

Exercise 16-8

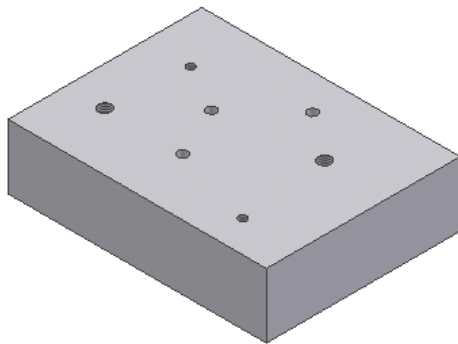
Exporting Data to an Excel Spreadsheet

File Name: New Visual Basic project

Estimated Time: 30 minutes

In this project, we will export hole data from a part into an Excel spreadsheet.

1.



Create a block in Inventor and place several different holes in it.

We will extract the hole information and place it into an Excel spreadsheet.

Excel needs to be open in order for this code to work.

Inventor should be open with the part you wish to use for this code to work.

2.



Start Visual Basic.

Go to **File**→**New Project**.

Excerpt from *Autodesk Inventor R8 Intermediate Level: Mastering the Rubicon*

by Elise Moss

Available from SDC Publications www.sdccad.com

3. Select Standard.EXE.



Press **OK**.

4. Right click on the tool panel.



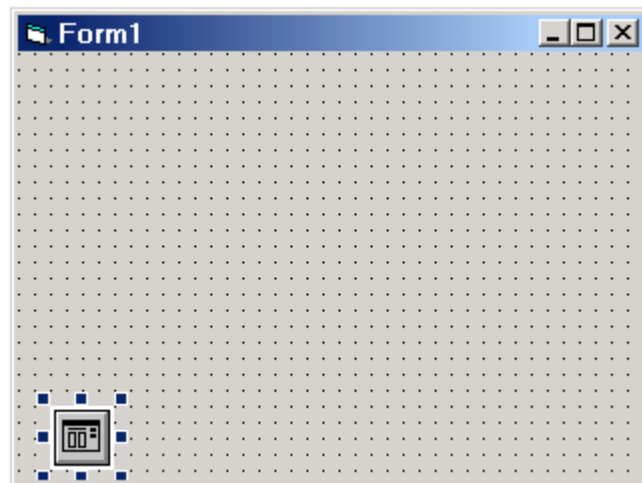
Select **Components**.

5. Enable the Common Dialog Control.



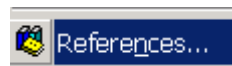
Press **OK**.

- 6.



Add the Common Dialog control to your form.

7. Go to **Project**→**References**.



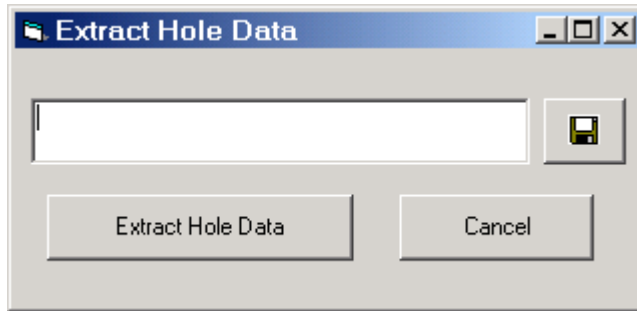
8. Enable the **Microsoft Excel Object Library**.



9. Enable the **Autodesk Inventor Object Library**.



10.



Create the user form using three command buttons and a text box.

Command Buttons:

Name – cmdBrowse, Style – Graphical, Picture – save.bmp

Name – cmdExtractData, Caption – Extract Hole Data

Name – cmdCancel, Caption – Cancel

Text Box –

Name – txtFileName, Caption – {blank}

11.

```
Private Sub CmdCancel_Click()  
    End  
End Sub
```

Add the code for the Cancel button.

12.

```
Public FileN As String
```

Define the file name as a public variable as it will be passed from the Browse command subroutine to the Extract Hole Data subroutine.

```
13. Private Sub cmdBrowse_Click()  
    CommonDialog1.Filter = "Excel Files (*.xls)|*.xls| All Files (*.*)|*.*"  
    CommonDialog1.FilterIndex = 1  
    CommonDialog1.DialogTitle = "Extract Data To:"  
    CommonDialog1.ShowSave  
    FileN = CommonDialog1.FileName  
    FrmHoleData.txtFileName.Text = FileN  
End Sub
```

In this case, we will be creating a new Excel file to be used to store our Hole Data.

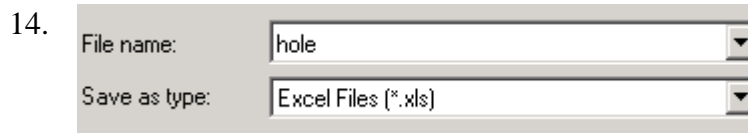
We define a filter to save as an Excel file.

Next, we enable the filter.

We assign a title to appear on the dialog box to assist the user.

Then, we bring up the Save As File dialog.

Finally, we display the path and file name in our text box.



Test the program we have so far and you see it works fine.

You may need to adjust the size of your text box to accommodate the file name and path.

Note that this does not actually create the file. This code only specifies the path and name for the file we will be creating.

As before, most of our code happens when you press the Extract Hole Data command button.

```
15. Private Sub cmdExtractData_Click()  
    'Inventor variables  
    Dim IV As Inventor.Application  
    Dim oDoc As Inventor.Document  
    Dim PartDef As PartComponentDefinition  
    Dim oHoles As Inventor.HoleFeatures  
    Dim oHole As Inventor.HoleFeature  
    Dim oHolePoint As Inventor.SketchPoint  
    Dim oHoleCoord As Inventor.Point2d  
    Dim oHoleTapInfo As Inventor.HoleTapInfo  
    Dim oHoleDepth As Inventor.PartFeatureExtent  
    Dim ToNr As Variant  
    Dim fileNum As Long  
    Dim HoleNr As Long  
    Dim oFileName As String  
    Dim HoleName As String  
    Dim XPoint As Double  
    Dim YPoint As Double  
    Dim HoleDia As String  
    Dim BoreDepth As String  
    Dim ThrdDepth As String
```

Define the Inventor variables.

```
16. 'Excel Variables  
  
    Dim Excel As Object  
    Dim excelWorkBook As Object  
    Dim excelSheet As Object  
  
    Dim excelUsed As Object  
    Dim iMaxRow As Integer  
    Dim iMaxCol As Integer
```

Define the Excel variables.

```
17. Set IV = GetObject(, "Inventor.Application")  
    Set oDoc = IV.ActiveDocument  
    Set PartDef = oDoc.ComponentDefinition  
    Set oHoles = PartDef.Features.HoleFeatures
```

The next section of code is used to set the Inventor variables to gather data from the open part file.

18.

```
On Error Resume Next
Set Excel = GetObject(, "Excel.Application")
If Err <> 0 Then
Set Excel = CreateObject("Excel.Application")
If Err <> 0 Then
MsgBox ("Could Not load Excel.")
End
End If
End If

On Error GoTo 0
```

Next we load Excel.

19.

```
'Maximize Excel
Excel.Visible = True
'Open new workbook
Set excelWorkBook = Excel.Workbooks.Add
'select sheet 1 of the workbook
Excel.Sheets("Sheet1").Select
Set excelSheet = Excel.ActiveWorkbook.Sheets("Sheet1")
```

We want to add a
workbook.

We set the active
worksheet to Sheet
1.

20.

```
Dim iColCount As Integer
Dim iRowCount As Integer
iRowCount = 2
iColCount = 7
```

We define where we are going to start on the sheet.

We want to start on Row 2 to add our data.

We will be using seven columns.

21. If oHoles.Count < 1 Then

```
MsgBox ("No holes exist in this part!")
```

Else

We check to see if there are indeed holes in the part. If there aren't any, a message box will appear advising the user.

22.

```
Else
  'write headers
  Worksheets ("Sheet1").Range ("A1").Value = "Nr"
  Worksheets ("Sheet1").Range ("B1").Value = "Name"
  Worksheets ("Sheet1").Range ("C1").Value = "X"
  Worksheets ("Sheet1").Range ("D1").Value = "Y"
  Worksheets ("Sheet1").Range ("E1").Value = "Dia"
  Worksheets ("Sheet1").Range ("F1").Value = "BoreDepth"
  Worksheets ("Sheet1").Range ("G1").Value = "ThreadDepth"
End If
```

Next, we write the headers on the first row of the worksheet.

23.

```
For Each oHole In oHoles
  With oHole
    HoleName = .Name
    Set oHoleDepth = .Extent
    ' if the extent is not through
    If Not oHoleDepth.Type = kThroughAllExtentObject Then

      ToNr = Split(oHoleDepth.Distance.Expression, "")
      BoreDepth = ToNr(0)
    Else
      BoreDepth = "Through"
    End If
    If Not .Tapped Then
      ToNr = Split(.HoleDiameter.Expression, "")
      HoleDia = ToNr(0)
      ThrdDepth = ""
    Else
      Set oHoleTapInfo = .TapInfo
      HoleDia = oHoleTapInfo.PitchDesignation
      If Not oHoleTapInfo.FullTapDepth Then
        Debug.Print oHoleTapInfo.ThreadDepth.Expression
        ToNr = Split(oHoleTapInfo.ThreadDepth.Expression, " ")
        ThrdDepth = ToNr(0)
      End If
    End If
  End If
  ' get coordinate data for holes
  For Each oHolePoint In .HoleCenterPoints
    Set oHoleCoord = oHolePoint.Geometry
    XPoint = oHoleCoord.X * 10
    YPoint = oHoleCoord.Y * 10
    HoleNr = HoleNr + 1
  End For
End For
```

The next section of code gathers information on each hole feature.

It may be helpful to you to use Watch to see what information is placed in the ToNr variable. We use this variable as a test to handle different hole data depending on whether it is tapped, if the hole is through or has a depth, etc.

The final section of code extracts the X and Y data for the hole centerpoint location on the sketch. This is in relation to the sketch origin.

24. ' write data to excel sheet

```
excelSheet.Cells(iRowCount, 1).Value = HoleNr  
excelSheet.Cells(iRowCount, 2).Value = HoleName  
excelSheet.Cells(iRowCount, 3).Value = XPoint  
excelSheet.Cells(iRowCount, 4).Value = YPoint  
excelSheet.Cells(iRowCount, 5).Value = HoleDia  
excelSheet.Cells(iRowCount, 6).Value = BoreDepth  
excelSheet.Cells(iRowCount, 7).Value = ThrdDepth
```

Next we write the data to the spreadsheet. Note that we use the iRowCount variable to locate each cell to place the data.

25. iRowCount = iRowCount + 1

We increment the row count so we can drop down a row when we move on the next hole.

26.

```
Next  
End With  
Next  
  
End If
```

We close the loops and the if holes exist statement.

Excerpt from *Autodesk Inventor R8 Intermediate Level: Mastering the Rubicon*

by Elise Moss

Available from SDC Publications www.sdccad.com

27.

```
'save the file  
excelWorkBook.SaveAs FileN  
  
' close the dialog  
Unload Me
```

We save the Excel spreadsheet.

We unload the dialog.

28. Try running the program and see if it works for you.

A sample version is available for download from the publisher's website.

The entire code follows:

```
Public FileN As String
```

```
-----
```

```
Private Sub cmdBrowse_Click()
```

```
    CommonDialog1.Filter = "Excel Files (*.xls)|*.xls| All Files (*.*)|*.*"
```

```
    CommonDialog1.FilterIndex = 1
```

```
    CommonDialog1.DialogTitle = "Extract Data To:"
```

```
    CommonDialog1.ShowSave
```

```
    FileN = CommonDialog1.FileName
```

```
    FrmHoleData.txtFileName.Text = FileN
```

```
End Sub
```

```
-----
```

```
Private Sub CmdCancel_Click()
```

```
    End
```

```
End Sub
```

```
-----
```

```
Private Sub cmdExtractData_Click()
```

```
    'Inventor variables
```

```
    Dim IV As Inventor.Application
```

```
    Dim oDoc As Inventor.Document
```

```
    Dim PartDef As PartComponentDefinition
```

```
    Dim oHoles As Inventor.HoleFeatures
```

```
    Dim oHole As Inventor.HoleFeature
```

```
    Dim oHolePoint As Inventor.SketchPoint
```

```
Dim oHoleCoord As Inventor.Point2d
```

```
Dim oHoleTapInfo As Inventor.HoleTapInfo
```

```
Dim oHoleDepth As Inventor.PartFeatureExtent
```

```
Dim ToNr As Variant
```

```
Dim fileNum As Long
```

```
Dim HoleNr As Long
```

```
Dim oFileName As String
```

```
Dim HoleName As String
```

```
Dim XPoint As Double
```

```
Dim YPoint As Double
```

```
Dim HoleDia As String
```

```
Dim BoreDepth As String
```

```
Dim ThrdDepth As String
```

```
'Excel Variables
```

```
Dim Excel As Object
```

```
Dim excelWorkBook As Object
```

```
Dim excelSheet As Object
```

```
Dim excelUsed As Object
```

```
Dim iMaxRow As Integer
```

```
Dim iMaxCol As Integer
```

```
Set IV = GetObject(, "Inventor.Application")
```

```
Set oDoc = IV.ActiveDocument
```

```
Set PartDef = oDoc.ComponentDefinition
```

```
Set oHoles = PartDef.Features.HoleFeatures
```

```
On Error Resume Next
```

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

```
If Err <> 0 Then
```

```
Set Excel = CreateObject("Excel.Application")
```

```
If Err <> 0 Then
```

```
MsgBox ("Could Not load Excel.")
```

```
End
```

```
End If
```

```
End If
```

```
On Error GoTo 0
```

```
'Maximize Excel
```

```
Excel.Visible = True
```

```
'Open new workbook
```

```
Set excelWorkBook = Excel.Workbooks.Add
```

```
'select sheet 1 of the workbook
```

```
Excel.Sheets("Sheet1").Select
```

```
Set excelSheet = Excel.ActiveWorkbook.Sheets("Sheet1")
```

```
Dim iColCount As Integer
```

```
Dim iRowCount As Integer
```

```
iRowCount = 2
```

```
iColCount = 7
```

```
If oHoles.Count < 1 Then

    MsgBox ("No holes exist in this part!")

Else

    'write headers

    excelSheet.Cells(1, 1).Value = "Nr"

    excelSheet.Cells(1, 2).Value = "Name"

    excelSheet.Cells(1, 3).Value = "X"

    excelSheet.Cells(1, 4).Value = "Y"

    excelSheet.Cells(1, 5).Value = "Dia"

    excelSheet.Cells(1, 6).Value = "BoreDepth"

    excelSheet.Cells(1, 7).Value = "ThreadDepth"

    excelSheet.Range("A1:G1").Font.Bold = True

    excelSheet.Range("A1:G1").Font.Underline = True

    'write data

    For Each oHole In oHoles

        With oHole

            HoleName = .Name

            Set oHoleDepth = .Extent

            ' if the extent is not through

            If Not oHoleDepth.Type = kThroughAllExtentObject Then

                ToNr = Split(oHoleDepth.Distance.Expression, "")

                BoreDepth = ToNr(0)

            Else

                BoreDepth = "Through"

            End If

            If Not .Tapped Then
```

```
ToNr = Split(.HoleDiameter.Expression, "")

HoleDia = ToNr(0)

ThrdDepth = ""

Else

Set oHoleTapInfo = .TapInfo

HoleDia = oHoleTapInfo.PitchDesignation

If Not oHoleTapInfo.FullTapDepth Then

    Debug.Print oHoleTapInfo.ThreadDepth.Expression

    ToNr = Split(oHoleTapInfo.ThreadDepth.Expression, " ")

    ThrdDepth = ToNr(0)

End If

End If

' get coordinate data for holes

For Each oHolePoint In .HoleCenterPoints

    Set oHoleCoord = oHolePoint.Geometry

    XPoint = oHoleCoord.X * 10

    YPoint = oHoleCoord.Y * 10

    HoleNr = HoleNr + 1

' write data to excel sheet

excelSheet.Cells(iRowCount, 1).Value = HoleNr

excelSheet.Cells(iRowCount, 2).Value = HoleName

excelSheet.Cells(iRowCount, 3).Value = XPoint

excelSheet.Cells(iRowCount, 4).Value = YPoint

excelSheet.Cells(iRowCount, 5).Value = HoleDia

excelSheet.Cells(iRowCount, 6).Value = BoreDepth

excelSheet.Cells(iRowCount, 7).Value = ThrdDepth

iRowCount = iRowCount + 1
```

Excerpt from *Autodesk Inventor R8 Intermediate Level: Mastering the Rubicon*

by Elise Moss

Available from SDC Publications www.sdccad.com

```
Next
End With
Next
End If
'save the file
excelWorkBook.SaveAs FileN

' close the dialog

Unload Me

End Sub
```

Excerpt from *Autodesk Inventor R8 Intermediate Level: Mastering the Rubicon*
by Elise Moss
Available from SDC Publications www.sdccad.com