Form("PDF Error", "PDF did not save",,,, ↗
                                   False,)
420         End If
421
422         Else
423             Functions.Error_Form("Document not a SLDDRW", "No Drawing File ↗
                                   opened")
424
425             End If
426             Return True
427         End If
428
429         Return False
430     End Function
431
432
433     Shared Function SW_FileName(path As String)
434         Dim FileName As String = path
435
436         FileName = FileName.Remove(0, FileName.LastIndexOf("\") + 1)
437         FileName = FileName.Remove(FileName.LastIndexOf("."))
438
439         Return FileName
440     End Function
441
442
443     Shared Function SW_Extension(path As String)
444         Dim Extension As String = path
445
446         Extension = Extension.Remove(0, Extension.LastIndexOf("."))
447
448         Return Extension
449     End Function
450
451
452     Shared Function PackandGo()
```

```
453
454     Dim swApp As SldWorks.SldWorks
455     Dim swDoc As ModelDoc2
456     Dim swPackAndGo As PackAndGo
457
458     Dim status As Boolean
459     Dim pgFileNames As Object = Nothing
460     Dim pgFileStatus As Object = Nothing
461     Dim statuses As Object
462     Dim Open_Docs As Object
463     Dim Filename = String.Empty
464     Dim Pathname = String.Empty
465     Dim ASSYname = String.Empty
466     Dim MyFolder = String.Empty
467     Dim ExistingFile = String.Empty
468
469     Dim TimeNow As DateTime = DateTime.Now
470     Dim format As String = "MM-dd-yyyy HH-mm-ss"
471
472     swApp = CreateObject("SldWorks.Application")
473     swApp.Visible = True
474     swDoc = swApp.ActiveDoc
475
476     'If swDoc Is Nothing Then
477     If swDoc.GetType <> 3 Then
478
479         Functions.Error_Form()
480     Else
481
482         swDoc.ForceRebuild3(True)
483         swDoc.Save3(1, 1, 2)
484
485         'check for existing file is true
486         ExistingFile = swDoc.GetPathName
487
488         If ExistingFile = "" Then
489
490             Functions.Error_Form("File Error", "File is not Saved",, "Please ↗
491                 save before using Pack n Go",,,)
492         Else
493
494             Filename = System.IO.Path.GetFileNameWithoutExtension ↗
495                 (ExistingFile)
496             Pathname = System.IO.Path.GetDirectoryName(ExistingFile)
497             ASSYname = Pathname + "\" + Filename + " (Pack and Go)"
498
499             swPackAndGo = swDoc.Extension.GetPackAndGo
500             swPackAndGo.IncludeDrawings = True
501             swPackAndGo.IncludeSimulationResults = False
502             'swPackAndGo.IncludeSuppressed = True
```

```
503      swPackAndGo.IncludeToolboxComponents = True
504      swPackAndGo.FlattenToSingleFolder = True
505
506
507      status = swPackAndGo.GetDocumentSaveToNames(pgFileNames,
508      pgFileStatus)
509
510      Dim External_Files As New List(Of String)
511      Dim exclude = {".slddrw", ".sldasm", ".sldprt"}
512      Dim Extension As String
513
514      For Each myFile As String In Directory.GetFiles(Pathname)
515          Extension = LCase(Path.GetExtension(myFile))
516          If Extension = exclude(0) Or Extension = exclude(1) Or
517          Extension = exclude(2) Then
518              MsgBox("failed")
519          Else
520              External_Files.Add(myFile)
521              MsgBox(myFile)
522          End If
523      Next
524
525      'Dim external_Files = Directory.GetFiles(Pathname)
526
527      status = swPackAndGo.AddExternalDocuments
528      (External_Files.ToArray)
529
530
531      'status = swPackAndGo.AddExternalDocuments
532      (swPackAndGo.GetDocumentNames(pgFileNames)) 'get files in
533      directory as an array
534
535
536      'Puts Pack and Go files in native Directory
537      status = swPackAndGo.SetSaveToName(True, Pathname)
538
539      statuses = swDoc.Extension.SavePackAndGo(swPackAndGo)
540
541      'Puts Pack and Go files in native Directory
542      status = swPackAndGo.SetSaveToName(True, ASSYname)
543
544      swPackAndGo.AddPrefix = Filename + " - "
545
546      'swPackAndGo.
547
548      'Puts Pack and Go files in new Directory named "Assembly name
549      (Pack and Go)"
550      'status = swPackAndGo.SetSaveToName(True, ASSYname)
551
552      statuses = swDoc.Extension.SavePackAndGo(swPackAndGo)
553
554      Open_Docs = Opened_Docs()
```

```
549
550         If swDoc.GetType = CInt(3) Then
551
552             For i = LBound(Open_Docs) To UBound(Open_Docs)
553                 swDoc = Open_Docs(i)
554                 Dim F_Name As String
555                 Dim errors As Integer
556                 Dim MoreErr As Integer
557
558                 If swDoc.GetType = 2 Or swDoc.GetType = 3 Then
559                     F_Name = SWFunctions.SW_FileName(swDoc.GetPathName
560 ( ))
561                     swApp.IActivateDoc3(F_Name, False, errors)
562                     swDoc.Save3(1, errors, MoreErr)
563                     swApp.QuitDoc(F_Name)
564
565                 End If
566
567             Next
568
569             swApp.QuitDoc(ExistingFile)
570             swApp.OpenDoc6(ExistingFile, 3, 1, "", 1, 2)
571
572         ElseIf swDoc.GetType = CInt(2) Then
573
574             For i = LBound(Open_Docs) To UBound(Open_Docs)
575                 swDoc = Open_Docs(i)
576                 Dim F_Name As String
577                 Dim errors As Integer
578                 Dim MoreErr As Integer
579
580                 If swDoc.GetType = 2 Or swDoc.GetType = 3 Then
581                     F_Name = SWFunctions.SW_FileName(swDoc.GetPathName
582 ( ))
583                     swApp.IActivateDoc3(F_Name, False, errors)
584                     swDoc.Save3(1, errors, MoreErr)
585
586                     swApp.QuitDoc(F_Name)
587
588                     i = UBound(Open_Docs)
589
590                 End If
591
592             Next
593
594             MsgBox(ExistingFile)
595             swApp.CloseAllDocuments(True)
596             swApp.OpenDoc6(ExistingFile, 2, 1, "", 1, 2)
597
598         Else
599             Functions.Error_Form("Pack And Go Failure", "Incompatible
600 ( )")
```

```

        file type to use Pack and Go feature",,,,,)
599     End If
600
601     If Directory.Exists(ASSYname) Then
602
603         status = swPackAndGo.SetSaveToName(True, ASSYname + " - " + ↗
            TimeNow.ToString(Format))
        statuses = swDoc.Extension.SavePackAndGo(swPackAndGo)
604
605     Else
606         status = swPackAndGo.SetSaveToName(True, ASSYname)
607         statuses = swDoc.Extension.SavePackAndGo(swPackAndGo)
608     End If
609
610
611 End If
612 If File.Exists(ASSYname + ".zip") Then
613
614     System.IO.Compression.ZipFile.CreateFromDirectory(ASSYname, ↗
        ASSYname + " - " + TimeNow.ToString(Format) + ".zip")
615
616 Else
617     System.IO.Compression.ZipFile.CreateFromDirectory(ASSYname, ↗
        ASSYname + ".zip")
618
619 End If
620
621 MsgBox("Complete",, "Packed and Zipped")
622
623 'ZipFile.CreateFromDirectory(ASSYname, ASSYname + " IR " + ".zip")
624
625 Return True
626 End If
627
628 Return False
629 End Function
630
631 Shared Function PackandGo2()
632
633 Dim swApp As SldWorks.SldWorks
634 Dim swDoc As ModelDoc2
635 Dim swPackAndGo As PackAndGo
636 Dim swPackAndGo_Zip As PackAndGo
637
638 Dim status As Boolean
639 Dim pgFileNames As Object = Nothing
640 Dim pgFileStatus As Object = Nothing
641 Dim statuses As Object
642 Dim Open_Docs As Object
643 Dim Filename = String.Empty
644 Dim Pathname = String.Empty
645 Dim ASSYname = String.Empty
646 Dim Pack_N_Go_name = String.Empty
647 Dim MyFolder = String.Empty
648 Dim ExistingFile = String.Empty

```

```
647
648     Dim swDocType As Integer
649     Dim namesCount As Integer
650
651
652     Dim TimeNow As DateTime = DateTime.Now
653     Dim format As String = "MM-dd-yyyy HH-mm-ss"
654
655     swApp = CreateObject("SldWorks.Application")
656     swApp.Visible = True
657     swDoc = swApp.ActiveDoc
658
659     'If swDoc Is Nothing Then
660     If swDoc.GetType <> 3 Then
661
662         Functions.Error_Form()
663     Else
664
665         swDoc.ForceRebuild3(True)
666         swDoc.Save3(1, 1, 2)
667
668         'check for existing file is true
669         ExistingFile = swDoc.GetPathName
670         If ExistingFile = "" Then
671             Functions.Error_Form("File Error", "File is not Saved",, "Please ↗
        save before using Pack n Go",,,)
672         Else
673
674
675             Filename = System.IO.Path.GetFileNameWithoutExtension ↗
        (ExistingFile)
676             Pathname = System.IO.Path.GetDirectoryName(ExistingFile)
677             ASSYname = Pathname + "\" + Filename + " (Pack and Go)"
678             Pack_N_Go_name = Filename + " (Pack and Go)"
679
680             swPackAndGo = swDoc.Extension.GetPackAndGo
681             swPackAndGo.IncludeDrawings = True
682             swPackAndGo.IncludeSimulationResults = False
683             swPackAndGo.IncludeSuppressed = True
684             swPackAndGo.IncludeToolboxComponents = True
685             swPackAndGo.FlattenToSingleFolder = True
686
687             swPackAndGo_Zip = swDoc.Extension.GetPackAndGo
688             swPackAndGo_Zip.IncludeDrawings = True
689             swPackAndGo_Zip.IncludeSimulationResults = False
690             swPackAndGo_Zip.IncludeSuppressed = True
691             swPackAndGo_Zip.IncludeToolboxComponents = True
692             swPackAndGo_Zip.FlattenToSingleFolder = True
693
694             namesCount = swPackAndGo.GetDocumentNamesCount
695
696
```

```
697      status = swPackAndGo.GetDocumentSaveToNames(pgFileNames,
698      pgFileStatus)
699
700
701
702      Dim External_Files As New List(Of String)
703      Dim exclude = {".slddrw", ".sldasm", ".sldprt"}
704      Dim Extension As String
705
706      For Each myFile As String In Directory.GetFiles(Pathname)
707          Extension = LCase(Path.GetExtension(myFile))
708          If Extension = exclude(0) Or Extension = exclude(1) Or
709              Extension = exclude(2) Then
710              'MsgBox("failed")
711          Else
712              External_Files.Add(myFile)
713              'MsgBox(myFile)
714          End If
715      Next
716      status = swPackAndGo.AddExternalDocuments
717          (External_Files.ToArray)
718
719      'status = swPackAndGo_zip.AddExternalDocuments
720          (External_Files.ToArray)
721
722      'Puts Pack and Go files in native Directory
723      status = swPackAndGo.SetSaveToName(True, Pathname)
724      statuses = swDoc.Extension.SavePackAndGo(swPackAndGo)
725
726      'Puts Pack and Go files in native Directory
727
728      'swPackAndGo_Zip.AddPrefix = Filename + " - "
729      'status = swPackAndGo_Zip.SetSaveToName(True, ASSYname)
730      'statuses = swDoc.Extension.SavePackAndGo(swPackAndGo_Zip)
731
732
733
734      Open_Docs = Opened_Docs()
735
736      If swDoc.GetType = CInt(3) Then
737
738          swDocType = swDoc.GetType
739
740          For i = LBound(Open_Docs) To UBound(Open_Docs)
741              swDoc = Open_Docs(i)
742              Dim F_Name As String
743              Dim errors As Integer
744              Dim MoreErr As Integer
```

```

745
746         If swDoc.GetType = 2 Or swDoc.GetType = 3 Then
747             F_Name = SWFunctions.SW_FileName(swDoc.GetPathName
748             ())
749             swApp.IActivateDoc3(F_Name, False, errors)
750             swDoc.Save3(1, errors, MoreErr)
751             swApp.QuitDoc(F_Name)
752         End If
753
754     Next
755
756
757
758     Dim pgFileNames2 As Object
759     Dim pgFileStatus2 As Object
760
761     'disconnect error when creating folder of assyname june
762     '15th, 2021.
763     'Resolved, can't close document before finishing pack and go
764
765     status = swPackAndGo_Zip.GetDocumentSaveToNames
766     (pgFileNames2, pgFileStatus2)
767
768     status = swPackAndGo_Zip.AddExternalDocuments
769     (External_Files.ToArray)
770
771     status = swPackAndGo_Zip.SetSaveToName(True, ASSYname +
772     ".zip")
773     swPackAndGo_Zip.AddPrefix = Filename + " - "
774
775     'ReDim pgFileNames2(namesCount - 1)
776     'ReDim pgFileStatus2(namesCount - 1)
777     'status = swPackAndGo_Zip.GetDocumentSaveToNames
778     (pgFileNames2, pgFileStatus2)
779     'Debug.Print("")
780     'Debug.Print(" My Pack and Go path and filenames after
781     adding prefix and suffix: ")
782     'For i = 0 To (namesCount - 1)
783     '    Debug.Print(" My path and filename is: " &
784     pgFileNames2(i))
785     'Next i
786
787     Dim Zipped_File As String
788     Zipped_File = ASSYname + ".zip"
789
790     If File.Exists(Zipped_File) Then
791
792         status = swPackAndGo_Zip.SetSaveToName(True, ASSYname +
793         " - " + TimeNow.ToString(format) + ".zip")

```



```
788         statuses = swDoc.Extension.SavePackAndGo
              (swPackAndGo_Zip)
789     Else
790
791         statuses = swDoc.Extension.SavePackAndGo
              (swPackAndGo_Zip)
792     End If
793     swApp.QuitDoc(ExistingFile)
794     swApp.OpenDoc6(ExistingFile, swDocType, 1, "", 1, 2)
795
796 End If
797
798 'disconnect error when creating folder of assyname june 15th,
      2021.
799 'If File.Exists(ASSYname + ".zip") Then
800
801     '    System.IO.Compression.ZipFile.CreateFromDirectory(ASSYname,
      ASSYname + " - " + TimeNow.ToString(format) + ".zip")
802 'Else
803     '    System.IO.Compression.ZipFile.CreateFromDirectory(ASSYname,
      ASSYname + ".zip")
804 'End If
805 'MsgBox("Complete",, "Packed and Zipped")
806
807 'IDK where this came from
808 'ZipFile.CreateFromDirectory(ASSYname, ASSYname + " IR " +
      ".zip")
809
810     Return True
811 End If
812 End If
813
814 Return False
815 End Function
816
817 Shared Function Save_Step()
818
819     Dim swApp As SldWorks.SldWorks
820     Dim swDoc As ModelDoc2
821     Dim status As Boolean = False
822
823     swApp = CreateObject("SldWorks.Application")
824     swDoc = swApp.ActiveDoc
825
826     If swDoc Is Nothing Then
827
828         Functions.Error_Form()
829
830     Else
831
832         If swDoc.GetType = 1 Then
833             'get directory path, sometimes it doesn't save in the correct
```

```

    folder
834     Dim PathName = String.Empty
835     PathName = swDoc.GetPathName()
836     PathName = SW_FileName(PathName)
837     swDoc.ClearSelection2(True)
838     status = swDoc.SaveAs(PathName + ".STEP")
839
840     Else
841     Functions.Error_Form("Not a Part File", "This is not a Part File")
842
843     End If
844
845     Return status
846 End Function
847
848
849 Shared Function Save_As()
850
851     Dim swApp As SldWorks.SldWorks
852     Dim swDoc As ModelDoc2
853     Dim status As Boolean = False
854     Dim bool As Boolean
855
856     swApp = CreateObject("SldWorks.Application")
857     swDoc = swApp.ActiveDoc
858
859     If swDoc Is Nothing Then
860
861     Functions.Error_Form()
862
863     Else
864
865     Dim OG_Path As String = swDoc.GetPathName()
866     bool = swDoc.Extension.RunCommand
867     (swCommands_e.swCommands_SaveAs, "")
868     Dim New_Path As String = swDoc.GetPathName()
869
870     If OG_Path = New_Path Then
871     status = False
872     Else
873     status = True
874     End If
875
876     End If
877
878     Return status
879 End Function
880
881 Shared Function Save_Doc()
882     Dim swApp As SldWorks.SldWorks
```

```
883     Dim bool As Boolean
884     Dim swDoc As ModelDoc2
885
886     swApp = CreateObject("SldWorks.Application")
887     swDoc = swApp.ActiveDoc
888
889     bool = swDoc.Extension.RunCommand(swCommands_e.swCommands_SaveAs, 7
        """)
890     Return bool
891 End Function
892
893
894 Shared Function View_Scale(Outline1() As Double, Optional Outline2() As 7
    Double = Nothing, Optional Outline3() As Double = Nothing)
895
896     Dim View1 As Double = 10
897     Dim View2 As Double = 10
898     Dim View3 As Double = 10
899     Dim Small_Scale As Double = 10 'Set as high value
900
901     View1 = Boundary_box(Outline1)
902
903     If Outline2 IsNot Nothing Then
904         View2 = Boundary_box(Outline2)
905     End If
906
907     If Outline3 IsNot Nothing Then
908         View3 = Boundary_box(Outline3)
909     End If
910
911     If View1 <> 1 Or View2 <> 1 Or View3 <> 1 Then
912         Dim Scales As Double() = {View1, View2, View3}
913
914         For Each element As Double In Scales
915             Small_Scale = Math.Min(Small_Scale, element)
916         Next
917         'MsgBox(View1 & ", " & View2 & ", " & View3 & ", " & 7
            Small_Scale)
918
919     End If
920
921     Small_Scale = Math.Round(1 / Small_Scale, 1)
922     'MsgBox(Small_Scale & " This is the scale value")
923
924     'If Small_Scale <= 0.5 Then
925
926     '    Small_Scale = Math.Round(1 / Small_Scale, 0)
927     '    MsgBox(Small_Scale)
928
929     'ElseIf Small_Scale > 0.25 Then
930     '    MsgBox(Small_Scale)
931     '    Small_Scale = (1 / Small_Scale) * 2
```

```

932         ' MsgBox(Small_Scale)
933         ' Small_Scale = Math.Round(Small_Scale, 1, MidpointRounding.AwayFromZero)
934         ' MsgBox(Small_Scale)
935         ' Small_Scale = Math.Floor(Small_Scale)
936         ' MsgBox(Small_Scale)
937         ' Small_Scale = Small_Scale / 2
938         ' MsgBox(Small_Scale)
939
940     'End If
941
942     Return Small_Scale
943 End Function
944
945
946 Shared Function Boundary_box(Outline() As Double)
947
948     Dim View_Scale_Factor As Decimal() = {0.0, 0.0}
949     Dim Boundary As Double() = {0.0, 0.0, 0.0}
950     Dim Scales_sw As Double() = {1 / 2, 1 / 3, 1 / 4, 1 / 5, 1 / 6, 1 / 7, 1 / 8, 1 / 10, 1 / 12, 1, 2 / 3, 2, 3}
951     Dim Scale_Factor As Double = 0.0
952
953     Boundary = Outline
954
955     Boundary(0) = Boundary(0) * 39.3701 ' x min
956     Boundary(1) = Boundary(1) * 39.3701 ' y min
957     Boundary(2) = Boundary(2) * 39.3701 ' x max
958     Boundary(3) = Boundary(3) * 39.3701 ' y max
959
960     Boundary(2) = Boundary(2) - Boundary(0)
961     Boundary(3) = Boundary(3) - Boundary(1)
962
963     View_Scale_Factor(0) = 4.875 / Boundary(3) 'Scale factor to achieve 4.875" on Y height
964     View_Scale_Factor(1) = 10.5 / Boundary(2) 'Scale factor to achieve 10.5" on X width
965
966     'MsgBox(View_Scale_Factor(0) & " Y - " & View_Scale_Factor(1))
967
968     Dim smallestdiff As Double = Math.Abs(View_Scale_Factor(0) - Scales_sw(0))
969     Dim smallestdiffIndex = 0
970     Dim Currdiff As Double
971     Dim j = 0
972     For j = 0 To Scales_sw.Count - 1
973         Currdiff = Math.Abs(View_Scale_Factor(0) - Scales_sw(j))
974         If Currdiff < smallestdiff Then
975             smallestdiff = Currdiff
976             smallestdiffIndex = j
977         End If
978

```

```
979         Next
980
981         View_Scale_Factor(0) = Scales_sw(smallestdiffIndex)
982
983         For j = 0 To Scales_sw.Count - 1
984             Currdiff = Math.Abs(View_Scale_Factor(1) - Scales_sw(j))
985             If Currdiff < smallestdiff Then
986                 smallestdiff = Currdiff
987                 smallestdiffIndex = j
988             End If
989
990         Next
991
992         View_Scale_Factor(1) = Scales_sw(smallestdiffIndex)
993
994         'Compare each View_Scale_Factor to common scale options, pick the one closest
995
996         'If Boundary(3) > 10 Then
997         '    'scale to make Boundary_Box_Size(3)=8.25
998         '    View_Scale_Factor(0) = (10 / Boundary(3))
999         '    MsgBox(View_Scale_Factor(0))
1000         '    View_Scale_Factor(0) = Math.Round(View_Scale_Factor(0), 1)
1001         '    MsgBox(View_Scale_Factor)
1002         'Else
1003         '    View_Scale_Factor(0) = 1
1004         'End If
1005
1006         Scale_Factor = Math.Min(View_Scale_Factor(0), View_Scale_Factor(1))
1007
1008         Return Scale_Factor
1009     End Function
1010
1011
1012     Shared Function Get_View_Info(sw_View As View) 'Send a View in the first view format
1013
1014         Dim Array_Views As New List(Of Object)()
1015         Dim i = 0
1016         Dim Views As View
1017         Dim Test As View
1018
1019         Views = sw_View
1020         'sw_View is the sheet view
1021         While Views IsNot Nothing
1022
1023             Views = Views.GetNextView
1024
1025             MsgBox(Views.Name)
1026             Array_Views.Add(Views)
1027
1028             Test = Array_Views(i)
```

```

1029         MsgBox(Test.Name)
1030         'sw_View = sw_View.GetNextView
1031         i += 1
1032
1033     End While
1034
1035
1036     Return Array_Views
1037
1038 End Function
1039
1040
1041 'Add_NoteInfo is obsolete use Add_NoteInfo2
1042 Shared Function Add_NoteInfo(swDoc As ModelDoc2, Assembly_Docs As Integer,
1043     Part_Docs As Integer, Drawing_View As String,
1044     File_name As String, Instance_Num As List(Of
1045         String))
1046
1047     Dim sw_View As View
1048     Dim swNote As Note
1049     Dim swAnno As Annotation
1050
1051     Dim Text_Add As String
1052     Dim Const_Text As String
1053
1054     Dim Files As Integer = Assembly_Docs + Part_Docs
1055     Dim x_pos As Double = 0.0
1056     Dim y_pos As Double = 0.0
1057     Dim z_pos As Double = 0.0
1058
1059     Dim ChartoTrim As Char() = {"@"}
1060
1061     sw_View = swDoc.GetFirstView
1062     sw_View = sw_View.GetNextView
1063
1064     Const_Text = "$PRPSMODEL:" & Chr(34) & "NOTE INFO" & Chr(34) & "
1065         $COMP:" & Chr(34) & File_name & "@" & Drawing_View & "/"
1066
1067     For i = 0 To Instance_Num.Count - 1
1068         If i = 0 Then
1069             Text_Add = "$PRPSMODEL:" & Chr(34) & "NOTE INFO" & Chr(34) & "
1070                 " $COMP:" & Chr(34) & File_name & "-1" & "@" & Drawing_View
1071                 & Chr(34)
1072             swNote = swDoc.InsertNote(Text_Add)
1073
1074         Else
1075             Text_Add = Const_Text + Instance_Num(i - 1) & Chr(34)
1076             swNote = swDoc.InsertNote(Text_Add)
1077
1078         End If
1079     End For

```

```

1076
1077         swAnno = swNote.GetAnnotation()
1078         swAnno.SetAttachedEntities(sw_View)
1079         swAnno.SetPosition2(x_pos, y_pos, z_pos)
1080         y_pos -= 0.00635
1081
1082     Next
1083
1084     swDoc.WindowRedraw()
1085
1086     Return True
1087
1088 End Function
1089
1090
1091 Shared Function Add_NoteInfo2(swDoc As ModelDoc2, Drawing_View As String,
1092                               File_name As String, Instance_Num As List(Of
1093                               New_Drawing.SW_Opened_Files), Optional ByVal Add_Bom As
1094                               Boolean = False)
1095
1096     Dim sw_View As View
1097     Dim swNote_Info As Note
1098     Dim swAnno_Info As Annotation
1099     Dim swNote_DESCRIPTION As Note = Nothing
1100     Dim swAnno_DESCRIPTION As Annotation
1101     Dim swNote_NOMENCLATURE As Note = Nothing
1102     Dim swAnno_NOMENCLATURE As Annotation
1103
1104     Dim swNote_SPEC As Note = Nothing
1105     Dim swAnno_SPEC As Annotation
1106
1107     Dim swLayer As Layer
1108     Dim swLayerMgr As LayerMgr
1109
1110     Dim Text_Add As String
1111     Dim NOTE_INFO As String
1112     Dim DESCRIPTION As String
1113     Dim NOMENCLATURE As String
1114     Dim SPEC As String
1115     Dim Bool As Integer
1116
1117     Dim x_pos As Double = 0.0
1118     Dim x_pos_Nom As Double = 0.1
1119     Dim x_pos_Des As Double = 0.2
1120     Dim x_pos_Spec As Double = 0.3
1121     Dim y_pos As Double = 0
1122     Dim z_pos As Double = 0.0
1123
1124     Dim ChartoTrim As Char() = {"@"}
1125
1126     'swLayer =
1127     swLayerMgr = swDoc.GetLayerManager

```

```

1126     Bool = swLayerMgr.SetCurrentLayer("NOTES")
1127     'swLayer = swLayerMgr.GetLayer("NOTES")
1128
1129
1130     sw_View = swDoc.GetFirstView
1131     sw_View = sw_View.GetNextView
1132
1133     NOTE_INFO = "$PRPSMODEL:" & Chr(34) & "NOTE INFO" & Chr(34) & " $COMP:" & Chr(34) & File_name & "@" & Drawing_View & "/"
1134
1135     DESCRIPTION = "$PRPSMODEL:" & Chr(34) & "DESCRIPTION" & Chr(34) & " $COMP:" & Chr(34) & File_name & "@" & Drawing_View & "/"
1136     NOMENCLATURE = "$PRPSMODEL:" & Chr(34) & "NOMENCLATURE" & Chr(34) & " $COMP:" & Chr(34) & File_name & "@" & Drawing_View & "/"
1137     SPEC = "$PRPSMODEL:" & Chr(34) & "SPEC" & Chr(34) & " $COMP:" & Chr(34) & File_name & "@" & Drawing_View & "/"
1138
1139     For i = 0 To Instance_Num.Count - 1
1140
1141         If i = 0 Then
1142
1143             Text_Add = "$PRPSMODEL:" & Chr(34) & "NOTE INFO" & Chr(34) & " $COMP:" & Chr(34) & File_name & "-1" & "@" & Drawing_View & Chr(34)
1144             swNote_Info = swDoc.InsertNote(Text_Add)
1145
1146             If New_Drawing.Add_BOM_Hardware.Checked = True Then
1147                 Text_Add = "$PRPSMODEL:" & Chr(34) & "DESCRIPTION" & Chr(34) & " $COMP:" & Chr(34) & File_name & "-1" & "@" & Drawing_View & Chr(34)
1148                 swNote_DESCRIPTION = swDoc.InsertNote(Text_Add)
1149
1150                 Text_Add = "$PRPSMODEL:" & Chr(34) & "NOMENCLATURE" & Chr(34) & " $COMP:" & Chr(34) & File_name & "-1" & "@" & Drawing_View & Chr(34)
1151                 swNote_NOMENCLATURE = swDoc.InsertNote(Text_Add)
1152
1153                 Text_Add = "$PRPSMODEL:" & Chr(34) & "SPEC" & Chr(34) & " $COMP:" & Chr(34) & File_name & "-1" & "@" & Drawing_View & Chr(34)
1154                 swNote_SPEC = swDoc.InsertNote(Text_Add)
1155             End If
1156         Else
1157
1158             Text_Add = NOTE_INFO + Instance_Num(i - 1).Instance_ID & Chr(34)
1159             'MsgBox(Instance_Num(i - 1).Instance_ID)
1160             swNote_Info = swDoc.InsertNote(Text_Add)
1161
1162             If Add_Bom = True Then
1163
1164                 Text_Add = DESCRIPTION + Instance_Num(i - 1).Instance_ID & Chr(34)

```



```

        & Chr(34)
1165         swNote_DESCRIPTION = swDoc.InsertNote(Text_Add)
1166
1167         Text_Add = NOMENCLATURE + Instance_Num(i - 1).Instance_ID & Chr(34)
1168         swNote_NOMENCLATURE = swDoc.InsertNote(Text_Add)
1169
1170         Text_Add = SPEC + Instance_Num(i - 1).Instance_ID & Chr
1171         (34)
1172         swNote_SPEC = swDoc.InsertNote(Text_Add)
1173     End If
1174
1175     swAnno_Info = swNote_Info.GetAnnotation()
1176     swAnno_Info.SetAttachedEntities(sw_View)
1177     swAnno_Info.SetPosition2(x_pos, y_pos, z_pos)
1178
1179     'If New_Drawing.Add_BOM_Hardware.Checked = True Then
1180     If Add_Bom = True And i <> 0 Then
1181         swAnno_NOMENCLATURE = swNote_NOMENCLATURE.GetAnnotation()
1182         swAnno_NOMENCLATURE.SetAttachedEntities(sw_View)
1183         swAnno_NOMENCLATURE.SetPosition2(x_pos_Nom, y_pos, z_pos)
1184
1185         swAnno_DESCRIPTION = swNote_DESCRIPTION.GetAnnotation()
1186         swAnno_DESCRIPTION.SetAttachedEntities(sw_View)
1187         swAnno_DESCRIPTION.SetPosition2(x_pos_Des, y_pos, z_pos)
1188
1189         swAnno_SPEC = swNote_SPEC.GetAnnotation()
1190         swAnno_SPEC.SetAttachedEntities(sw_View)
1191         swAnno_SPEC.SetPosition2(x_pos_Spec, y_pos, z_pos)
1192     End If
1193
1194
1195     y_pos -= 0.00889
1196
1197     Next
1198
1199     'swDoc.WindowRedraw()
1200
1201     Return True
1202
1203 End Function
1204
1205
1206 Shared Function Add_Docs(ByVal swComp As Component2, ByVal nLevel As Integer)
1207
1208     'try swApp.GetDocumentDependencies2 method
1209
1210     swApp = CreateObject("SldWorks.Application")
1211
1212     Dim swPart As ModelDoc2

```

```

1213 Dim swPart2 As ModelDoc2
1214 Dim swPart3 As ModelDoc2
1215 Dim swPart4 As ModelDoc2
1216 Dim swChildComp As Component2
1217 Dim swParent As Component2
1218
1219 Dim vChildComp As Object
1220
1221 Dim Status As Boolean = False
1222 Dim Used As Boolean = False
1223 Dim Add_To_Part As Boolean = False
1224 Dim Add_To_List As Boolean = False
1225 Dim isAssy As Boolean = False
1226 Dim isPart As Boolean = False
1227
1228
1229 Dim ValOut = String.Empty
1230 Dim Dash_XX = String.Empty
1231 Dim wasResolved As Boolean
1232 Dim linkToProp As Boolean
1233 Dim Dash_Name = String.Empty
1234 Dim Temp_Name = String.Empty
1235
1236 Dim errorval As Integer
1237
1238 Dim CusProp As String() = {"PART NUMBER", "NOMENCLATURE", "DESCRIPTION", "
1239 Dim RecAssyCusProp As String() = {"N/A", "N/A", "N/A", "N/A", "N/A", "N/
1240
1241
1242 If nLevel = 1 Then
1243
1244     'swAssy_Docs.Clear()
1245     'swPart_Docs.Clear()
1246     'swDwg_Docs.Clear()
1247     swComp_Assy = swComp.Name2
1248     swPart2 = swComp.GetModelDoc2
1249
1250     If swPart2.GetType = swDocumentTypes_e.swDocASSEMBLY Then
1251
1252         swModelDocExt_Assy = swPart2.Extension
1253         CusProperties_Assy = swModelDocExt_Assy.CustomPropertyManager
1254         ("")
1255
1256         For propNum = 0 To UBound(CusProp)
1257             Dash_Name = CusProperties_Assy.Get6(CusProp(propNum), True,
1258             ValOut, RecAssyCusProp(propNum), wasResolved, linkToProp)
1259         Next
1260
1261         If CusProperties_Assy.Get6(CusProp(0), True, ValOut,
1262             RecAssyCusProp(0), wasResolved, linkToProp) = 2 Then

```

```

1260         If RecAssyCusProp(0) <> "" And RecAssyCusProp(0) <> "-XX"
1261             Then
1262                 Temp_Name = RecAssyCusProp(0).Substring(0, 1)
1263
1264                 If Temp_Name = "-" Then
1265                     Add_To_List = True
1266                 End If
1267             End If
1268
1269         End If
1270         If Add_To_List = True Then
1271             swAssy_Docs.Add(New Assy_Docs(swComp_Assy, "Null",
1272                                     swComp.GetSelectByIDString(), RecAssyCusProp(0),
1273                                     RecAssyCusProp(1), RecAssyCusProp(2), RecAssyCusProp(3),
1274                                     RecAssyCusProp(4), RecAssyCusProp(5)))
1275             End If
1276
1277         End If
1278     End If
1279
1280     vChildComp = swComp.GetChildren
1281     For i = 0 To UBound(vChildComp)
1282         Used = False
1283         Dim pString = String.Empty
1284         Dim aString = String.Empty
1285         Dim Part_To_Open = String.Empty
1286         Dim RecCusProp = {"N/A", "N/A", "N/A", "N/A", "N/A", "N/A"}
1287
1288         swChildComp = vChildComp(i)
1289
1290         If swChildComp.IsSuppressed() = False Then
1291             Add_To_List = False
1292             swParent = swChildComp.GetParent
1293
1294             If swParent Is Nothing Then
1295                 swPart = swChildComp.GetModelDoc2
1296                 aString = swChildComp.Name2
1297                 'MsgBox("Assy - " + aString)
1298
1299                 If swPart.GetType = 2 Then
1300                     isAssy = True
1301                     aString = aString.Substring(0, aString.LastIndexOf("-"))
1302                 ElseIf swPart.GetType = 1 Then
1303                     isPart = True
1304                     pString = aString.Substring(0, aString.LastIndexOf("-"))
1305                 End If
1306             Else
1307

```

```

1308 Dim tString As String
1309 swPart4 = swChildComp.GetModelDoc2
1310 swPart = swParent.GetModelDoc2
1311 If swPart4.GetType = 2 Then
1312     'MsgBox("Assy ")
1313     isAssy = True
1314 ElseIf swPart4.GetType = 1 Then
1315     'MsgBox("Part")
1316     isPart = True
1317 End If
1318
1319 tString = swParent.Name2.Substring(0,
1320                                     swParent.Name2.LastIndexOf("-"))
1321 'aString = swChildComp.Name2
1322
1323 'If swPart.GetType = 2 Then
1324 '    isAssy = True
1325 '    aString = aString.Substring(0, aString.LastIndexOf("-"))
1326 'ElseIf swPart.GetType = 1 Then
1327 '    isPart = True
1328 '    pString = aString.Substring(0, aString.LastIndexOf("-"))
1329 'End If
1330 pString = swChildComp.Name2
1331 'MsgBox("Part - " + pString)
1332
1333 While pString.IndexOf("/") <> -1
1334     pString = pString.Substring(pString.IndexOf("/") + 1)
1335 End While
1336
1337 pString = pString.Substring(0, pString.LastIndexOf("-"))
1338
1339 End If
1340
1341
1342
1343 If pString IsNot "" Then
1344
1345     If swPart_Docs.Count > 0 Then
1346         For f = 0 To swPart_Docs.Count - 1
1347             If swPart_Docs(f).subcomp = pString Then
1348                 Used = True
1349                 isAssy = False
1350                 isPart = False
1351             End If
1352         Next
1353     End If
1354
1355     If Used = False Then
1356
1357         swPart3 = swApp.ActivateDoc3(pString, 0, 1, errorval)
1358

```

```

1359         If swPart3.GetType = swDocumentTypes_e.swDocPART Then
1360
1361             swModelDocExt_Part = swPart3.Extension
1362             CusProperties_Part =
1363                 swModelDocExt_Part.CustomPropertyManager("")
1364
1365             For propNum = 0 To UBound(CusProp)
1366                 Dash_Name = CusProperties_Part.Get6(CusProp
1367                 (propNum), True, ValOut, RecCusProp(propNum), wasResolved,
1368                 linkToProp)
1369             Next
1370
1371             If CusProperties_Assy.Get6(CusProp(0), True, ValOut,
1372             RecAssyCusProp(0), wasResolved, linkToProp) = 2 Then
1373                 If RecCusProp(0) <> "" And RecCusProp(0) <> "-
1374                 XX" Then
1375
1376                     Temp_Name = RecCusProp(0).Substring(0, 1)
1377
1378                     If Temp_Name = "-" Then
1379                         Add_To_List = True
1380                     End If
1381                 End If
1382             End If
1383
1384             ElseIf swPart3.GetType = swDocumentTypes_e.swDocASSEMBLY
1385             Then
1386
1387                 swModelDocExt_Assy = swPart3.Extension
1388                 CusProperties_Assy =
1389                     swModelDocExt_Assy.CustomPropertyManager("")
1390
1391                 For propNum = 0 To UBound(CusProp)
1392                     Dash_Name = CusProperties_Assy.Get6(CusProp
1393                     (propNum), True, ValOut, RecCusProp(propNum), wasResolved,
1394                     linkToProp)
1395                 Next
1396
1397                 If CusProperties_Assy.Get6(CusProp(0), True, ValOut,
1398                 RecAssyCusProp(0), wasResolved, linkToProp) = 2 Then
1399                     If RecCusProp(0) <> "" And RecCusProp(0) <> "-
1400                     XX" Then
1401
1402                         Temp_Name = RecCusProp(0).Substring(0, 1)
1403
1404                         If Temp_Name = "-" Then
1405                             Add_To_List = True
1406                         End If
1407                     End If
1408                 End If
1409             End If

```

```
1400
1401         End If
1402         swApp.CloseDoc(pString)
1403     End If
1404
1405
1406     ElseIf swPart.GetType = swDocumentTypes_e.swDocPART Then
1407
1408         swModelDocExt_Part = swPart.Extension
1409         CusProperties_Part = ↗
1410         swModelDocExt_Part.CustomPropertyManager("")
1411
1412         For propNum = 0 To UBound(CusProp)
1413             Dash_Name = CusProperties_Part.Get6(CusProp(propNum), ↗
1414             True, ValOut, RecCusProp(propNum), wasResolved, linkToProp)
1415         Next
1416
1417         If CusProperties_Assy.Get6(CusProp(0), True, ValOut, ↗
1418         RecAssyCusProp(0), wasResolved, linkToProp) = 2 Then
1419             If RecCusProp(0) <> "" And RecCusProp(0) <> "-XX" Then
1420
1421                 Temp_Name = RecCusProp(0).Substring(0, 1)
1422
1423                 If Temp_Name = "-" Then
1424                     Add_To_List = True
1425                 End If
1426             End If
1427         End If
1428
1429     ElseIf swPart.GetType = swDocumentTypes_e.swDocASSEMBLY Then
1430
1431         swModelDocExt_Assy = swPart.Extension
1432         CusProperties_Assy = ↗
1433         swModelDocExt_Assy.CustomPropertyManager("")
1434
1435         For propNum = 0 To UBound(CusProp)
1436             Dash_Name = CusProperties_Assy.Get6(CusProp(propNum), ↗
1437             True, ValOut, RecCusProp(propNum), wasResolved, linkToProp)
1438         Next
1439
1440         If CusProperties_Assy.Get6(CusProp(0), True, ValOut, ↗
1441         RecAssyCusProp(0), wasResolved, linkToProp) = 2 Then
1442             If RecCusProp(0) <> "" And RecCusProp(0) <> "-XX" Then
1443
1444                 Temp_Name = RecCusProp(0).Substring(0, 1)
1445
1446                 If Temp_Name = "-" Then
1447                     Add_To_List = True
1448                 End If
1449             End If
1450         End If
1451     End If
```

```
1446         End If
1447
1448     End If
1449
1450     Add_To_Part = True
1451
1452     If isAssy = True And Add_To_List = True Then
1453         isAssy = False
1454         If swPart.GetType = swDocumentTypes_e.swDocASSEMBLY Then
1455             swAssy_Docs.Add(New Assy_Docs(swComp_Assy, aString,
1456                                     swChildComp.GetSelectByIDString(), RecCusProp(0), RecCusProp(
1457                                     1),
1458                                     RecCusProp(2), RecCusProp(
1459                                     3), RecCusProp(4), RecCusProp(5)))
1460         End If
1461     End If
1462
1463     If isPart = True And Add_To_List = True Then
1464         isPart = False
1465         If swPart_Docs.Count = 0 Then
1466             swPart_Docs.Add(New Part_Docs(swComp_Assy, pString,
1467                                     swChildComp.GetSelectByIDString(), RecCusProp(0), RecCusProp(
1468                                     1),
1469                                     RecCusProp(2), RecCusProp(
1470                                     3), RecCusProp(4), RecCusProp(5)))
1471             Add_To_Part = False
1472             'Else
1473             '    For q = 0 To swPart_Docs.Count - 1
1474             '        If swPart_Docs(q).subcomp = pString Then
1475             '            Add_To_Part = False
1476             '            Exit For
1477             '        End If
1478             '    Next
1479         End If
1480         If Add_To_Part = True Then
1481             swPart_Docs.Add(New Part_Docs(swComp_Assy, pString,
1482                                     swChildComp.GetSelectByIDString(), RecCusProp(0), RecCusProp(
1483                                     1),
1484                                     RecCusProp(2), RecCusProp(
1485                                     3), RecCusProp(4), RecCusProp(5)))
1486             Add_To_Part = False
1487         End If
1488     End If
1489     'Debug.Print(swChildComp.Name2)
1490     Status = Add_Docs(swChildComp, nLevel + 1)
1491 End If
1492 Next i
1493
1494 Return True
1495
1496 End Function
```

```

1489
1490
1491     Shared Function Compare()
1492         Dim status As Boolean = True
1493
1494         Return status
1495     End Function
1496
1497     Shared Function Out_Put()
1498
1499         If swAssy_Docs.Count > 0 Then
1500
1501             swAssy_Docs = swAssy_Docs.OrderByDescending(Function(x)           ↗
1502                 x.Part_Number).ToList
1503
1504             'For z = 0 To swAssy_Docs.Count - 1
1505             '    Debug.Print("Assembly - " & swAssy_Docs(z).instance_ID)
1506
1507             '    If swAssy_Docs(z).Part_Number <> "" Then
1508
1509                 '        If swAssy_Docs(z).Part_Number.Substring(0, 1) = "-" And           ↗
1510                     swAssy_Docs(z).Part_Number <> "-XX" Then
1511
1512                     '            Debug.Print("Assembly - " & swAssy_Docs(z).Comp & " :- " ↗
1513                         & swAssy_Docs(z).subcomp & " :- " &
1514                     '            " --- " & swAssy_Docs(z).Part_Number & " , " &           ↗
1515                         swAssy_Docs(z).Nomenclature &
1516                     '            " , " & swAssy_Docs(z).Description & " , " &           ↗
1517                         swAssy_Docs(z).Spec & " , " & swAssy_Docs(z).Material &
1518                     '            " , " & swAssy_Docs(z).Weight)
1519                     '            '&
1520                     '            'System.Environment.NewLine & "           ↗
1521                     " & swAssy_Docs(z).instance_ID)
1522
1523                 '        End If
1524             '    End If
1525             'Next
1526         End If
1527
1528         Debug.Print(System.Environment.NewLine & System.Environment.NewLine)
1529
1530         If swPart_Docs.Count > 0 Then
1531
1532             swPart_Docs = swPart_Docs.OrderByDescending(Function(x)           ↗
1533                 x.Part_Number).ToList
1534
1535             'For z = 0 To swPart_Docs.Count - 1
1536             '    MsgBox(swPart_Docs(z).Part_Number + " - " + swPart_Docs           ↗
1537                 (z).Comp + " - " + swPart_Docs(z).subcomp)
1538             '    If swPart_Docs(z).Part_Number <> "" Then
1539
1540                 '        If swPart_Docs(z).Part_Number.Substring(0, 1) = "-" And           ↗

```



```

        swPart_Docs(z).Part_Number <> "-XX" Then
1533
1534         '           Debug.Print("Part - " & swPart_Docs(z).Comp & " :- " &
        swPart_Docs(z).subcomp & " :-: " &
1535         '           " --- " & swPart_Docs(z).Part_Number & " , " &
        swPart_Docs(z).Nomenclature &
1536         '           " , " & swPart_Docs(z).Description & " , " &
        swPart_Docs(z).Spec & " , " & swPart_Docs(z).Material &
1537         '           " , " & swPart_Docs(z).Weight) ' &
1538         '           'System.Environment.NewLine & "
        " & swPart_Docs(z).instance_ID)
1539
1540         '           End If
1541         '       End If
1542     'Next
1543 End If
1544
1545 If swDwg_Docs.Count > 0 Then
1546
1547     For z = 0 To swDwg_Docs.Count - 1
1548         Debug.Print("Drawing - " & swDwg_Docs(z).Comp)
1549     Next
1550 End If
1551
1552
1553 Return True
1554
1555 End Function
1556
1557 Shared Function Docs_To_Excel(Path As String, Filename As String)
1558
1559     Dim xlApp As Microsoft.Office.Interop.Excel.Application
1560     Dim xlWorkBook As Microsoft.Office.Interop.Excel.Workbook
1561     Dim xlWorkSheet As Microsoft.Office.Interop.Excel.Worksheet
1562     Dim Range As Microsoft.Office.Interop.Excel.Range
1563     Dim misValue As Object = System.Reflection.Missing.Value
1564
1565     Dim i As Integer
1566     Dim j As Integer
1567     Dim Assy_row As Integer = 3
1568     Dim Assy_col As Integer = 1
1569     Dim Part_row As Integer = Assy_row + swAssy_Docs.Count + 2
1570     Dim Part_col As Integer = 1
1571
1572     xlApp = New ApplicationClass
1573
1574     Dim Assy_Array(swAssy_Docs.Count - 1, 8) As String
1575     Dim Assy_Count = 0
1576
1577     Dim Part_Array(swPart_Docs.Count - 1, 8) As String
1578     Dim Part_Count = 0
1579

```

```
1580     For Each Stringitem As Assy_Docs In swAssy_Docs
1581
1582         Assy_Array(Assy_Count, 0) = "Assembly"
1583         Assy_Array(Assy_Count, 1) = Stringitem.Comp
1584         Assy_Array(Assy_Count, 2) = Stringitem.subcomp
1585         Assy_Array(Assy_Count, 3) = Stringitem.Part_Number
1586         Assy_Array(Assy_Count, 4) = Stringitem.Nomenclature
1587         Assy_Array(Assy_Count, 5) = Stringitem.Description
1588         Assy_Array(Assy_Count, 6) = Stringitem.Spec
1589         Assy_Array(Assy_Count, 7) = Stringitem.Material
1590         Assy_Array(Assy_Count, 8) = Stringitem.Weight
1591
1592         Assy_Count = +1
1593
1594     Next
1595
1596     For Each Stringitem As Part_Docs In swPart_Docs
1597
1598         Part_Array(Part_Count, 0) = "Part"
1599         Part_Array(Part_Count, 1) = Stringitem.Comp
1600         Part_Array(Part_Count, 2) = Stringitem.subcomp
1601         Part_Array(Part_Count, 3) = Stringitem.Part_Number
1602         Part_Array(Part_Count, 4) = Stringitem.Nomenclature
1603         Part_Array(Part_Count, 5) = Stringitem.Description
1604         Part_Array(Part_Count, 6) = Stringitem.Spec
1605         Part_Array(Part_Count, 7) = Stringitem.Material
1606         Part_Array(Part_Count, 8) = Stringitem.Weight
1607
1608         Part_Count += 1
1609
1610     Next
1611
1612     xlWorkBook = xlApp.Workbooks.Open("T:\Engineering\Non-Site Specific      ↗
1613         \PARTS\SW Macros\Model Creator\Resources\Part Numbers.xlsx")
1614     xlApp.WindowState = xlWindowState.xlMaximized
1615     xlApp.Visible = True
1616
1617     xlWorkSheet = xlWorkBook.Sheets(1)
1618
1619     With xlWorkSheet
1620         Range = .Range(.Cells(Assy_row, Assy_col), .Cells(UBound(Assy_Array, ↗
1621             1) - LBound(Assy_Array, 1) + Assy_row, UBound(Assy_Array, 2) - ↗
1622             LBound(Assy_Array, 2) + Assy_col))
1623
1624     End With
1625     Range.Value = Assy_Array
1626
1627     With xlWorkSheet
1628         Range = .Range(.Cells(Part_row, Part_col), .Cells(UBound(Part_Array, ↗
1629             1) - LBound(Part_Array, 1) + Part_row, UBound(Part_Array, 2) - ↗
1630             LBound(Part_Array, 2) + Part_col))
1631
1632     End With
```

```
1627     Range.Value = Part_Array
1628
1629     Path = Path + "\" + Filename + " - Doc Data.xlsx"
1630     xlWorkbook.SaveAs(Path)
1631     'xlWorkbook.Close()
1632
1633     Return True
1634 End Function
1635
1636 Shared Function Name_Tables(swFeat_Table As Feature)
1637
1638     Dim swGeneralTableFeature As GeneralTableFeature
1639     Dim swTableAnnotation As TableAnnotation
1640     Dim nbrTableAnnotations As Integer
1641     Dim Insert_WeightTable As Boolean = True
1642     Dim Insert_BOMTable As Boolean = True
1643     Dim Columns_max As Integer
1644     Dim tableAnnotations() As Object
1645
1646
1647     swGeneralTableFeature = swFeat_Table.GetSpecificFeature2
1648     nbrTableAnnotations = swGeneralTableFeature.GetTableAnnotationCount
1649     tableAnnotations = swGeneralTableFeature.GetTableAnnotations
1650     swTableAnnotation = tableAnnotations(0)
1651     Columns_max = swTableAnnotation.ColumnCount()
1652     If Columns_max = 4 Then
1653
1654         swFeat_Table.Name = "WEIGHT TABLE"
1655         Insert_WeightTable = False
1656     ElseIf Columns_max > 4 Then
1657
1658         swFeat_Table.Name = "PART LIST TABLE"
1659         Insert_BOMTable = False
1660     Else
1661
1662     End If
1663
1664
1665     Return True
1666     'Return Insert_WeightTable, Insert_BOMTable
1667
1668
1669 End Function
1670
1671 End Class
1672
1673
```