Top Ten CATIA V5 CATScript Macros

Emmett Ross

The following programs are the snippets of code I find myself using the most. While each piece of code by itself may not be the final solution to your problem, they can quickly and easily be placed into larger, more complicated programs. You can increase your programming speed by reusing snippets of code you've already written that you know work. Feel free to use these in any way that will benefit you.

How to Run Macros in CATIA V5

To run the macros:

1. Go to Tools>Macro>Macros (or hit Alt+F8) to open the macro window.

2. Create a new macro of type **CATScript.**

3. Copy and paste the code into the CATScript editor.

4. Save

5. Run

Once a CATScript macro is created, there are multiple ways to open the macros window to run the code:

1. **Go to Tools>Macro>Macros**
2. CATIA macros window keyboard shortcut: **Alt+F8**
3. Assign an icon (or create your own) for each macro

Quick Start Guide to CATIA V5 Macros

Never created or ran a macro in CATIA before? Join my CATIA Macro Email Course for a quick start guide on how you can begin saving yourself tons of time and completing your projects faster.

http://www.scripting4v5.com/catia-macro-email-course

10. What is this?

This macro returns the name and type of all objects selected before the program is run. The purpose of this program is to help you figure out the syntax for pieces of geometry. For example, you can find out if a plane is HybridShapePlaneExplicit or HybridShapePlaneNormal.

```
Sub CATMain()


     Dim oSel As Selection
     Set oSel = CATIA.ActiveDocument.Selection

     MsgBox "Number of objects selected: " & oSel.Count

     Dim i As Integer

     For i =1 to oSel.Count

          Dim oSelEl As SelectedElement
          Set oSelEl = oSel.Item(i)

Msgbox " The selected element's name is " & oSelEl.Value.Name
Msgbox "The Selected element's type is " & oSelEl.Type



Next

oSel.Clear

End Sub
```

9. Search by Name

Do you have an enormous specification tree with thousands of pieces of geometry? Do you need to find one specific element within the tree? Rid yourself hours of endless scrolling by searching for those components directly by name. The search is not case sensitive.

```
Sub CATMain()


Dim oSelection as Selection
Set oSelection = CATIA.ActiveDocument.Selection


Dim iCount


Dim geoName As String
geoName = Inputbox("Please enter EXACT name of geometry to
search for.")


oSelection.Search "Name=" & geoName & "*,all"


iCount = oSelection.Count


msgbox "Number of shapes found: "&icount


For i=1 to iCount


CATIA.StartCommand "Center Graph"


Next
End Sub
```

8. Launch Excel from CATIA

One of the biggest reasons for learning CATIA macro programming is to be able to automatically export properties from CATIA to an Excel spreadsheet. Export to Excel is highly useful as you can create custom bills of material (BOMs) or part lists at the click of a button. The first step to exporting any information from inside CATIA is to be able to launch or open Excel. Use the code shown below.

```
Sub CatMain()

Dim Excel As Object

On Error Resume Next

'if Excel is already running, then get the Excel object

     Set Excel = GetObject(, "Excel.Application")

     If Err.Number <> 0 Then

'If Excel is not already running, then create a new session

          Set Excel = CreateObject("Excel.Application")

          Excel.Visible = True

          End If

'add a new workbook
        Excel.Workbooks.Add

'set the workbook as the active document
        Set WBK = Excel.ActiveWorkbook.Sheets(1)

End Sub
```

7. Screen capture

Automate the process of taking and saving images of your CATParts and CATProducts by using a screen capture macro. The images can be saved to a file or inserted directly into PowerPoint. This macro takes a screen shot with a white background and saves it in a folder.

```
Sub CATMain()

Dim ObjViewer3D As Viewer3D
Set objViewer3D = CATIA.ActiveWindow.ActiveViewer

Dim objCamera3D As Camera3D
Set objCamera3D = CATIA.ActiveDocument.Cameras.Item(1)

'Input box to name the screen capture image file
Dim partname As String
partName = Inputbox ("Please name the image.")
If partName="" Then
MsgBox "No name was entered. Operation aborted.", vbExclamation,
"Cancel"
Else
'turn off the spec tree
Dim objSpecWindow As SpecsAndGeomWindow
Set objSpecWindow = CATIA.ActiveWindow
objSpecWindow.Layout = catWindowGeomOnly
'Toggle Compass
CATIA.StartCommand("Compass")
'change background color to white
Dim DBLBackArray(2)
objViewer3D.GetBackgroundColor(dblBackArray)
Dim dblWhiteArray(2)
dblWhiteArray(0) = 1
dblWhiteArray(1) = 1
dblWhiteArray(2) = 1
objViewer3D.PutBackgroundColor(dblWhiteArray)
```

```
'file location to save image
Dim fileloc As String
fileloc = "C:\User\Macro Files\"

Dim exten As String
exten = ".png"

Dim strName as string
strname = fileloc & partName & exten

If MsgBox ("To reframe and automatically switch to ISO view
click Yes. To take the image as shown on screen click No.",
vbYesNo) = vbYes Then

objViewer3D.Viewpoint3D = objCamera3D.Viewpoint3D
'reframe
objViewer3D.Reframe()

'zoom in
objViewer3D.ZoomIn()

'clear selection for picture
        CATIA.ActiveDocument.Selection.Clear()

'increase to fullscreen to obtain maximum resolution
objViewer3D.FullScreen = True
'MsgBox strname

'take picture with auto ISO view and reframe ON***
objviewer3D.Capturetofile 4,strname
```

```
'*********take picture as is with NO reframe or iso view

Else
'zoom in
'objViewer3D.ZoomIn()

'clear selection for picture
        CATIA.ActiveDocument.Selection.Clear()

'increase to fullscreen to obtain maximum resolution
objViewer3D.FullScreen = True

'take picture
objviewer3D.Capturetofile 4,strname

End If

'*******************RESET**********************
objViewer3D.FullScreen = False

'change background color back
objViewer3D.PutBackgroundColor(dblBackArray)

'turn the spec tree back on
objSpecWindow.Layout = catWindowSpecsAndGeom

'toggle compass
CATIA.StartCommand("Compass")
End If
End Sub
```

6. Product to design mode

If you’re going to run a macro on a CATProduct, usually the first thing you will want to do is to check to see if the top level product document is in design mode. If not then this macro will display a message box asking the user’s permission to automatically switch the product document to design mode. Assumes the active document is a CATProduct.

```
Sub CATMain()


Dim prdRoot As Product
Set prdroot = CATIA.ActiveDocument.Product
'check if a product is in Design Mode

If (prdRoot.PartNumber = “”) Then
' propose user to switch it in design mode
Dim vbResponse

vbResponse = MsgBox(“Product ” & prdRoot.Name & ” is not in
design mode. Would you like to switch it?”, vbYesNo, “Warning”)


If (vbResponse = vbYes) Then

prdRoot.ApplyWorkMode DESIGN_MODE

End If

Else

Msgbox “product already in design mode”

End If

End Sub
```

5. Export Specification Tree

To get a quick list of all the components in the specification tree, you can quickly export it to either Excel or TXT format. Assumes active document is a CATProduct.

```
Sub CATMain()
Dim productDocument1 As Document
Set productDocument1 = CATIA.ActiveDocument

'Input box to select txt or xls
Dim exportFormat As String
exportFormat = Inputbox ("Please choose format to export the
tree as._      Type either 'xls' or 'txt'")

IF exportFormat <> "xls" THEN
IF exportFormat <> "txt" THEN

MsgBox "Did not enter txt or xls. Program cancelled please retry
macro."

Else

'Input box to enter name of file
Dim partName As String
partName = Inputbox ("Please enter the file name.")

'Input box to enter file location
Dim oLocation As String
oLocation = "C:\Macro Files\"
productDocument1.ExportData oLocation & partName & "." & _
exportFormat,"txt"

End If
End If
End Sub
```

4. Create Drawing Frame and Title Block

The macro recorder doesn’t work in the Drafting workbench, so it’s always a good idea to keep a couple of drafting macros around to see the syntax. Instead of using a template, use this macro to create a drawing frame and title block on the fly. Each step is explained with notes inside the code.

```
Option Explicit
Sub CATMain()
    Dim oDrwDoc As DrawingDocument
    Set oDrwDoc = CATIA.ActiveDocument
'memorize which View was active before the macro
    Dim oViewerActive As DrawingView
    Set oViewerActive =
oDrwDoc.DrawingRoot.ActiveSheet.Views.ActiveView
'get the background View
    Dim oView As DrawingView
    Set oView =
oDrwDoc.DrawingRoot.ActiveSheet.Views.Item("Background View")
    oView.Activate
'create a Frame
    Dim oFact As Factory2D
    Set oFact = oView.Factory2D
    Dim dW As Double
    dW = oDrwDoc.DrawingRoot.ActiveSheet.GetPaperWidth
    Dim dH As Double
    dH = oDrwDoc.DrawingRoot.ActiveSheet.GetPaperHeight
    Dim oLine As Line2D
‘draw the frame 10 units in from the edges of the paper
    Set oLine = oFact.CreateLine(10#, 10#, dW - 10#, 10#)
    Set oLine = oFact.CreateLine(dW - 10#, 10#, dW - 10#, dH -
10#)
    Set oLine = oFact.CreateLine(dW - 10#, dH - 10#, 10#, dH -
10#)
    Set oLine = oFact.CreateLine(10#, dH - 10#, 10#, 10#)

‘text objects are used to write notes on the drawing
    Dim oNote As DrawingText
    Set oNote = oView.Texts.Add(" ", 100#, 100#)
    oNote.Text = "GENERAL NOTES - UNLESS OTHERWISE SPECIFIED"
    oNote.Text = oNote.Text & vbCrLf & "1. Dimensions are mm"
    oNote.Text = oNote.Text & vbCrLf & "2. Edges to be deburred"
    oNote.AnchorPosition = catTopLeft
    oNote.SetFontSize 0, 0, 3.5
    oNote.WrappingWidth = 200#
```

```
'create the title block as a table
    Dim oTables As DrawingTables
    Set oTables = oView.Tables

    Dim oTable As DrawingTable
    Set oTable = oTables.Add(dW - 360#, 70#, 3#, 3#, 20#, 90#)
    oTable.SetColumnSize 1, 90
    oTable.SetColumnSize 2, 200
    oTable.SetColumnSize 3, 60
    oTable.MergeCells 1, 1, 1, 3
    oTable.SetCellString 1, 1, "TITLE BLOCK"
    oTable.SetCellAlignment 1, 1, CatTableMiddleCenter

    'set the font sizes for each table cell
    oTable.GetCellObject(1, 1).SetFontSize 0, 0, 7
    oTable.GetCellObject(2, 1).SetFontSize 0, 0, 5
    oTable.GetCellObject(3, 1).SetFontSize 0, 0, 5
    oTable.GetCellObject(2, 2).SetFontSize 0, 0, 7
    oTable.GetCellObject(3, 2).SetFontSize 0, 0, 7
    oTable.GetCellObject(2, 3).SetFontSize 0, 0, 5
    oTable.GetCellObject(3, 3).SetFontSize 0, 0, 5

‘use the VB date function to display the current date
    oTable.SetCellString 2, 1, "DATE: " & Date
    oTable.SetCellAlignment 2, 1, CatTableMiddleCenter


    oTable.SetCellString 3, 1, "SCALE: 1:" &
oDrwDoc.DrawingRoot.ActiveSheet.Scale
    oTable.SetCellAlignment 3, 1, CatTableMiddleCenter


‘use the count property to count the total number of sheets
    oTable.SetCellString 2, 3, "SHEET 1 of " &_
oDrwDoc.DrawingRoot.Sheets.Count
    oTable.SetCellAlignment 2, 3, CatTableMiddleCenter
```

```
‘link to the part document to displayed the desired properties
    Dim oPrtDoc As PartDocument
    Set oPrtDoc = CATIA.Documents.Item("Part1.CATPart")

‘get the front view
    Dim oViewF As DrawingView
    Set oViewF = 
oDrwDoc.DrawingRoot.ActiveSheet.Views.Item("Front")

    Dim oParent As Product
    Set oParent = oViewF.GenerativeBehavior.Document

‘display the part number from the CATPart in the table
    oTable.SetCellString 2, 2, "PART NO: " & oParent.PartNumber
    oTable.SetCellAlignment 2, 2, CatTableMiddleCenter

‘display the mass of the part model in the table
    oTable.SetCellString 3, 2, "WEIGHT: " & oParent.Analyze.Mass 
& "kg"
    oTable.SetCellAlignment 3, 2, CatTableMiddleCenter

    oView.SaveEdition

    'activate the originally active view
    oViewerActive.Activate

    'update everything in the end
    oDrwDoc.DrawingRoot.Update

End Sub
```

3. Save Drawings as PDF

One of the best uses of macros is for batch processes, such as converting CATDrawings into PDF. If you have a folder of a hundred drawings, this macro will convert each one into a PDF, a process that would take hours if done manually.

```
Sub CatMain()
Dim fileSys
Set fileSys = CATIA.FileSystem
Dim FolderPath
FolderPath = InputBox( "Enter a folder path:", "Folder path to
convert the drawings" ,sDocPath & "P:\DrawingtoPDF")


Dim filefolder
Set filefolder = FileSys.GetFolder(FolderPath)


Dim i as Integer


'loop through all files in the folder
For i = 1 To filefolder.Files.Count
Dim IFile
Set IFile = filefolder.Files.Item(i)


'if the file is a CATDrawing, then open it in CATIA
If InStr(IFile.Name, ".CATDrawing") <> 0 Then
Dim Doc
Set Doc = CATIA.Documents.Open(IFile.Path)
Set partDocument1 = CATIA.ActiveDocument
Dim drawingName as String
drawingName =len(CATIA.ActiveDocument.Name)
pdfName =left( CATIA.ActiveDocument.Name,drawingName-11)
'msgbox partt
PartDocument1.ExportData FolderPath &"\"& pdfName, "pdf"


'close the open drawing document
CATIA.ActiveDocument.Close()
End IF


Next 'go to the next drawing in the folder
End Sub
```

2. Find and Delete Deactivated Features

I’ve created a very useful CATIA macro to help clean up my CATIA files before I send them to my customers by automatically deleting all deactivated features. This helps keep the file size down and the tree looks cleaner. The code has two main steps:

1. Displays the number of deactivated features within a part document
2. Gives the user the option to delete all the deactivated components (minus sketches)

When I’m working on a CATIA part model making numerous changes I don’t delete anything as I am going along in case I need to go back and use what I made. My design philosophy is that it’s a lot easier and quicker to delete something than to create it from scratch. Therefore, I simply deactivate components I don’t need for now – that way I can activate them again if I find out I actually do need them later on saving me a lot of time in the process. When my model is complete and I’m ready to send the results to the customer I simply run my clean up macro which finds and deletes all of those deactivated components within the CATPart file.

```
'this CATScript macro deletes all deactive components _
          (except for sketches)
Sub CATMain()
'error handling
On Error Resume Next

Dim partDocument1 'As Document
Set partDocument1 = CATIA.ActiveDocument

Dim part1 As Part
Set part1 = partDocument1.Part

If Err.Number=0 Then

        Dim selection1 'As Selection
        Set selection1 = partDocument1.Selection
      selection1.Search "CATPrtSearch.PartDesign
Feature.Activity=FALSE"

        'if no deactivated components then end program
        If selection1.Count = 0 Then
                        Msgbox "No deactivated features."
                        Exit Sub
        Else
If MsgBox ("The number of deactivated components is: "&_
selection1.Count & ".  Click yes to delete or click no to
exit.",_ vbYesNo) = vbYes Then
'delete all deactivated components then update the part
                        selection1.Delete
                        part1.Update
                End If
        End If
'error handling
Else
        Msgbox "Not a part document! Open a single part
document."
End If
End Sub
```

1. Recursively Scroll through the Tree

This macro will walk down the tree and display the part number for every component and if it is a part or product. There are many uses for being able to scroll down through every item in the tree, whether you’re just counting the total number of parts, taking a screenshot of every component, or exporting data to Excel. The uses for this code are endless and I find I am always referring back to it.

```
Sub CATMain()
    'Get the current CATIA assembly
    Dim oProdDoc As ProductDocument
    Set oProdDoc = CATIA.ActiveDocument
    Dim oRootProd As Product
    Set oRootProd = oProdDoc.Product
    'Begin scroll down the specification tree
     Call WalkDownTree(oRootProd)
End Sub
'---------------------------------------------------------------
' WalkDownTree is a recursive function to scroll down the spec
tree and output names of each item
Sub WalkDownTree(oInProduct As Product)


   Dim oInstances As Products
   Set oInstances = oInProduct.Products
'-----No instances found then this is CATPart


   If oInstances.Count = 0 Then


     Msgbox "This is a CATPart with part number " &
oInProduct.PartNumber
Exit Sub
   End If
'-----Found an instance therefore it is a CATProduct
Msgbox "This is a CATProduct with part number " &
oInProduct.ReferenceProduct.PartNumber


Dim k As Integer
     For k = 1 To oInstances.Count
          Dim oInst As Product
          Set oInst = oInstances.Item(k)

          'apply design mode
          oInstances.Item(k).ApplyWorkMode DESIGN_MODE
          Call WalkDownTree(oInst)
      Next
End Sub
```

# # #

Thanks for reading!

Questions or comments? I'd love to hear how you are able to use this macros in your own work and projects. Email me: Emmett@scripting4v5.com

To learn how to setup your Macro Libraries in order to implement these macros, join my Quick Start Guide to CATIA V5 Macro Programming. I'll teach you how to program your own macros so you will be able to customize these codes for your specific needs.

http://www.scripting4v5.com/catia-macro-email-course