iLogic Design for Success

Jason Hunt
Designer IV, Group Lead
FS-Elliott Co., LLC

About the Speaker

- NPD Lead Designer for a global centrifugal compressor manufacturer, FS-Elliott Co., that is headquartered outside of Pittsburgh, Pa.
- Based out of the Williamsville, NY office.
- Over 20 years of compressor design experience (Rotary and Centrifugal)
- Provides lead design services to current New Product Development (NPD) projects and helps drive current CAD standards and best practices in the NPD team.
- Second time presenting at AU and third time attending AU.
- Have been a member of AUGI® (NAAUG) since 2012

Lab Assistants

Kevin Smedley *Engineering Systems Manager, FS-Elliott Co., LLC*

- PL20876 – A Manufacturing CAD Environment, Admin Style

Mike Ostrowski *CAD Manager, FS-Elliott Co., LLC*

Todd Schmoock Applications Consultant, Synergis Technologies, LLC

- GEN21060 – Migrating from AutoCAD to AutoCAD Electrical

Our Success Story

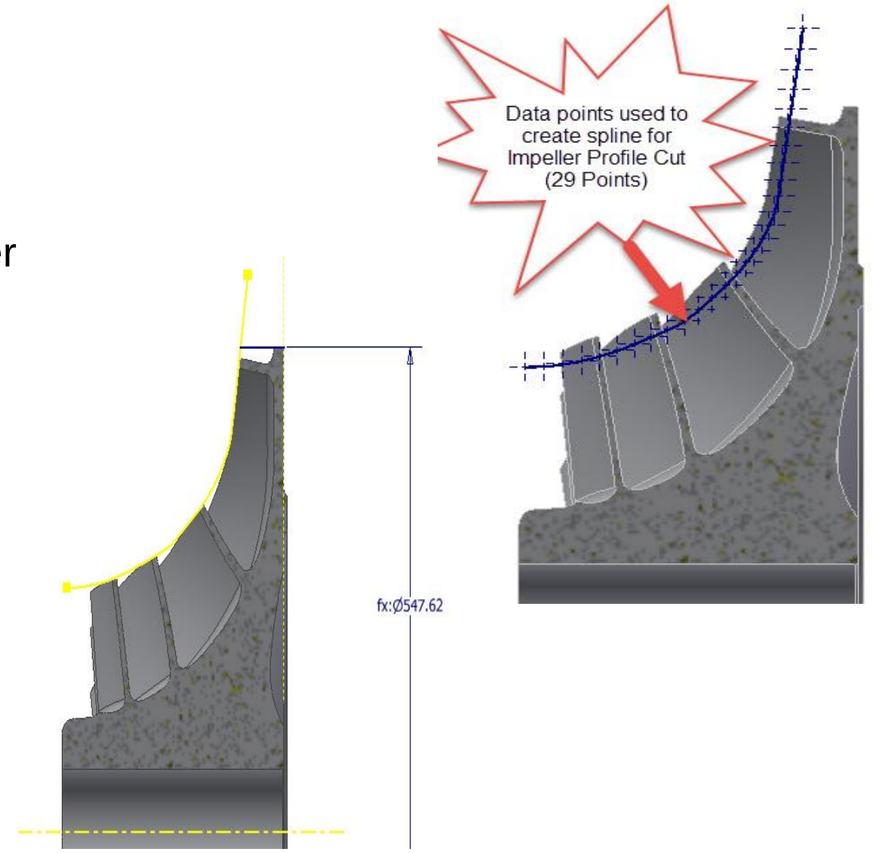
Impeller Model Background

- In the beginning, the impeller model to the right would take one to two days to model manually
- We then progressed to create a master model using iLogic and spreadsheets.
  - New program created 70 impeller models in an hour.
  - The data had to be consistent, no variations
  - Wouldn't work with legacy data (not flexible)
- Currently we have a program that does the following:
  - Entire model is created via iLogic
  - Data doesn't have to be consistent
  - Works with any design (flexible)
  - Creates about 70 impeller models in an hour
- Todays Lab is based on a simple fictitious impeller model example.

Class summary

Today, we are going to learn how to write a little more advanced iLogic code that will allow a designer to efficiently automate design variations of an impeller model.

- Embed a spread sheet to assist you in writing iLogic code.
- The lab begins with a short lecture discussing the following:
  - Reasons to automate your designs
  - Touch on iLogic basics
  - Using API, “For Next” loops & “Try-Catch” statements to create user parameters.
- Programming Challenges 1 & 2
- After these two challenges we will have a short discussion on:
  - “Do While” loops,
  - Embedded spread sheets
  - iLogic forms as a user interface
- Programming Challenges 3 & 4
- Conclusions



Key learning objectives

At the end of this class, you will be able to:

- Explore the reasons to automate designs, via iLogic
- Use the API to generate user parameters
- Utilize “For Next” & “Do While” loops in iLogic programming to automate the creation of multiple parameters and parts
- Discover the power of using an embedded spread sheet in tandem with iLogic

Automating Your Design
“Models”

Why Does Automating Your Design Matter?

Questions to Ask:

- Is the work repetitive?
- Will automation save time, so a designer can focus on other tasks?
- Is there a need or desire to improve consistency?
- Do you want to improve efficiency?
- Does the data output need to be provided in a certain manner, every time?
- Do you need to create a lot of models in a short amount of time?

If you answered yes to any of the questions above, then using iLogic to automate your design should be an important facet of your design process.

iLogic Basics

iLogic Definition

What is iLogic?

- iLogic enables rule driven design that provides a simple way to capture and reuse your work. It allows the user to standardize and automate the design process.

- iLogic allows you to become a coding expert without having to learn much actual code

iLogic Rules

What are iLogic Rules?

- A Visual Basic (VB.NET) program
  - Monitors and controls Inventor parameters, features, or components.

Rule Types

- Internal Rules: iLogic Rules saved within a document
  - Used on one part (local)
- External Rules: Saved on your local or network drive.
  - Used Globally

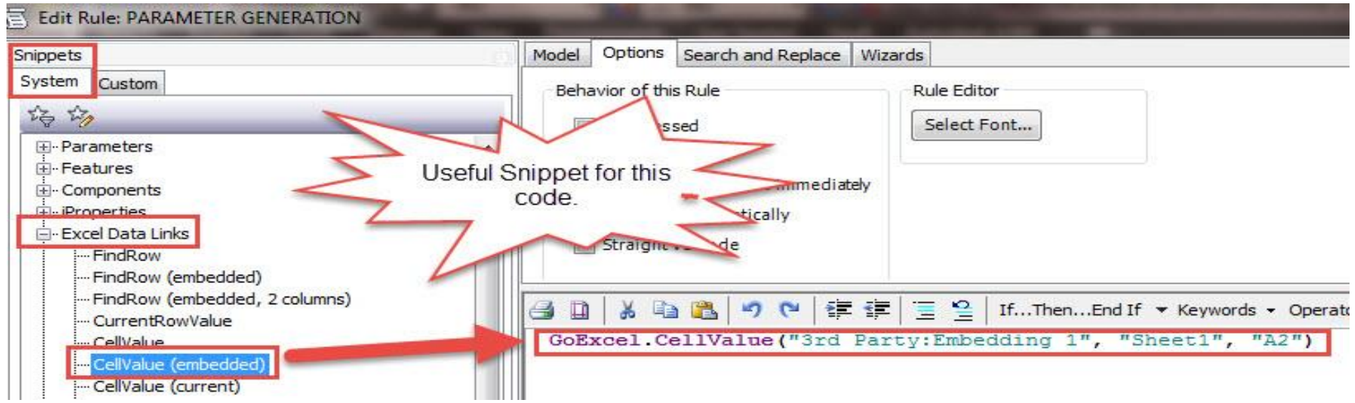
iLogic Code Snippets

What are Snippets?

- Shortcuts, for frequently used pieces of code.
  - Using snippets allows the user to insert them into your code that you would normally have to type in manually.
  - Using snippets also helps reduce the possibility of errors in your program, due to typographical errors.

- Snippets can be found in the area of the Edit Rule dialog box. This area features two tabs:

  - The ***System tab*** includes a set of predefined snippets, arranged by category.
  - The ***Custom tab*** allows you to add your own snippets, or create custom copies of System snippets.

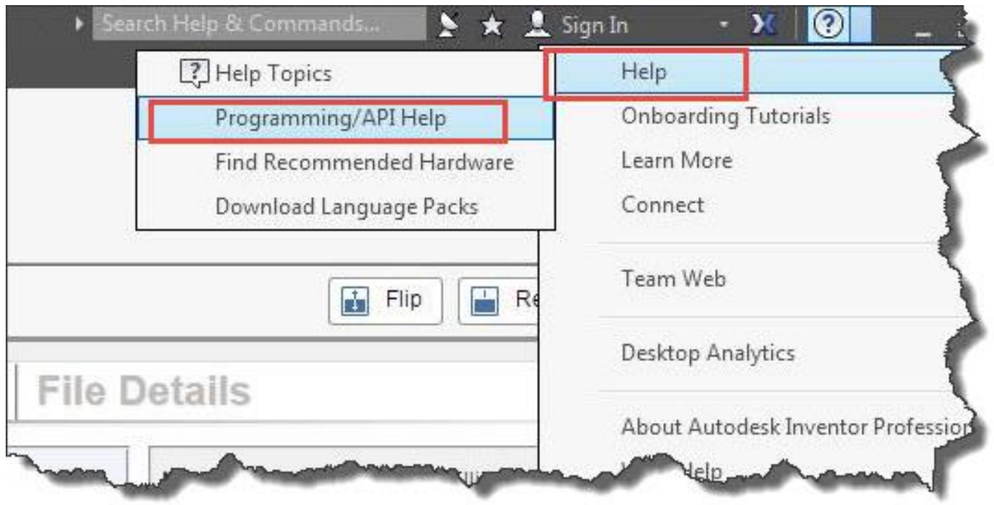


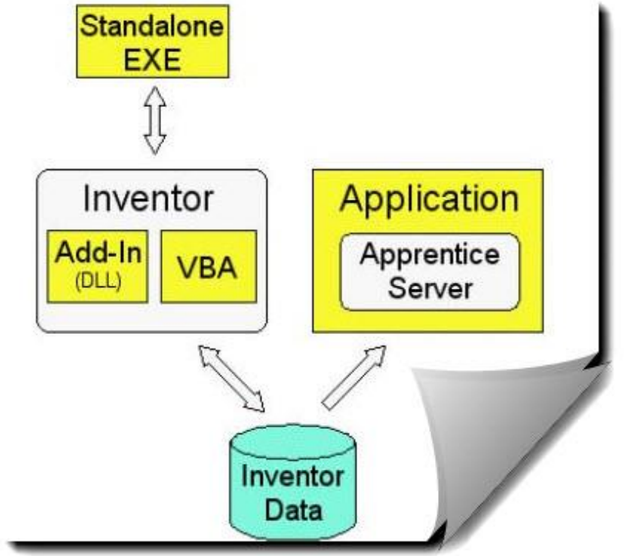
API & “For Next” Loops

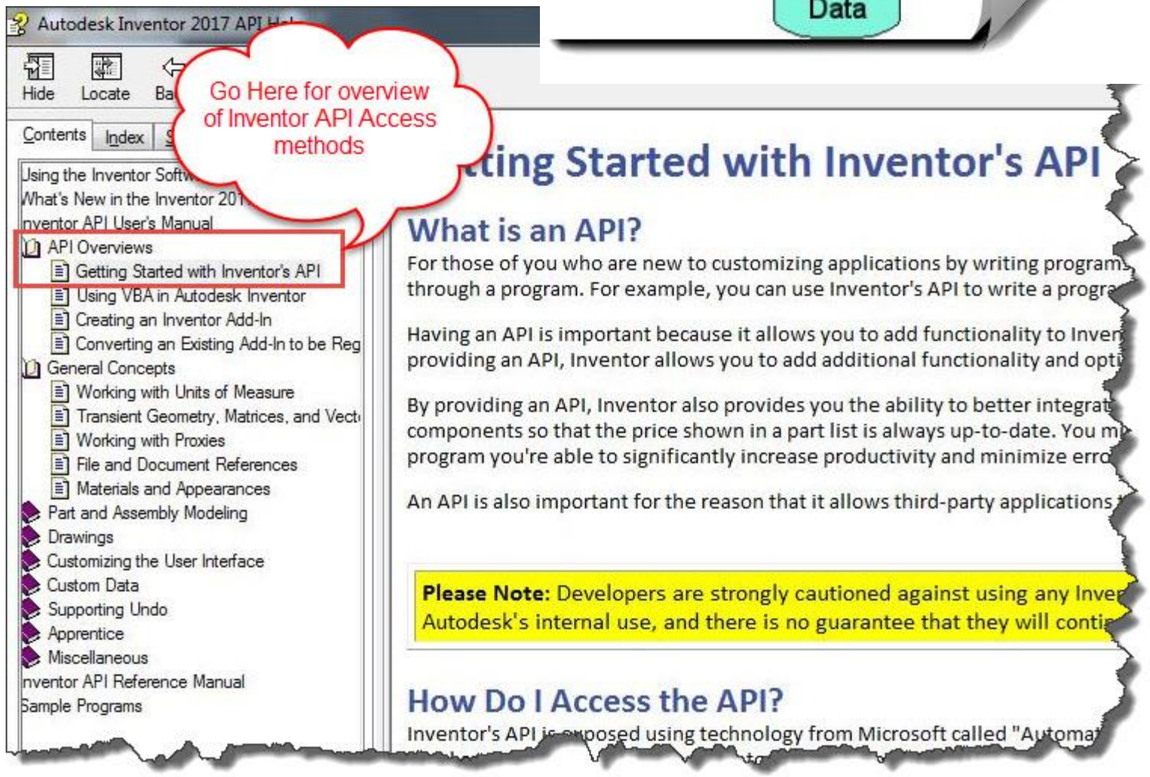
API

What is API?

- API is an interface that allows the user write a program that will perform the same types of operations that would normally be able to use when working in inventor interactively.
- The diagram shown to the right, from the Inventor API Help site, illustrates the different ways to access Inventor's API.
- This lab accesses API using Inventor
- Utilizing API help







API & Parameter Creation

Some API expressions to create user parameters are as follows:

- Write this expression to save typing time in your code, as it is repeated a lot. Only need to define this once in your code.

```
'----------SHORTENS THE AMOUNT OF TEXT TO TYPE WHEN CREATING PARAMETER CODE
oMyParameter=ThisApplication.ActiveDocument.ComponentDefinition.Parameters.UserParameters

'----------CREATE A UNITLESS PARAMETER
oParameter=oMyParameter.AddByExpression("UNITLESS", 0, "ul")

'----------CREATE A USER DEFINED TEXT PARAMETER
 oParameter=oMyParameter.AddByValue("TEXT", "TEST", UnitsTypeEnum.kTextUnits)

'----------CREATE A USER DEFINED PARAMETER WITH MILLIMETER UNITS
 oParameter=oMyParameter.AddByExpression("MM", "0", UnitsTypeEnum.kMillimeterLengthUnits)

'----------CREATE A USER DEFINED PARAMETER WITH INCH UNITS
oParameter=oMyParameter.AddByExpression("IN", "0", UnitsTypeEnum.kInchLengthUnits)

'----------CREATE A USER DEFINED ANGLE PARAMETER
oParameter=oMyParameter.AddByExpression("ANGLE", "0", UnitsTypeEnum.kDefaultDisplayAngleUnits)
```

“For Next” Loops & “Try-Catch” Statements

- The “For Next” loop iterates through all 29 user parameter scenarios to allow the “Try-Catch” statement to check if the user parameter has been created. If it hasn’t been created the user parameter will be created through the code.

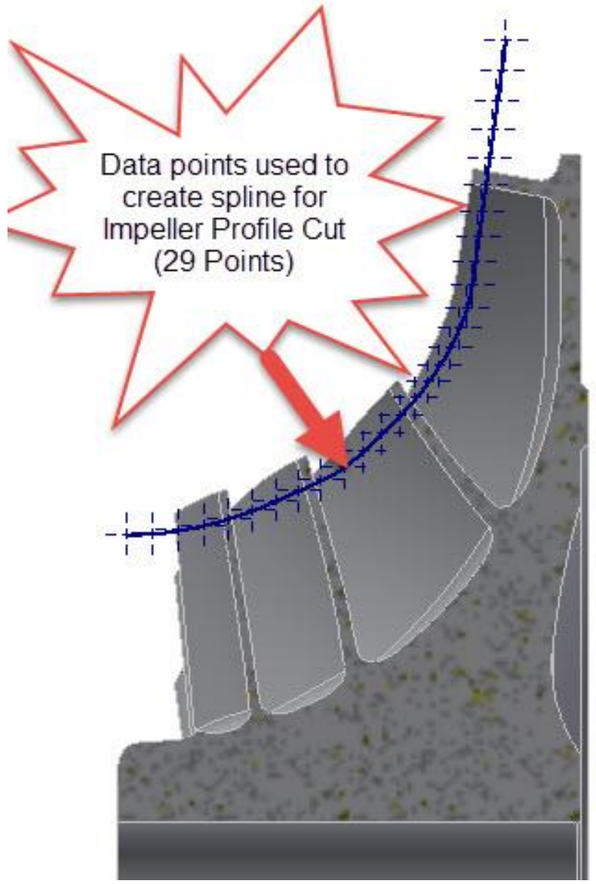
- The code will check to see if the 29 user parameters (R1, R2, R3,……R28, R29) have been created, using the try block.

- If an error exists from the Try block (the user parameter doesn’t exist) then the Catch clause will create the user parameter.

```
'----------SHORTENS THE AMOUNT OF TEXT TO TYPE WHEN CREATING PARAMETER CODE
oMyParameter=ThisApplication.ActiveDocument.ComponentDefinition.Parameters.UserParameters
'----------FOR STATEMENT TO GENEREATE ALL THE PARAMETERS
  For Y = 1 To 29
'----------R DATA POINT PARAMETER CHECK
    Try
      oTest1 = Parameter(R & Y)
    Catch
'----------SETTING UP R DATA POINT USER PARAMETERS AS MILLIMETERS
    oParameter=oMyParameter.AddByExpression(R & Y, "0", UnitsTypeEnum.kMillimeterLengthUnits)
    End Try

'----------INCREMENTS THE DATA POINT PARAMETERS FOR Z & R
  Next
```

Programming Challenge #1



The Purpose of this Exercise is to show you how to:

- Generate Impeller Shroud Data Point Parameters Using API and “For Next Loops”
- Utilize “Try & Catch” statements to avoid creating a duplicate parameter
- Embed an Excel Document into an Inventor File

“Programing Challenge 1 - PD20790-L.docx” located in:

C:\Datasets\Lab02 San Polo 3403\PD20790-L\Programming Challenges

7-12 Minutes

Programming Challenge #1 Code

```
'-----------------MESSAGE BOX ASKING THE USER IF THEY WANT TO CONTINUE
U = MessageBox.Show("This operation will generate parameters for 29 impeller shroud profile data points. Pr
If U = 2 Then
    Exit Sub
Else
```

*****This is where you will start entering your code*****

```
'------SHORTENS THE AMOUNT OF TEXT TO TYPE WHEN CREATING PARAMETER CREATION CODE
  oMyParameter=ThisApplication.ActiveDocument.ComponentDefinition.Parameters.UserParameters

'----------------FOR STATEMENT TO GENEREATE ALL THE PARAMETERS
  For Y = 1 To 29
'-----------------R DATA POINT PARAMETER CHECK TO SEE IF THE PARAMETERS EXISIT
    Try
      oTest1 = Parameter(R & Y)
    Catch
'------------------ CREATING R DATA POINT USER PARAMETERS AS MILLIMETERS
    oParameter=oMyParameter.AddByExpression(R & Y, "0", UnitsTypeEnum.kMillimeterLengthUnits)
    End Try
'-----------------Z DATA POINT PARAMETER CHECK TO SEE IF THE PARAMETERS EXISIT
    Try
      oTest2 = Parameter(Z & Y)
    Catch
'-----------------CREATING Z DATA POINT USER PARAMETERS AS MILLIMETERS
    oParameter=oMyParameter.AddByExpression(Z & Y, "0", UnitsTypeEnum.kMillimeterLengthUnits)
    End Try
'---------------INCREMENTS THE DATA POINT PARAMETERS FOR Z & R
  Next
```

Programming Challenge #2

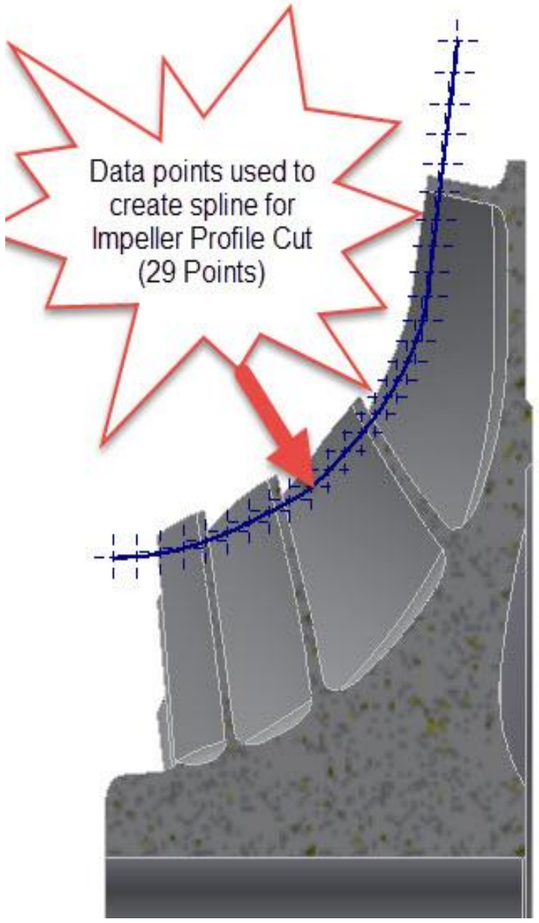
The Purpose of this Exercise is to show you how to:

- Write a simple program to populate Impeller Shroud Data Point Parameters from an embedded Excel file, using “For Next Loops”.

“Programing Challenge 2 - PD20790-L.docx” located in:

C:\Datasets\Lab02 San Polo 3403\PD20790-L\Programming Challenges

5-10 Minutes



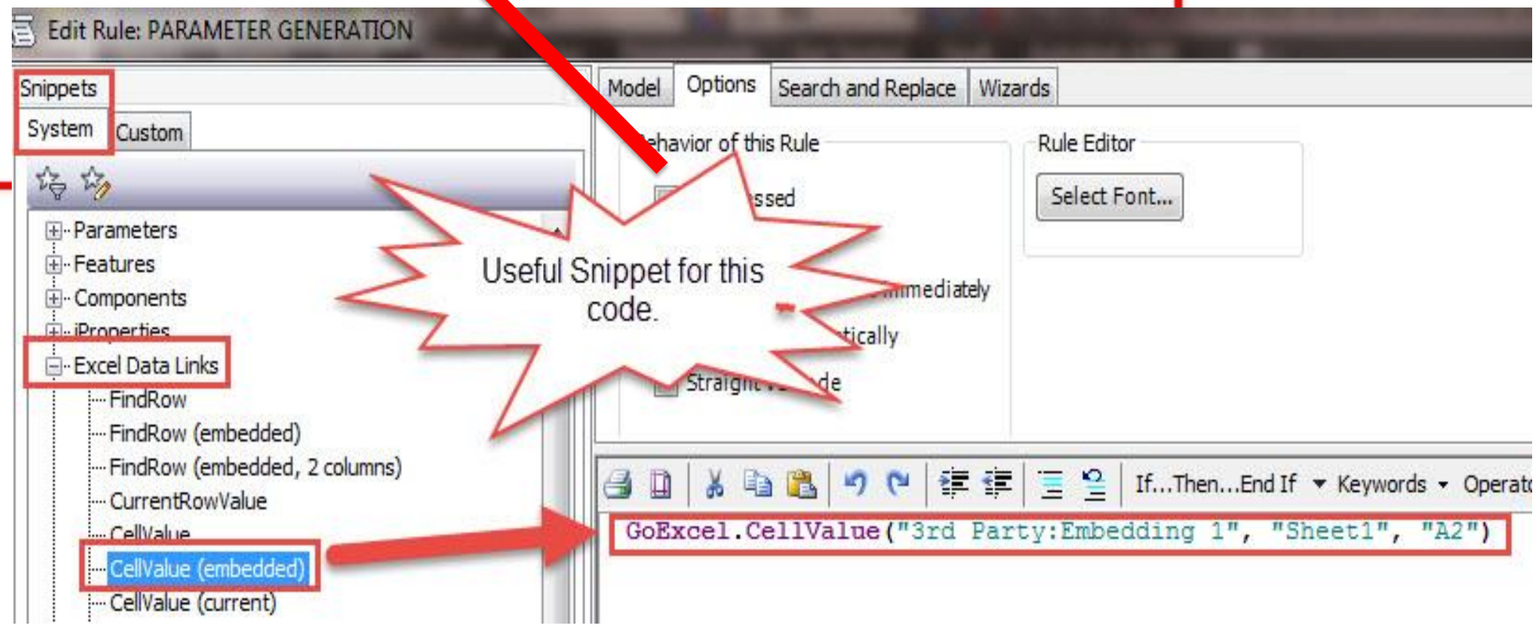
	H	I	J	
1				
2		*Impeller Data*		
3	Units are in Millimeters	Shroud Profile		
4		z	r	
5		156.860	138.011	1
6		148.011	138.369	2
7		139.225	139.449	3
8		130.543	141.232	4
9		122.032	143.673	5
10		113.700	146.667	6
11		105.524	150.111	7
12		97.495	153.848	8
13		89.710	157.655	9
14		81.986	161.955	10
15		75.494	167.965	11
16		69.001	174.158	12
17		62.735	180.589	13
18		57.195	186.929	14
19		52.745	192.761	15
20		48.872	198.633	16
21		45.341	204.955	17
22		42.161	211.851	18
23		39.443	219.306	19
24		37.711	227.366	20
25		36.568	235.869	21
26		35.415	244.691	22
27		34.221	253.822	23
28		32.989	263.246	24
29		31.717	272.999	25
30		30.404	283.032	26
31		29.045	293.370	27
32		27.645	304.013	28
33		26.202	314.960	29
34	Total # of Points =	29		

Programming Challenge #2 Code

```
'----------DEFINE COUNTER VALUE DEFINITION FOR THE "Z & R" DATA POINTS
Dim B As Integer
'----------DEFINE COUNTER VALUE FOR USER PARAMETER
Dim Y As Integer
Y = 1
'----------POPULATE SHROUD PROFILE "Z-VALUES"
   For B = 5 To 33
   Parameter("Z"&Y) = GoExcel.CellValue("3rd Party:Embedding 1", "Sheet1", "I"&B)
'----------POPULATE SHROUD PROFILE "R-VALUES"
   Parameter("R"&Y) = GoExcel.CellValue("3rd Party:Embedding 1", "Sheet1", "J"&B)

'----------COUNT UP TO NEXT DATA POINT
   Y = Y + 1
   Next
'----------RESET VALUES
Y = 1
```

Match your "3rd Party Embedding" file name



Useful Snippet for this code.

“Do While” Loops, Spread Sheets & Forms

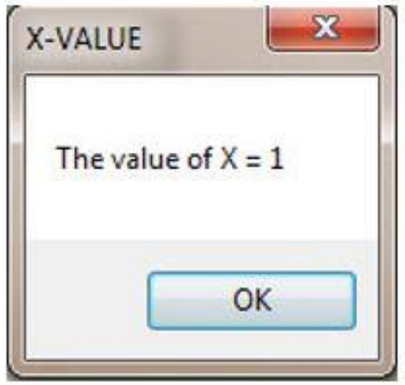
“Do While Loops”

What is a “Do While Loop”?

- The “Do While Loop” provides a way to continue iterating through your code, while one or more conditions are true.

For Example: A “Do While Loop” can have one or more conditions in its expression. In the below example, there is one condition that continues iterating while the Variable “X” is from 1 to less than or equal to 5. The below code will launch a message box stating what the current X value is, as long as the condition of X is from 1 to less than or equal to 5 is met.

```
Dim X As Integer
X = 1
Do While X <= 5
   MessageBox.Show("The value of X = " &X, "X-VALUE")
   X = X+1
Loop
```



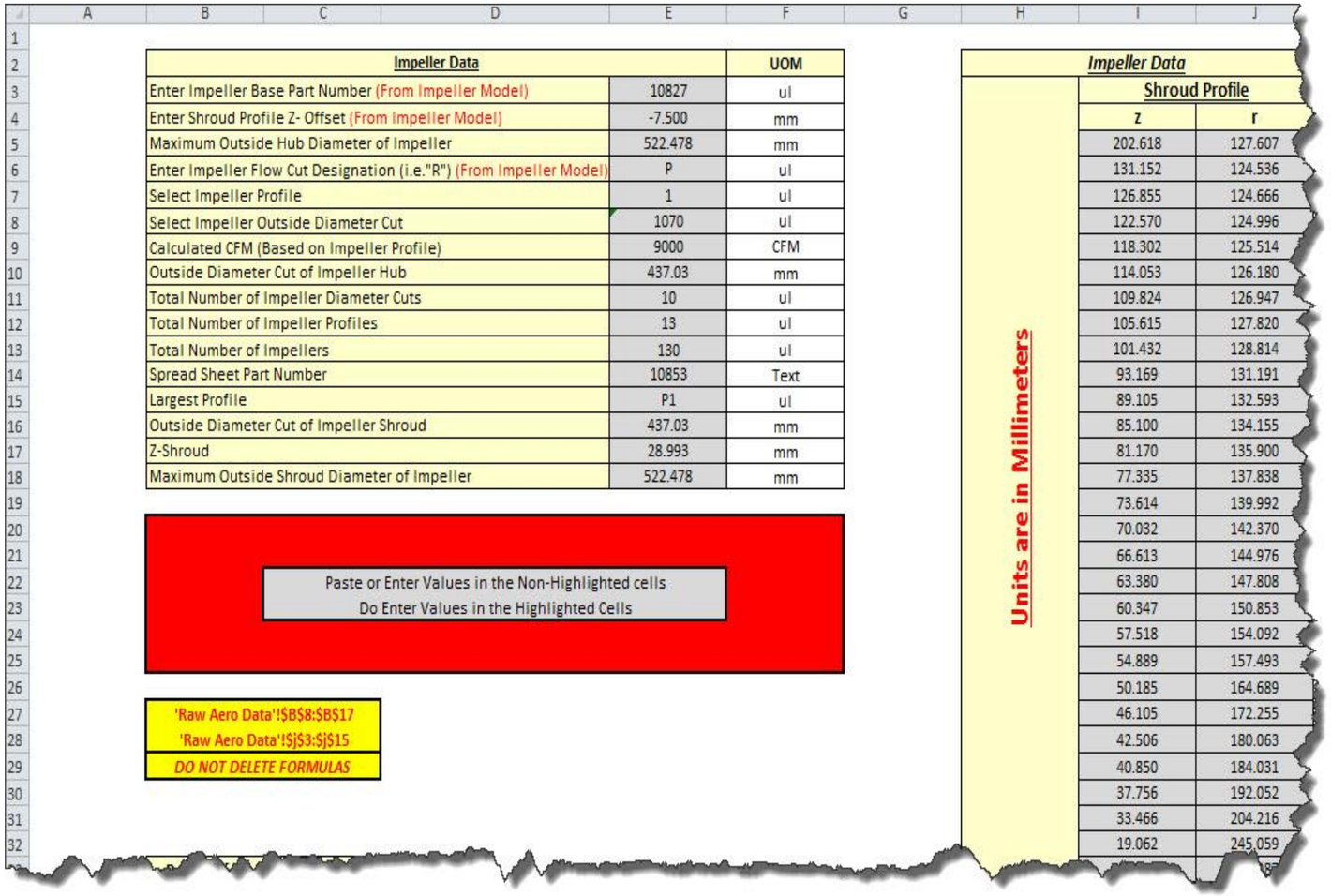
- The downside to this type of loop is that it can easily lead to infinite loops.

- My suggestion would be to make sure you save your file before executing your rule, or you may end up killing your session before a save and chance losing a lot of work.

iLogic and Spread Sheets

Why Use Spread Sheets?

- Common practice to store design data and to use spreadsheets to calculate outputs.
- Data can be read from Excel
- Data can be pushed to Excel
- Excel can process data by using various functions within the spread sheet.
- Allows data inputs and outputs to be entered and read in a consistent manner.
- You can read and input data from any sheet within the spreadsheet.
- Spread Sheet can be internal or external to the inventor file



Impeller Data		UOM
Enter Impeller Base Part Number (From Impeller Model)	10827	ul
Enter Shroud Profile Z- Offset (From Impeller Model)	-7.500	mm
Maximum Outside Hub Diameter of Impeller	522.478	mm
Enter Impeller Flow Cut Designation (i.e."R") (From Impeller Model)	P	ul
Select Impeller Profile	1	ul
Select Impeller Outside Diameter Cut	1070	ul
Calculated CFM (Based on Impeller Profile)	9000	CFM
Outside Diameter Cut of Impeller Hub	437.03	mm
Total Number of Impeller Diameter Cuts	10	ul
Total Number of Impeller Profiles	13	ul
Total Number of Impellers	130	ul
Spread Sheet Part Number	10853	Text
Largest Profile	P1	ul
Outside Diameter Cut of Impeller Shroud	437.03	mm
Z-Shroud	28.993	mm
Maximum Outside Shroud Diameter of Impeller	522.478	mm

Paste or Enter Values in the Non-Highlighted cells
Do Enter Values in the Highlighted Cells

'Raw Aero Data'!$B$8:$B$17
'Raw Aero Data'!$J$3:$J$15
DO NOT DELETE FORMULAS

Impeller Data

Units are in Millimeters

Shroud Profile	
z	r
202.618	127.607
131.152	124.536
126.855	124.666
122.570	124.996
118.302	125.514
114.053	126.180
109.824	126.947
105.615	127.820
101.432	128.814
93.169	131.191
89.105	132.593
85.100	134.155
81.170	135.900
77.335	137.838
73.614	139.992
70.032	142.370
66.613	144.976
63.380	147.808
60.347	150.853
57.518	154.092
54.889	157.493
50.185	164.689
46.105	172.255
42.506	180.063
40.850	184.031
37.756	192.052
33.466	204.216
19.062	245.059

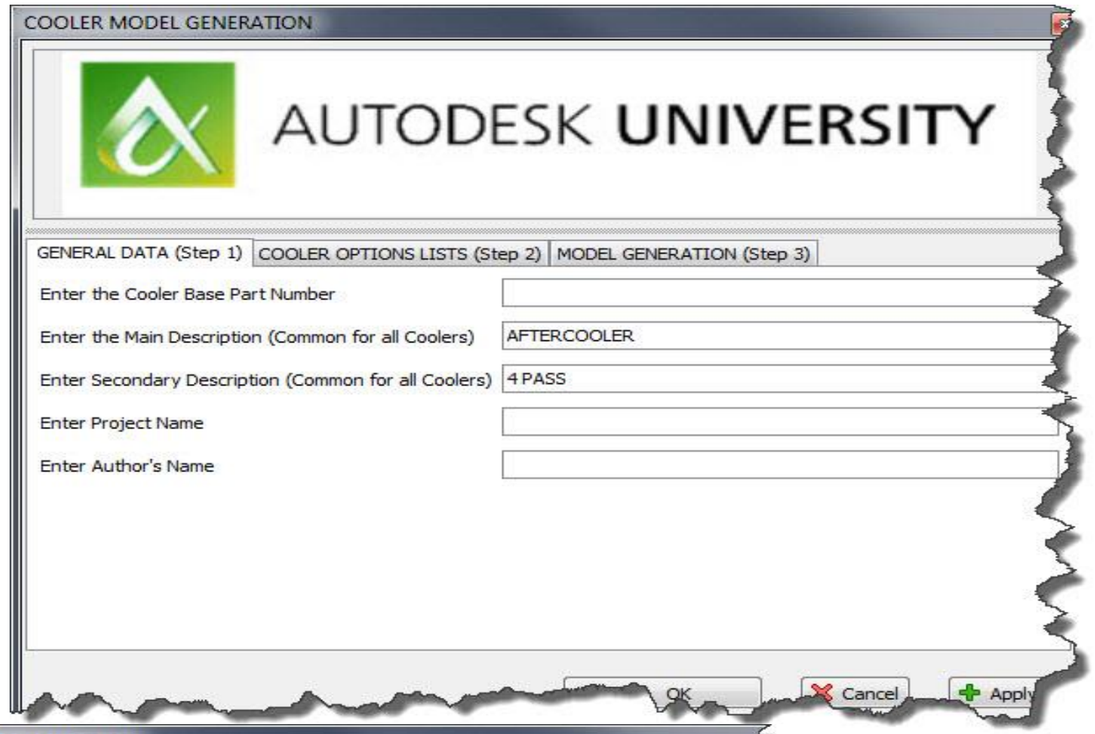
iLogic Forms

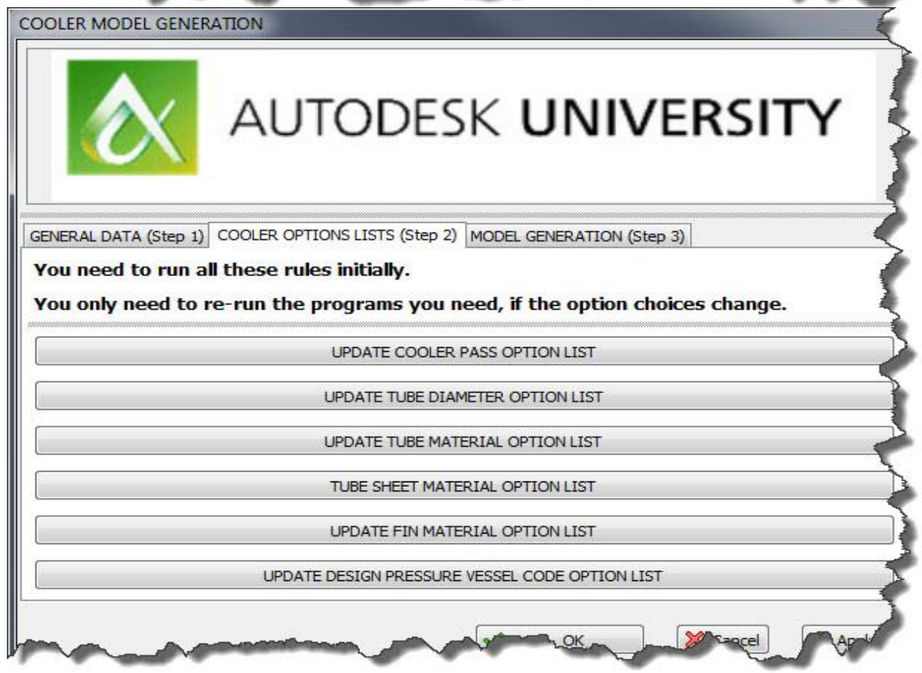
Why Use Forms?

- Forms are a great tool to make it easier for the user to work on the Inventor file.
- User interface

The form can assist the user in many ways

- Running the rules
- Entering data
- Pictures can be added to illustrate clarity to what data needs to be entered and why
- Efficiency





Programming Challenge #3

The Purpose of this Exercise is to show you how to:

- Write a program to generate multiple Impellers with different Profile Flow Cuts and Impeller Diameter cuts.
- Use “Do While Loops”
- Utilize the embedded Excel file to make the iLogic code simpler, by reading data from an output table. The outputs are provided based on inputs from the program into the Excel file.

“Programing Challenge 3 - PD20790-L.docx” located in:

C:\Datasets\Lab02 San Polo 3403\PD20790-L\Programming Challenges

20-25 Minutes

Programming Challenge #3 Code

```
Do While J<=L    ' 14<=16   (VALUE IS PUSHED TO EXCEL TO CALCULATE DATA CELL TO READ FROM)
'----------COLLECT EXCEL COLUMN LETTER FOR DIAMETER CUT
    COLLECT = GoExcel.CellValue("3rd Party:Embedding 1", "Sheet1", "N68")
'----------COLLECT IMPELLER CUT PERCENTAGE DIAMETER FROM EXCEL
    IC = GoExcel.CellValue("3rd Party:Embedding 1", "Sheet1", COLLECT & "72")
'----------PUSH IMPELLER CUT PERCENTAGE DIAMETER TO EXCEL FORM
    GoExcel.CellValue("3rd Party:Embedding 1", "Sheet1", "E8") = IC
'----------COLLECT IMPELLER CUT DIAMETER VALUE BASED ON PERCENTAGE VALUE
    SOD = Round(GoExcel.CellValue("3rd Party:Embedding 1", "Sheet1", "E10"),3)
'----------DO WHILE LOOP FOR IMPELLER FLOW CUTS
    Do While Z <= NP    '1<=5
'----------PUSH PROFLE CUT NUMBER TO EXCEL
        GoExcel.CellValue("3rd Party:Embedding 1", "Sheet1", "E7") = Z
'----------READ ALL PROFILE CUT VALUES FROM EXCEL SPREAD SHEET
        iLogicVb.RunRule("EXCEL")

'----------UPDATE DOCUMENT AND PARAMETERS
        iLogicVb.RunRule("UPDATE")
'----------IMPELLER CUT CHECK TO SEE IF IT IS LARGEST
        iLogicVb.RunRule("IMPELLER CUT")
'----------UPDATE DOCUMENT AND PARAMETERS
        iLogicVb.RunRule("UPDATE")
'----------NEW PROFILE DESIGNATION
        K = PD & Z
'----------PROFILE CUT NAME i.e. "AU1-1050"
        P=K+"-"+IC
'----------PN DESIGNATION i.e. "11111-AU1-1050"
        O=IPN+"-"+P
'----------THIS IS FILE NAME GENERATION i.e. "11111-AU1-1050.ipt"
        N=O+".ipt"

'----------UPDATE PART
        iLogicVb.UpdateWhenDone = True
        iLogicVb.RunRule("UPDATE")
```

```
'----------SAVE MODEL TO A SPECIFIC FOLDER
        FileToSave = ThisDoc.Path & "\"+IPN+"\" + N
        ThisDoc.Document.SaveAs(FileToSave, True)
'----------COUNT UP THE PROFILE NUMBER TO NEXT ONE
        Z=Z+1
        CIE = Z
'----------NEW PROFILE DESIGNATION
        K = PD & Z
'----------PROFILE DESIGNATION PARAMETER
        CI = K
'----------UPDATE PARAMETERS AND DOCUMENT
        iLogicVb.RunRule("UPDATE")
'----------END OF DO WHILE LOOP FOR IMPELLER FLOW CUTS
    Loop
'----------RESET PROFILE DESIGNATION COUNTER
    Z = 1
    CIE = 1
'----------RESET PROFILE DESIGNATION
    K = PD & Z
'----------RESET PROFILE DESIGNATION PARAMETER
    CI = K
'----------GO TO NEXT DIAMETER CUT VALUES
    J = J+1
'----------PUSH DIAMETER CUT VALUE TO SPREAD SHEET
    GoExcel.CellValue("3rd Party:Embedding 1", "Sheet1", "M68") = J
'----------UPDATE MODEL
    iLogicVb.RunRule("UPDATE")
'----------END OF WHILE LOOP FOR DIAMETER CUTS
Loop
```

Programming Challenge #4

The Purpose of this Exercise is to show you how to:

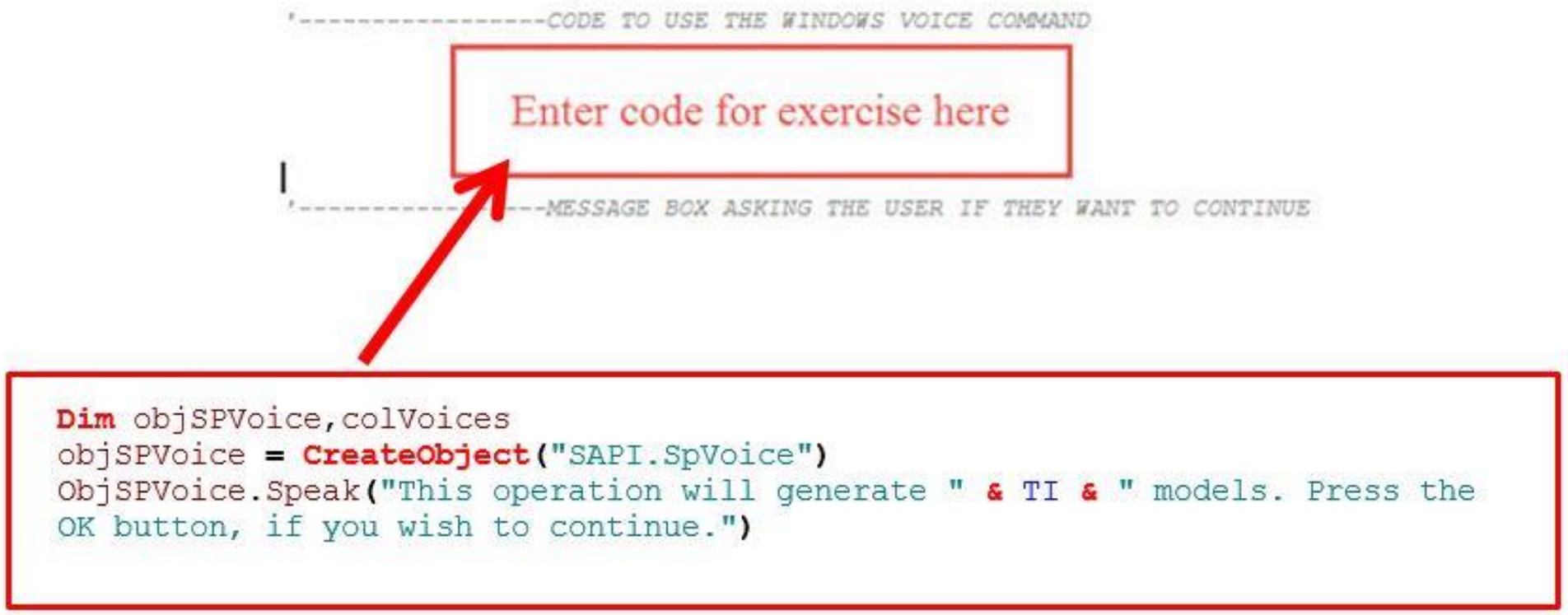
- Create a User interface to run the program
- Use the User interface to visually watch the program run
- Create simple code, for fun, to send the user a verbal message to verify if they want to run the program.

“Programing Challenge 4 - PD20790-L.docx” located in:

C:\Datasets\Lab02 San Polo 3403\PD20790-L\Programming Challenges

5-10 Minutes

Programming Challenge #4 Code



Conclusion

There are a number of ways to write code to automate your design process. I have shown you one of many. Hopefully this has started to get the thought process churning to think of ways that you can automate your design process.

Follow these Best Practices, when writing iLogic Code:

- Use comments in your code to make it easier to understand what your code is doing. This will help you and others down the road to understand how it works.
- Don’t overdo it by overcomplicating your rule. Sometimes more rules are better than putting them all into one.
- Consider making your rules so they can be reused in other projects. Why reinvent the wheel.
- When writing code it is always good to be consistent in your methods.
- Did I mention to use comments?

Conclusion (Con’t)

Autodesk Websites / Forums:

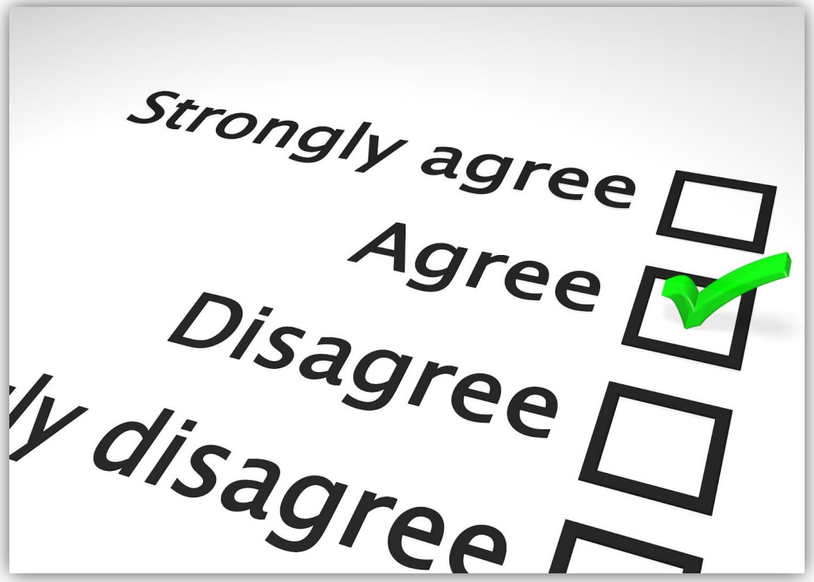
Autodesk Community Forums: https://forums.autodesk.com/
Autodesk Exchange Apps: https://apps.exchange.autodesk.com/en

Inventor Blogs:

From the Trenches with Autodesk Inventor (Curtis Waguespack):
http://inventortrenches.blogspot.com/

How did I do?

- Your class feedback is critical. Fill out a **class survey** now.
- Use the AU mobile app or fill out a class survey online.
- Give feedback after each session.
- AU speakers will get feedback in real-time.
- **Your feedback results in better classes and a better AU experience.**



Questions?