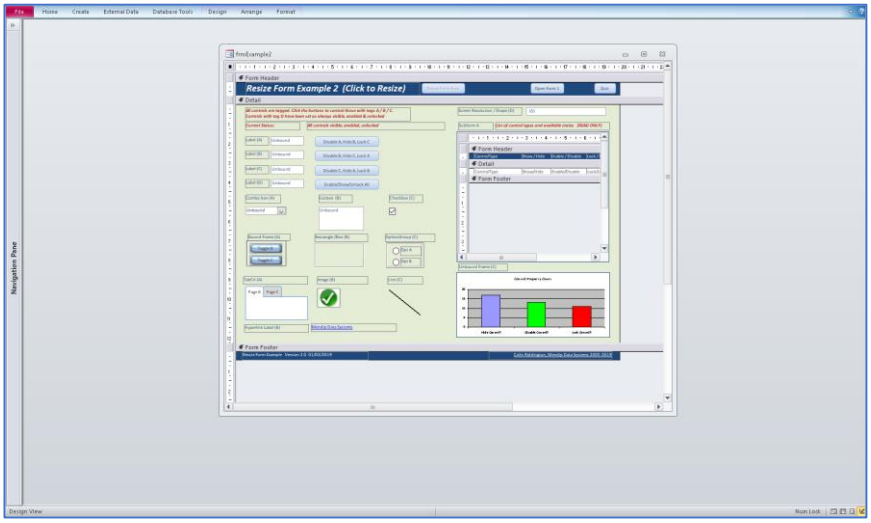
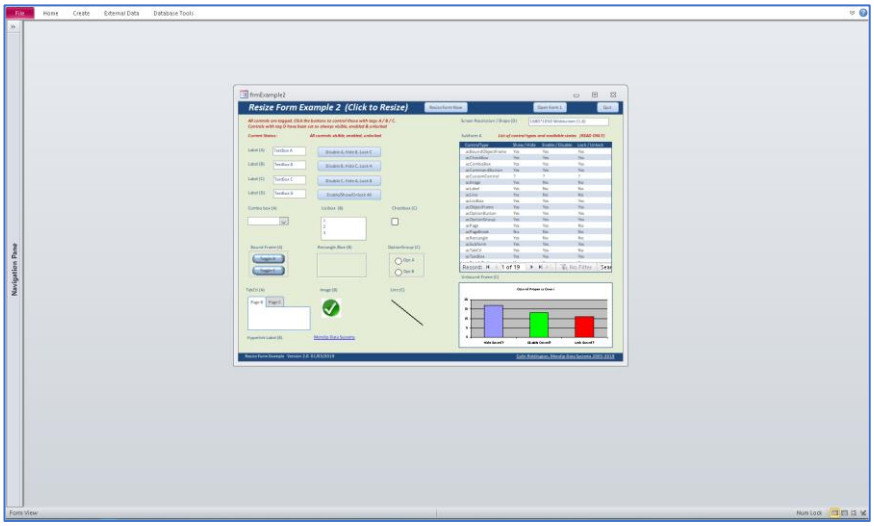
ResizeForm Me – A Tutorial in Automatic Form Resizing

All developers will be aware that forms designed for a specific screen size/resolution may look dreadful on a different monitor with a higher/lower resolution and /or screen size or shape (4:3, 16:9 or widescreen)

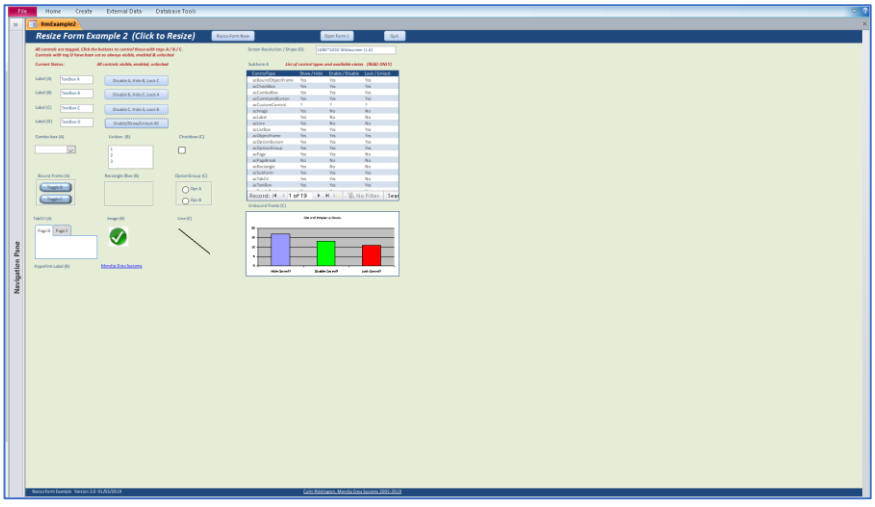
For example, this form was designed at a low resolution **(800*600)**



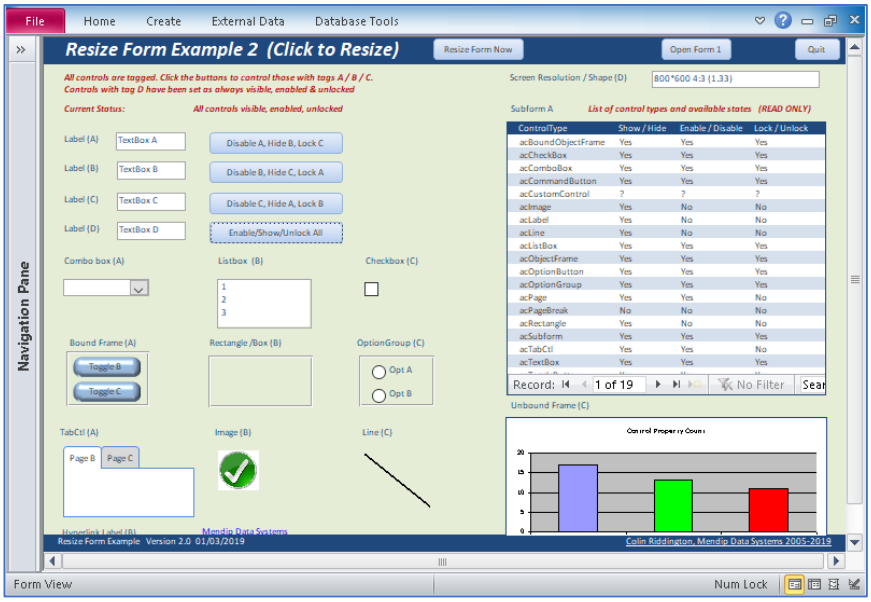
If it is viewed at a higher resolution **(1680*1020)**, it only fills part of the screen and each item is tiny
This is using overlapping windows



Or with **tabbed documents**, the screen is filled but the contents remain squashed in the top left corner

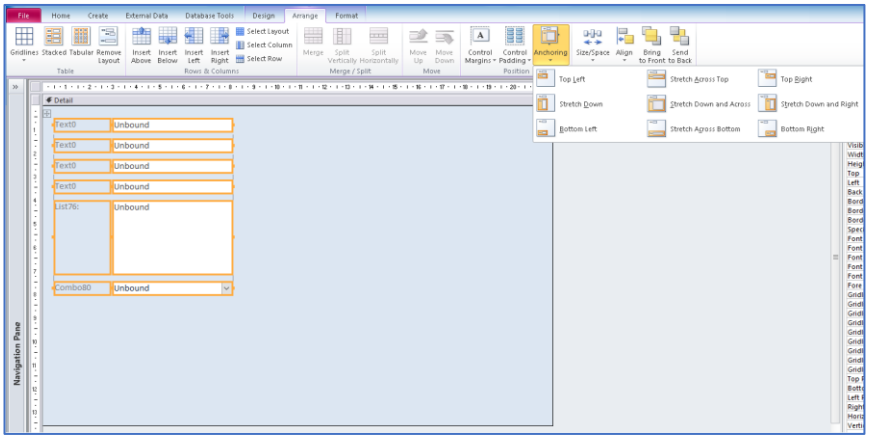


On a smaller screen or lower resolution, the opposite problem occurs with part of the form not shown unless the user scrolls in both directions



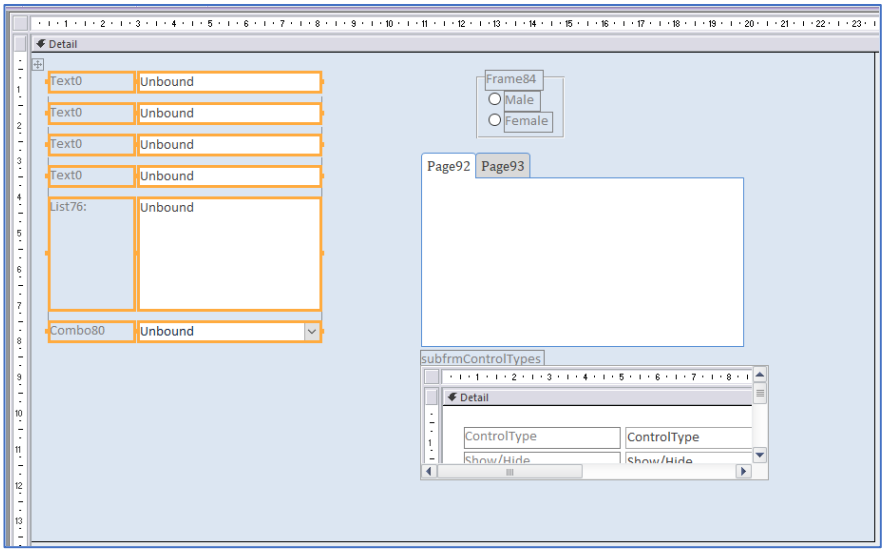
The effect is the same whether using overlapping windows or tabbed documents

Some developers get around this issue to some extent by using layout guides to group a number of controls together. For example, a stacked layout has been used for these controls.
All controls in the group are automatically made the same width

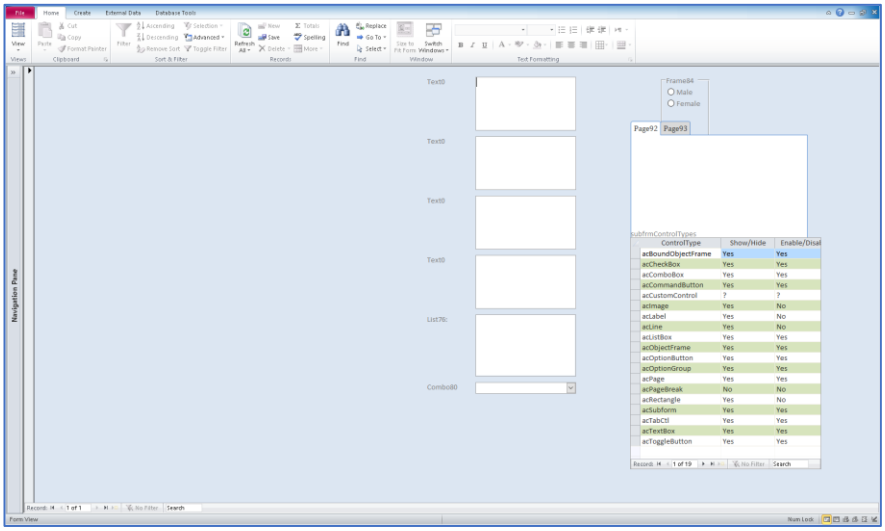


In addition anchoring can be used to adjust the position/size on the screen.
In this case **Stretch Down and Across** has been applied

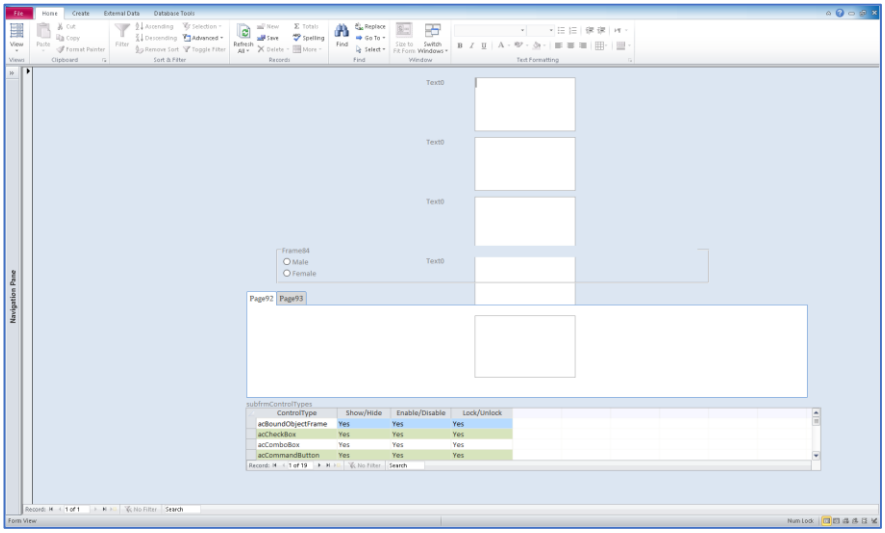
The results can be quite successful for simple form layouts. However, that isn't necessarily the case for more complex form designs. For example, extra controls have been added in design view:



Depending on the anchoring choices, this may result in this (with lots of empty space on the left)



Or another variation with various controls overlapping / obscured … etc



With planning, reasonable results can still be obtained.
However, users do not have full control over positioning and size of each control on the form

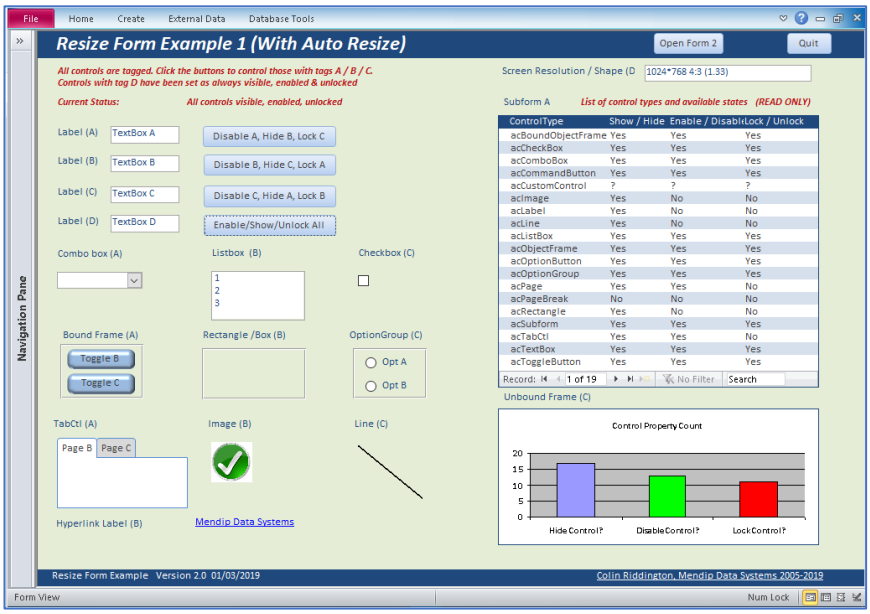
Form resizing code is designed to fix all such issues. Forms can be **automatically resized** for any **screen size & resolution** whilst allowing developers full control over the form layout and appearance

There are various examples available including commercial packages such as:

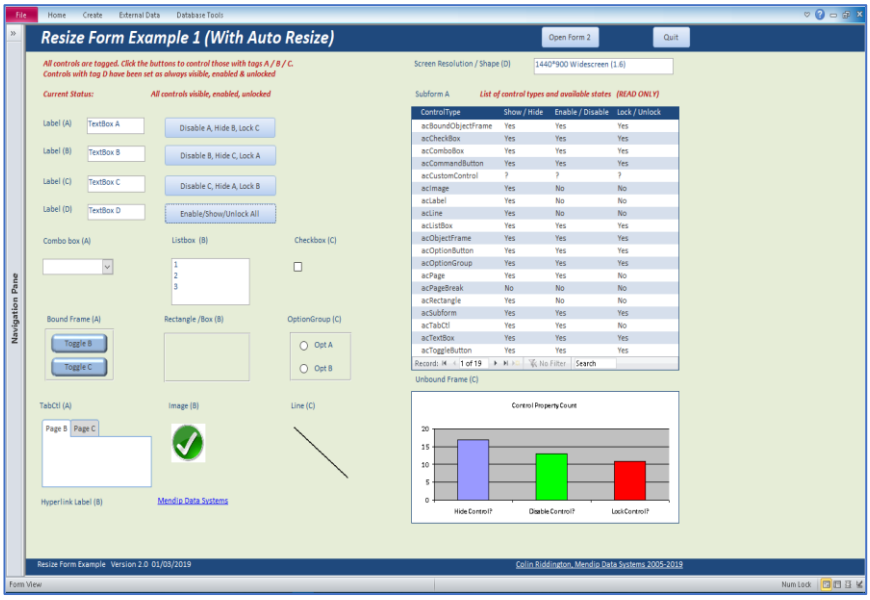
- **Shrinker Stretcher** available from http://www.peterssoftware.com/ss.htm
- **Total Access Components** from http://www.fmsinc.com/microsoftaccess/controls.html

However, for many years, I have used a modified version of **free open source code** by **Jamie Czernik** originally written in 2003. The original code is still available at http://jamieczernik.powweb.com/articles/resolution.html though various improvements have been added since by others including myself.

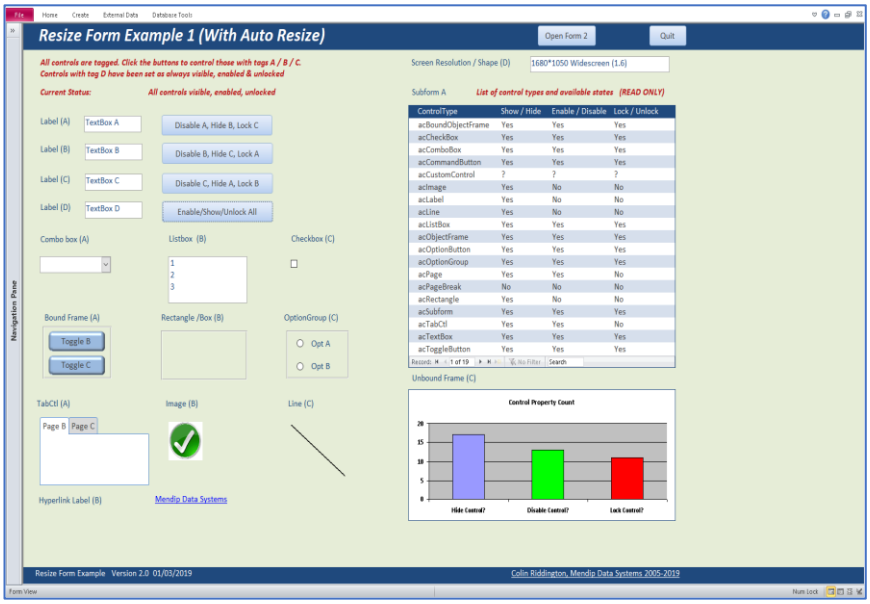
This is an **identical form** to that shown earlier in this article but with form resizing code added.
Here is the form viewed at a resolution of **1024*768**



And again at a resolution of **1440 *900**



And yet again at a resolution of **1680*1050**

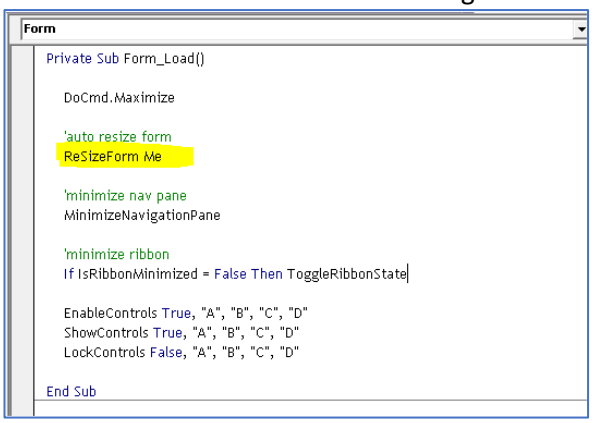


As you can see the entire form is shown whatever resolution is used as long as that is the same as or higher than the 'design resolution'. The code intelligently enlarges the form itself to fit the screen and moves each control by an appropriate amount to keep the form in proportion. It also enlarges each control proportionately together with the contents of that control.
The code works for each type of control including subforms

NOTE:

1. To allow for different form factors (screen shapes) i.e. 4:3, 16:9, widescreen some empty space may be left on the right of the screen when using a maximised form as above
2. The code has been extensively tested on a wide range of monitors and form factors such as 22 inch widescreen, 14 inch 16:9, 15 inch 4:3 and a 10 inch 16:9 tablet

So what code is neeed to work this magic? Just **one line** in the **Form_Load** event – **ResizeForm Me**



OK, I hear you say, surely that can't be all there is to it! Well not quite ….
Behind the scenes there is a lengthy module **modResizeForm** which contains all the code required to adjust the forms as required. This is included with the attached example application and has been updated to work in both 32-bit and 64-bit Access

If you import the module to your own applications and add the above line in the **Form_Load** events, all forms will be resized. However, the results will not at first be at all successful as the forms were not designed with resizing in mind

It is STRONGLY recommended that you start with new forms and plan for resizing from the start

Designing with the form resizer

As a general rule, you always need to develop using the lowest resolution that your users are likely to have.
Form resizing upwards (stretching) is much easier and has far fewer issues than resizing downwards (shrinking).

The declarations section of the **modResizeForm** module includes the following lines:

```
'----------------------------MODULE CONSTANTS & VARIABLES----------------------------
Private Const DESIGN_HORZRES As Long = 800  '<- CHANGE THE VALUE ABOVE TO THE RESOLUTION YOU DESIGNED YOUR FORM IN.
                                            '(e.g. 800 X 600 -> 800)

'Colin Riddington 16/02/2019 - ' the next item is currently not in use and can be disabled
Private Const DESIGN_VERTRES As Long = 600   '<- CHANGE THIS VALUE TO THE RESOLUTION YOU DESIGNED YOUR FORMS IN.
                                            '(e.g. 800 X 600 -> 600)

Private Const DESIGN_PIXELS As Long = 96      '<- CHANGE THE VALUE ABOVE TO THE DPI SETTING YOU DESIGNED YOUR FORM IN.
                                            '(If in doubt do not alter the lngDesignPixels setting.)
```

The first line is the default or baseline horizontal resolution – in this case 800
You can adjust this to any other suitable minimum value such as 1024 if you prefer

The next line is the baseline vertical resolution – 600
This can also be amended to suit, but in the current version of the code this is NOT used & can be disabled

The third line is the **pixels per inch** setting which is normally 96 for 100%
NOTE: if the screen is magnified to e.g. 125% this value becomes 120 dpi

Setting the default form size

The code currently specifies a 'base resolution' of ***800*600*** though that can be changed to any resolution that suits your needs. However I certainly don't develop in that resolution. My primary monitor is in fact ***1680*1050***.

It actually means that my base form size in design view is such that if I did use ***800*600***, it would fill the screen.
By experiment, I found that size should be ***20.5cm*12.5cm*** approximately

So my forms are designed in that size & will scale up to 'fill the screen' in any other higher resolution
More accurately it will fill the total height of the screen.

Due to different form factors ***(4:3, 5:4, 16:9, 16:10 etc)*** the form may be slightly less than the total monitor width.
Whilst I could stretch it to fit horizontally, there would be some distortion by doing so
Alternatively you could increase the default width of your forms in design view

I also design with default font = Calibri 7pt which after resizing becomes Calibri 11pt

I could change the base resolution to say ***1024*768*** and if I did so, the base form size would be proportionately larger ***(25.85*17.2 cm approx)*** & font size 9pt. However, it would scale up from that equally well.

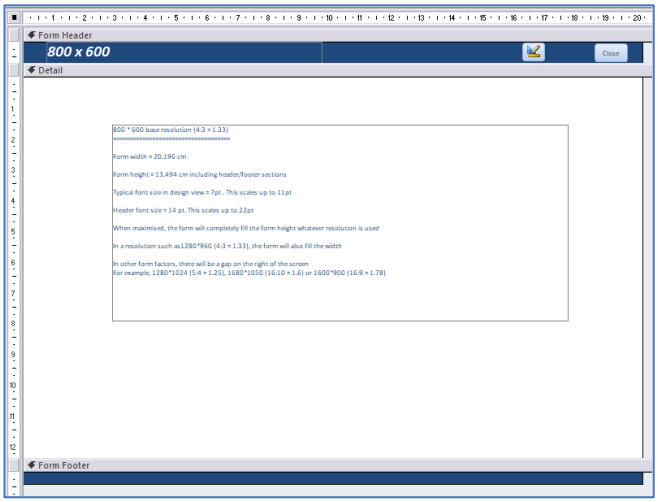
You may wish to adjust the default width if, for example, all users have monitors with widescreen form factor.

You may also need to adapt the specified sizes slightly if you normally do any of the following:

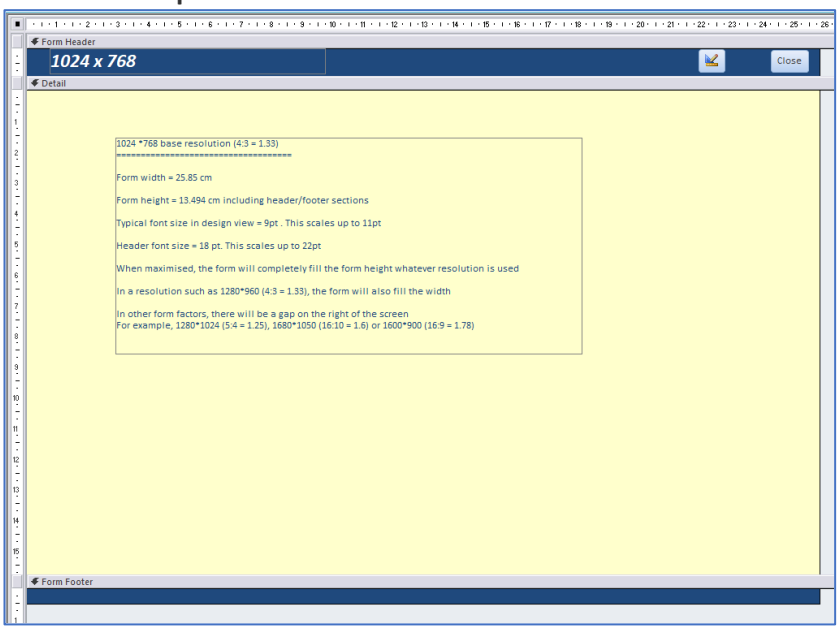
- Set ribbon and / or navigation pane maximised
- Design forms with no header / footer section
- Use tabbed documents instead of overlapping windows
- Use popup forms

The attached application includes two example template forms ***(800x600 & 1024x768)*** which may be useful to set maximum form sizes to ensure they fit properly in different screen resolutions.

800*600 template



1024*768 template



How Does the Code Work?

The **GetFactor** function calculates the multiplying factor based on screen size & resolution:

```
Public Function GetFactor() As Single

Dim sngFactorP As Single

On Error Resume Next

   If NewRes.DPI <> 0 Then
      sngFactorP = DESIGN_PIXELS / NewRes.DPI
   Else
      sngFactorP = 1 'error with dpi reported so assume 96 dpi
   End If
   '============================================================
   'modification by Jeff Blumsom 18/1/07
   Select Case FormFactor
      Case "4:3"
         GetFactor = (NewRes.Width / DESIGN_HORZRES) * sngFactorP
      Case "5:4"
         GetFactor = (NewRes.Width / DESIGN_HORZRES) * sngFactorP
      Case "Widescreen"
         GetFactor = ((4 / 3) * NewRes.Height / DESIGN_HORZRES) * sngFactorP
   End Select
   '============================================================

    'Debug.Print sngFactorP; GetFactor, FormFactor

End Function
```

The **Resize** procedure then uses that information to adjust the height and width of the form together with the size and position of each control on the form

```
Private Sub Resize(sngFactor As Single, frm As Access.Form)

Dim ctl As Access.control
On Error Resume Next
    'Resize height for each section:
    With frm
        .Width = .Width * sngFactor
       .Section(Access.acHeader).Height = .Section(Access.acHeader).Height * sngFactor
       .Section(Access.acDetail).Height = .Section(Access.acDetail).Height * sngFactor
       .Section(Access.acFooter).Height = .Section(Access.acFooter).Height * sngFactor
    End With
    'Resize and locate each control:
    For Each ctl In frm.Controls
        If (ctl.ControlType <> Access.acPage) Then 'ignore pages in TAB controls
            With ctl
                .Height = .Height * sngFactor
                .Left = .Left * sngFactor
                .Top = .Top * sngFactor
                .Width = .Width * sngFactor

                'next line moved back to individual control properties below
               ' .FontSize = .FontSize * sngFactor
               ' Enhancement by Myke Myers ------------------------------------->
                'Fix certain combo box, list box and tab control properties:
                Select Case .ControlType
                    Case acLabel, acCommandButton, acTextBox, acToggleButton
                        .FontSize = .FontSize * sngFactor
                    Case acListBox
                        .FontSize = .FontSize * sngFactor
                        .ColumnWidths = AdjustColumnWidths(.ColumnWidths, sngFactor)
                    Case acComboBox
                        .FontSize = .FontSize * sngFactor
                        .ColumnWidths = AdjustColumnWidths(.ColumnWidths, sngFactor)
                        .ListWidth = .ListWidth * sngFactor
                    Case acTabCtl
                        .FontSize = .FontSize * sngFactor
                        .TabFixedWidth = .TabFixedWidth * sngFactor
                        .TabFixedHeight = .TabFixedHeight * sngFactor
                    Case Else  'no other code here
                        'acRectangle, acCheckBox, acImage, acLine, acPageBreak, acSubform
                        'acOptionButton, acOptionGroup, acObjectFrame, acBoundObjectFrame
                End Select
                '-------------------------------------> End enhancement by Myke Myers
            End With
        End If
    Next ctl
End Sub
```

Certain controls receive 'special treatment' – **list boxes, combo boxes and tab controls**
Tab control pages are excluded as are the **contents** of **subforms** (the subform **container** is resized automatically)

As previously stated, the code line **ResizeForm Me** needs to be added to each form being resized.
This code line means the **ResizeForm** procedure is applied to the loaded form (**Me**)

To scale up the **subform control positions / sizes**, add the line **ResizeForm Me** to the **Form_Load** event of the **subform**. Alternatively, add a line like **ReSizeForm subFormName.Form** to the **Form_Load** event of the **main form**

```
Public Sub ReSizeForm(frm As Access.Form)

On Error GoTo Err_Handler

Dim rectWindow As tRect
Dim lngWidth As Long
Dim lngHeight As Long
Dim sngFactor As Single
On Error Resume Next
    GetScreenResolution
    sngFactor = GetFactor 'local function
    If NewRes.Width <> DESIGN_HORZRES Then 'no resize necessary
        Resize sngFactor, frm 'local procedure

'=============================================================================================
'modification by Jeff Blumsom 18/1/07
'the following code controls the positioning of pop-up forms
'but only if the form tag is null.  This allows more control where it causes a problem
'Colin Riddington 16/02/2019 - I've NEVER used this section - suggest not applying resize to popup forms
        If WM_apiIsZoomed(frm.hwnd) = 0 Then            'Don't change window settings for max'd form.
            Access.DoCmd.RunCommand acCmdAppMaximize    'Max Access Window
            Call WM_apiGetWindowRect(frm.hwnd, rectWindow)
            With rectWindow
                lngWidth = .Right - .Left
                lngHeight = .Bottom - .Top
            End With
            If Nz(frm.Tag, 1) <> 1 Then
                Call WM_apiMoveWindow(frm.hwnd, ((NewRes.Width - _
                (sngFactor * lngWidth)) / 2) - GetLeftOffset, _
                ((NewRes.Height - (sngFactor * lngHeight)) / 2) - GetTopOffset, _
                lngWidth * sngFactor, lngHeight * sngFactor, 1)
            End If
        End If
'=============================================================================================
    End If

Exit_Handler:
    Exit Sub

Err_Handler:
    MsgBox "Error " & Err.Number & " " & Err.description & " in ReSizeForm procedure:", vbExclamation, "Program error"
    Resume Exit_Handler

End Sub
```

The code first checks whether resizing is required then uses the **Resize** procedure above

Using the example applications

The attached application **ResizeFormExample_v2.0.accdb** has been designed for use with **overlapping windows**. It contains two almost identical versions of the form shown above:

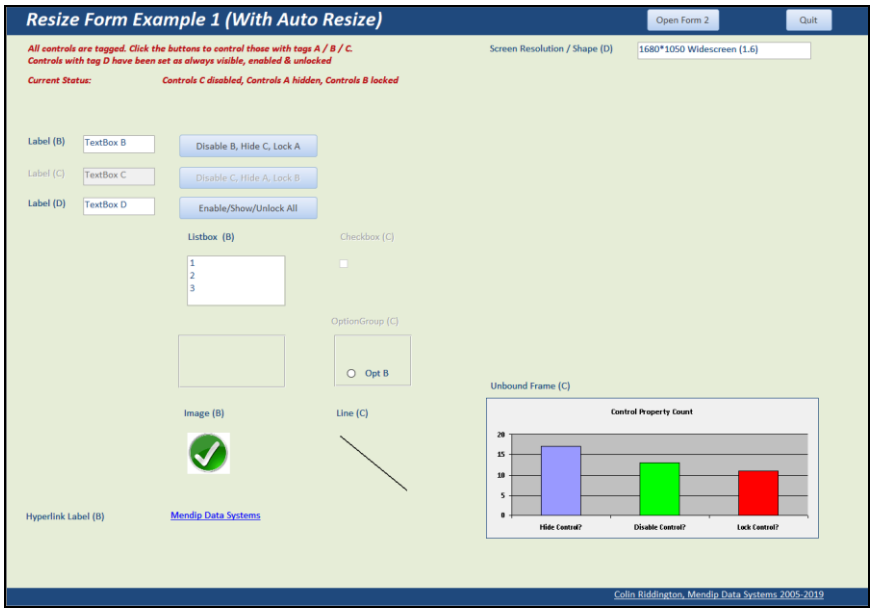
- **frmExample1** – automatically resized on form load
- **frmExample2** – not resized on form load BUT can be resized by clicking a button in the form header

Each form also includes a subform

NOTE: Issues will occur if this form is changed to **tabbed documents**. See the next section for details

The forms used in this example were originally designed to demonstrate the use of the **Tag** property to make **groups** of **controls visible/hidden, enabled/disabled, locked/unlocked**.

In this screenshot, controls with tag A have been hidden, tag B controls locked and tag C controls disabled.



All the code used in that feature is contained in the module **modControlState**
For more details, see http://www.mendipdatasystems.co.uk/set-controls/4594398114

NOTE:
Not all controls can be hidden / disabled / locked.
The subform summarises the properties available for each control type

Subform A *List of control types and available states (READ ONLY)*

ControlType	Show / Hide	Enable / Disable	Lock / Unlock
acBoundObjectFrame	Yes	Yes	Yes
acCheckBox	Yes	Yes	Yes
acComboBox	Yes	Yes	Yes
acCommandButton	Yes	Yes	Yes
acCustomControl	?	?	?
acImage	Yes	No	No
acLabel	Yes	No	No
acLine	Yes	No	No
acListBox	Yes	Yes	Yes
acObjectFrame	Yes	Yes	Yes
acOptionButton	Yes	Yes	Yes
acOptionGroup	Yes	Yes	Yes
acPage	Yes	Yes	No
acPageBreak	No	No	No
acRectangle	Yes	No	No
acSubform	Yes	Yes	Yes
acTabCtl	Yes	Yes	No
acTextBox	Yes	Yes	Yes
acToggleButton	Yes	Yes	Yes

Record: 1 of 19 No Filter Search

Issues and Solutions

The resizing code is most successful with **maximised forms** and the **overlapping windows** option.
If using **non-maximised forms**, you should check the effect of resizing at different resolutions.
If necessary, adapt the form size/shape for best results.

NOTE: Built in Access forms like message boxes, input boxes and error messages are NOT resized.

Popup forms appear on top of the application window and therefore may not fit fully on the screen if resized.
It may be necessary to exclude certain popup forms from being resized.

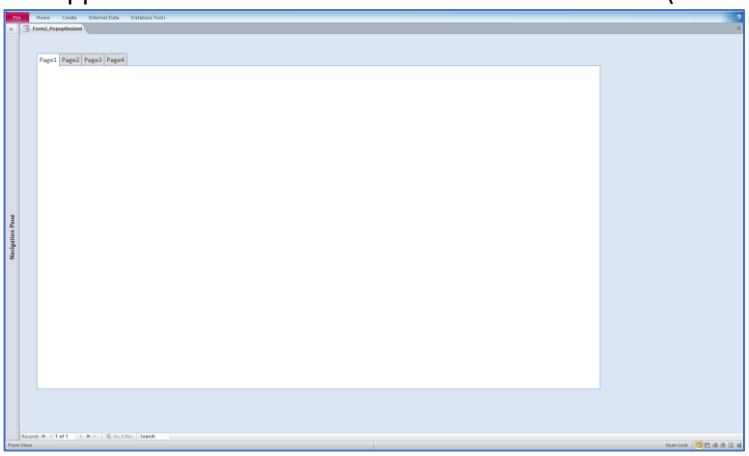
The **Tabbed documents** option can cause issues with some resized forms. For example, controls such as **subforms** can be moved off screen causing the form to be shifted to the right and/or downwards
It is STRONGLY recommended that forms are both designed and displayed using the same display option to minimise such issues.

A particular issue relates to the use of **popup forms** and **tab controls** when using **tabbed documents**. A correction factor may be needed in such cases. The **ResizeForm** code is modified where a form is assigned a **Tag value = 1**.

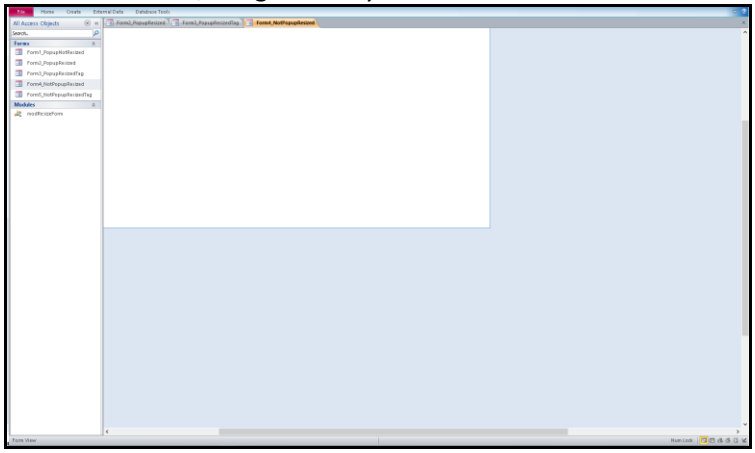
The attached example app **PopupOrNot_TD.accdb** shows the effect and correction in action
It contains 5 forms each with a tab control:

- Form1 - Popup – not resized
- Form2 – Popup – resized
- Form3 – Popup – resized with Tag value = 1
- Form4 – Not Popup – resized
- Form5 – Not Popup – resized with Tag value = 1

The appearance of the resized forms should be like this (with no scrollbars visible):



However **Form4**, is significantly shifted downwards and to the right so it looks like this:



Using both scrollbars, the form can be correctly 'realigned'.
Similarly if the navigation bar and / or ribbon is maximised then minimised again.
However even if the form is saved after doing so, it will again be shifted the next time it is opened

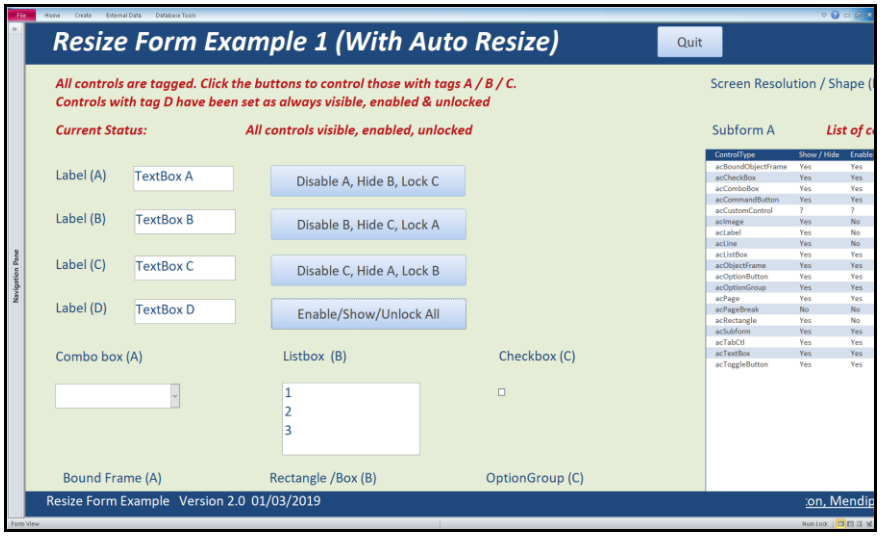
The solution is to apply a **tag value = 1 to the form.**
Doing so causes the exemption code in the **ResizeForm** procedure to be implemented.

NOTE:
a) the issue **MAY not occur** with **popup forms** containing **tab controls**
b) The issue does **NOT occur** when using **overlapping windows** – see **PopupOrNot_OW.accdb**

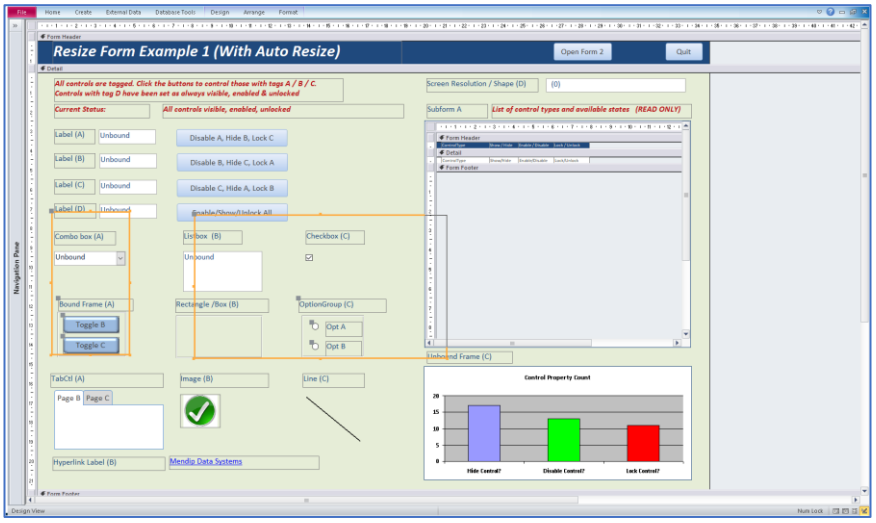
Datasheets can NOT be resized. Recommend instead using a **continuous form** to emulate a datasheet

Button captions may not quite fit when forms are resized for small screens such as a tablet
To prevent this issue, ensure the button width is slightly greater than the length of the caption.

Option groups can occasionally become **'over enlarged'** switching between **form view & design view** during the development process. This can cause the entire form to expand far more than required



The cause of the problem can be seen in design view:



Luckily this is rare if the form is well designed.
However, it is a particular issue if the **option group** is near the **bottom or right edge** of the form.
To prevent this issue, **place option groups as far left and near to top of the form section** as possible.

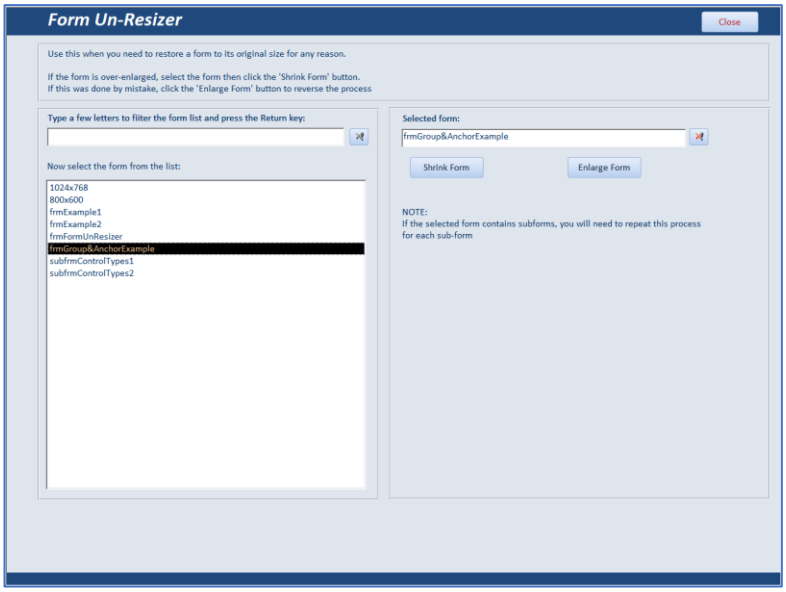
The following quote is taken from the help file supplied with the original code by Jamie Czernik:
Tab controls and options groups are difficult to resize as Access tries to keep the child controls within the frame while the child controls are being moved and resized. This can lead to distortion if the controls are too close to either the top/bottom or left/right of the form being resized. If you forms tend to be scaled up to higher resolutions then try to keep tab controls and option groups as far left and near to top of the form section as possible. The reverse is true if forms tend to be scaled down to lower resolutions so in this situation try to keep them as far right and near to the bottom as possible.

If a form does become 'over enlarged' it can be fixed using one of these methods:

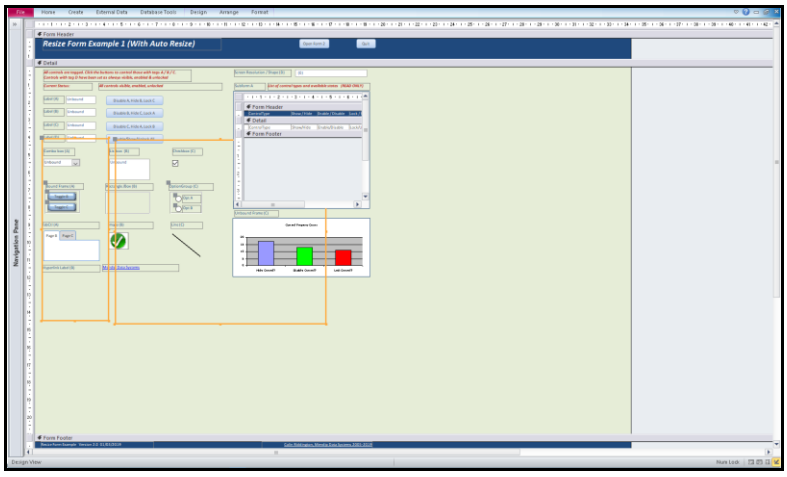
a) Open the **FixFormSize** procedure and enter the form name where indicated
Run the procedure whilst the form is open in **design view**

```
Sub FixFormSize()

'Colin Riddington 13/06/2014

On Error GoTo Err_Handler
'Make sure the form you need to un-resize is open in DESIGN VIEW before running this process
Dim frm As Access.Form
Set frm = Forms!frmPostalAddresses 'modify name as appropriate
UnReSizeForm frm
'ReSizeForm frm

Exit_Handler:
    Exit Sub

Err_Handler:
    MsgBox "Error " & Err.Number & " " & Err.description & " in FixFormSize procedure:", vbExclamation, "Program error"
    Resume Exit_Handler

End Sub
```

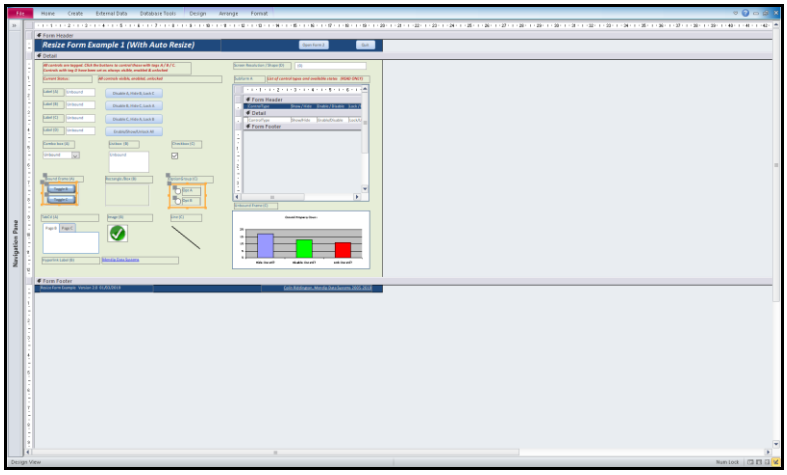
b) Use the form **frmFormUnresizer.** Select the over enlarged form from the **list** then click **Shrink Form**



In each case the form will be scaled down and controls shifted back into position.



However, you will still need to **restore the original size** of the **option group controls** that were over enlarged then **reduce the height and width** of the form yourself.



Finally **save** and **close** the form

NOTE: The process can be reversed if necessary by clicking the **Enlarge Form** button

Using form resizing with your own applications

You will need to import the module **frmResizeForm** and (optionally) form **frmFormUnresizer.**
Then follow the instructions as above.

Related items:

- The original **auto form resize** utility by **Jamie Czernik** – **afr.zip (**MDB zipped)
- 3 example apps referenced in this article - **ResizeFormExamples.zip** (ACCDB files zipped)
  - **ResizeFormExample v2.0.accdb / PopupOrNot_OW.accdb / PopupOrNot_TD.accdb**
- Extended version of this article as a PDF document – **ResizeForm Me.pdf**

If you have any comments or questions, please contact me using the website feedback form or by email (see contact page on website)

Colin Riddington *Mendip Data Systems* *28/02/2019*