
Exercises
Introduction to VBA Programming with ArcObjects
July 18, 2007
UNI GeoTree Center

Overall Goal: The overall goal of the exercises is to learn the basics of customizing the ArcGIS interface and developing custom tools using VBA and ArcObjects. The following areas will be covered.

1. Customizing ArcGIS interface.
2. VBA environment
3. VBA programming concepts
4. ArcObjects overview
5. Custom tool development
    - Map display
    - Selecting features and records
    - Geoprocessing

Assumption: These exercises assume that you have a strong background using ArcGIS and that you have the desire to learn how to develop customized tools.

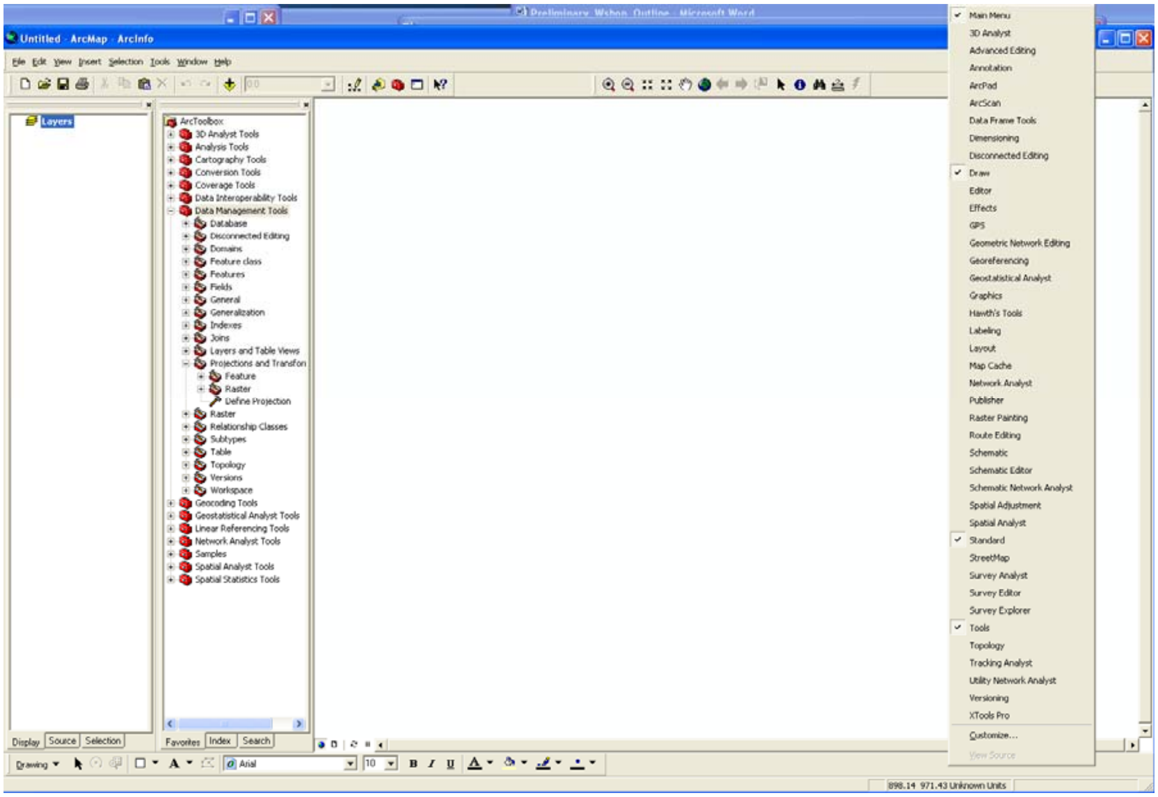
Exercise 1: Customizing ArcMap Interface

This exercise will provide hands-on experience in customizing the user interface of ArcMap.

This set of exercises is meant to practice and reinforce the steps necessary to customize the ArcMap interface.

Turn toolbars on and off

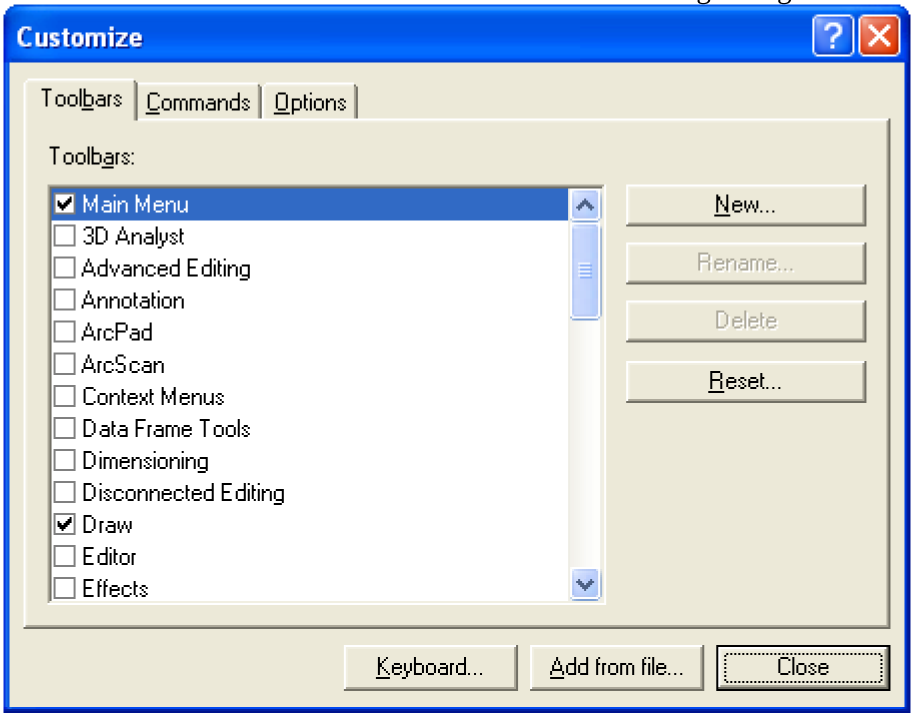
1. Open ArcMap.
2. Right-click anywhere in the toolbar area at the top of the map document and practice turning the toolbars on and off.



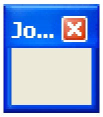
3. Turn on/off toolbars so interface is as it was.

Add new toolbar

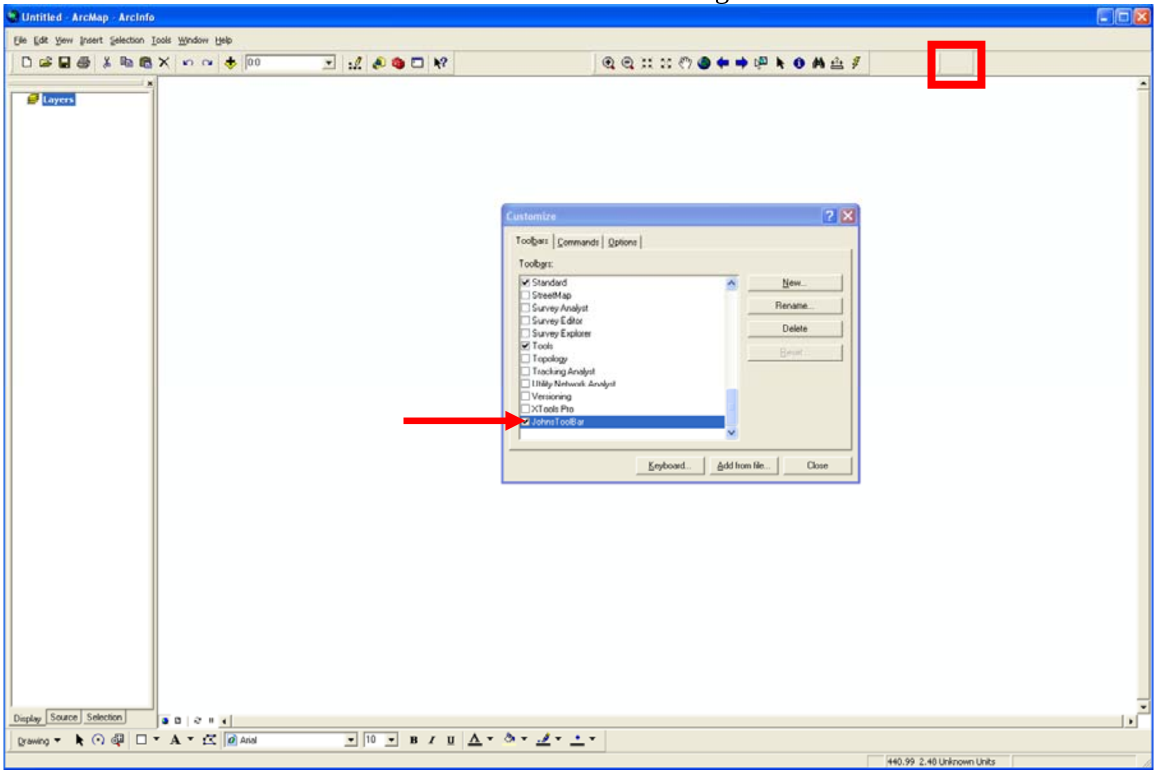
4. Right-click again as above in Step 2 and choose Customize.
5. Make sure that the 'Toolbars' tab is chosen in following dialog.



6. Click the ‘New…’ button on the right. In the dialog that comes up set the Toolbar Name: to something like ‘YourNameToolbar’. Set the ‘Save In:’ combobox to ‘Untitled’. ***Do not set to ‘Normal.mxt’***.
7. Click OK and you should see a toolbar has been added to the map and is floating.

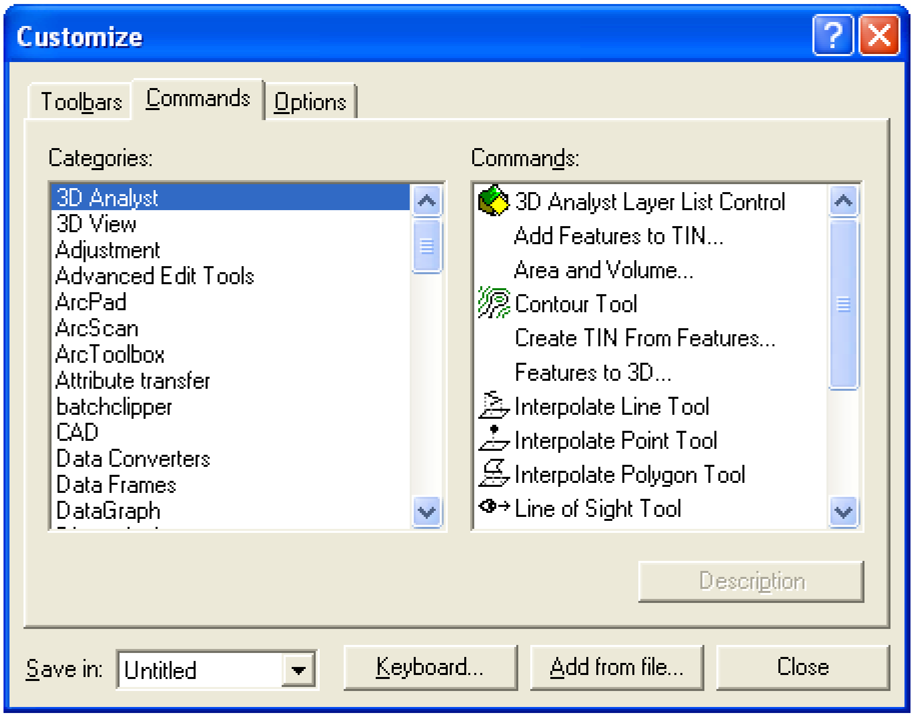


8. Dock the toolbar in the toolbar area. You should now see an empty toolbar and the toolbar should be listed in the list in the Customize dialog.

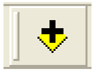


Add a new command to the toolbar

9. To add a new button to the toolbar switch tabs in the Customize dialog by clicking on the Commands tab.
10. The dialog will now have two panels: ‘Categories:’ and ‘Commands:’. Make sure that ‘Untitled’ is selected in the ‘Save in:’ combo box at the bottom. ***Do not choose ‘Normal.mxt***’.



11. In the 'Categories:' tab choose 'File'
12. In the 'Commands:' tab click on the Add Data... command and drag it to your new toolbar.
13. Your toolbar should now look like the following:



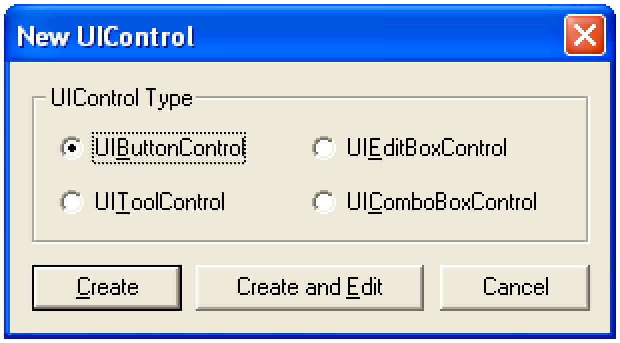
14. Close the Customize dialog.
15. Now click on the button. Navigate to the ….. directory and add the IowaCounty.shp in D:\VBAWshop\Spatial data to the map.

Add a tool

16. Open the Customize dialog again and choose 'Selection' from the 'Categories:' panel and 'Identify' Identify from the 'Commands:' panel. Click and drag this on to your toolbar.
17. Close customize dialog and use the information button in relation to the county dataset.
18. Close the identify window.

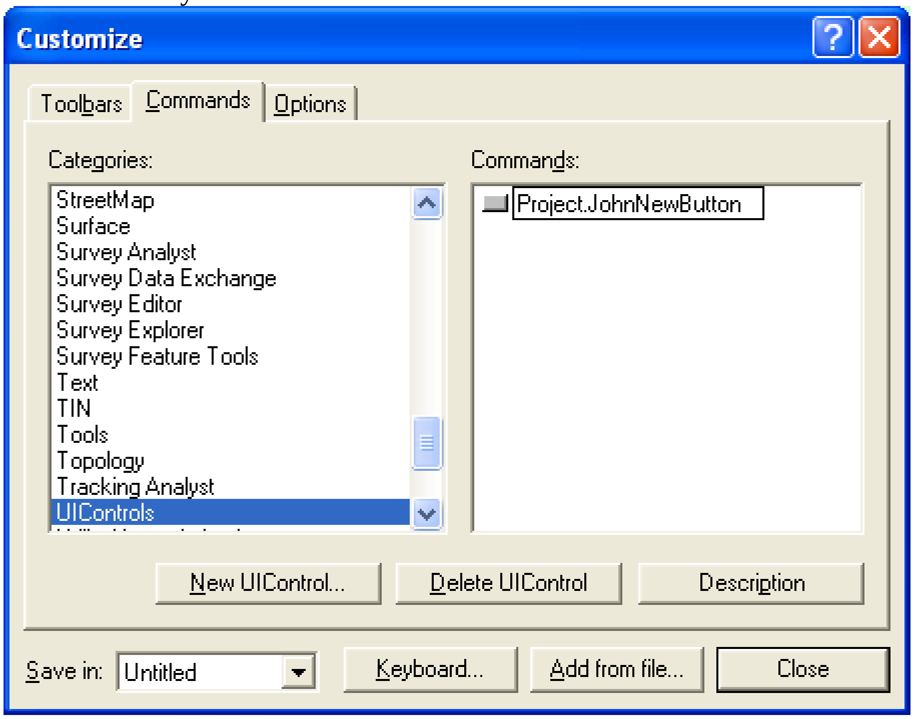
Add a new button to your toolbar

19. Open the Customize dialog again and in the 'Categories:' panel choose 'UIControls'. Make sure that 'Untitled' is selected in the 'Save in:' combo box at the bottom. ***Do not choose 'Normal.mxt***'.
20. Click the New UIControl button. You will see the following dialog:



● UIButtonControl

21. Make sure the UIButtonControl radio button is selected. Click the Create button.
22. You will see that a control has been added to the 'Commands:' panel. Rename the button by clicking on the control twice slowly and placing your cursor at the end of the name. Backspace or delete the characters that says 'UIButtonControl1'. Type in a new name such as 'MyNameNewButton'.

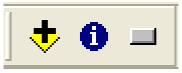


Categories: UIControls

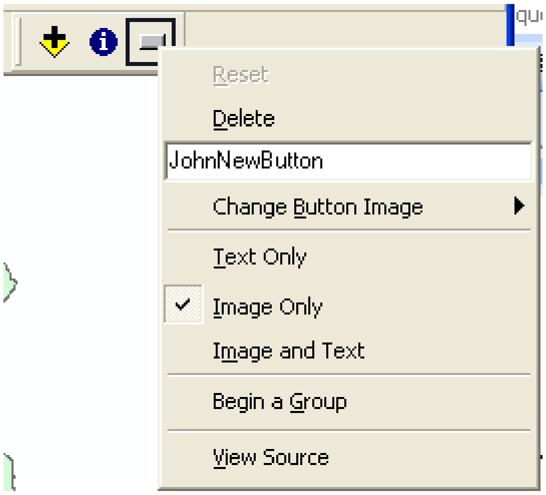
Commands: Project.JohnNewButton

Save in: Untitled

23. Now click on the control in the 'Commands:' panel and drag it onto the end of your new toolbar. Your toolbar should now look like:



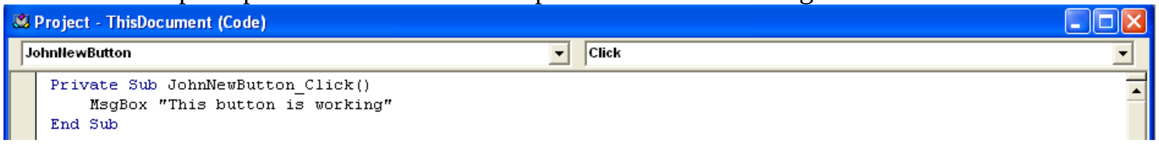
24. With the Customize dialog still open right click on the button just added to the toolbar.



25. In the context menu you see different options for the control. You can have the button have an image, text, or both to represent the button. Choose the ‘Change Button Image’ and choose the Happy Face icon ☺.

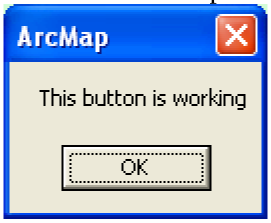
Attach code to control

26. Close the Customize dialog and right click on the happy face icon on your new toolbar and choose ‘View Source’.
27. This will open up the VBA interface. It opens an area for writing code



```
Private Sub JohnNewButton_Click()
    MsgBox "This button is working"
End Sub
```

28. The cursor will be on the blank line in between the ‘Private….’ And ‘End Sub’ lines. In the blank line type the code that is shown above (i.e. msgBox “This button is working”).
29. Now close the Visual Basic window by clicking the red and white x in the upper right of the ‘Microsoft Visual Basic – Project’ window.
30. In the ArcMap click on the happy face icon. You will see the following message box.



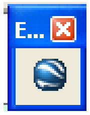
This button is working

31. Save the .mxd as Exercise1_Final.mxd to D:\VBAWshop\ArcMap\YourResults.

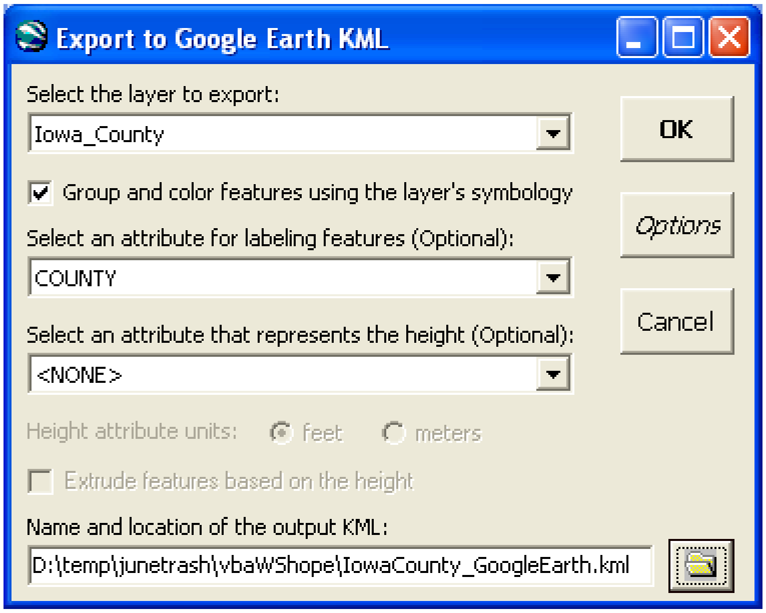
Challenge – Exercise 1

1. If not open, reopen the .mxd you were working with.
2. Open up the Customize dialog again.
3. Make sure the ‘Exercise1_Final’ is chosen in the ‘Save in:’ combobox. ***Do not choose ‘Normal.mxt***’.
4. Click the ‘Add from file…’ button.

5. Navigate to D:\VBAWshop\misc and choose the ‘ExportToKML.dll’ and click ‘Open’. You will see the following dialog listing the objects that have been added to the document.
6. Click on the ‘Toolbars’ tab in the Customize dialog.
7. You will see that a new toolbar called ‘Export to KML’ has been added. Check this toolbar to turn it on.
8. You will see a new toolbar has been added with a single button.



9. Click the Google Earth button.
10. In the dialog that open set the ‘Select the layer to export:’ to ‘Iowa_County’ and the ‘Select an attribute for labeling features (Optional):’ to ‘COUNTY’. Navigate to D:\VBAWshop\Spatial and name the output .kml file ‘IowaCounty_GoogleEarth’ and set the output file for ‘Name and location of the output KML:’. Leave the other parameters the same. Click OK.



Export to Google Earth KML

Select the layer to export: Iowa_County

☑ Group and color features using the layer's symbology

Select an attribute for labeling features (Optional): COUNTY

Select an attribute that represents the height (Optional): <NONE>

Height attribute units: ● feet ○ meters

□ Extrude features based on the height

Name and location of the output KML: D:\temp\junetrash\vbaWShope\IowaCounty_GoogleEarth.kml

11. Click Yes when it asks if you would like to open the file in Google Earth.
12. In Google Earth navigate to Iowa and you should see your layer overlain.

Exercise 2: VBA Environment

This exercise is meant to reinforce your familiarity with the VBA environment.

1. Open ArcMap
2. Save this map as Exercise2.mxd in D:\VBAWshop\ArcMap\YourResults
3. Add IowaCounty.shp data from D:\VBAWshop\Spatial.

Add a new tool bar

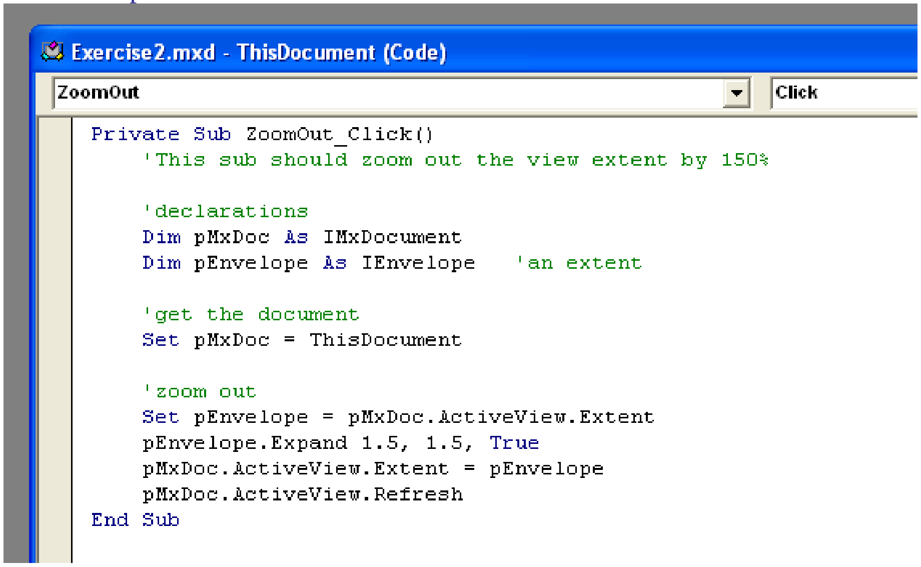
4. Open the Customize window by right-clicking in the toolbar area and choosing Customize.

5. Add a new toolbar following steps 4-8 in Exercise 1. Make sure when you create the new toolbar, you set it to ‘Exercise2’ in ‘Save in:’ combo box. ***Do not choose ‘Normal.mxt***’. You can call the toolbar ‘Exercise2Toolbar’.

Add a button and code

6. Add a new button to the toolbar by following steps 19-25. Set the name of the button to ‘ZoomOut’. Set the icon to whatever you would like but possibly use .
7. Right-click on the button that you just added to the toolbar and choose ‘View Source’.
8. This will open up the VBA editor to the ZoomOut_Click event sub in ThisDocument (in the Exercise2.mxd Project).
9. Copy and paste (or type) the code below inside the stub of the ZoomOut_Click sub (Note: it is common to indent the code inside the routine).

```
'This sub should zoom out the view extent by 150%

'declarations
Dim pMxDoc As IMxDocument
Dim pEnvelope As IEnvelope   'an extent

'get the document
Set pMxDoc = ThisDocument

'zoom out
Set pEnvelope = pMxDoc.ActiveView.Extent
pEnvelope.Expand 1.5, 1.5, True
pMxDoc.ActiveView.Extent = pEnvelope
pMxDoc.ActiveView.Refresh
```

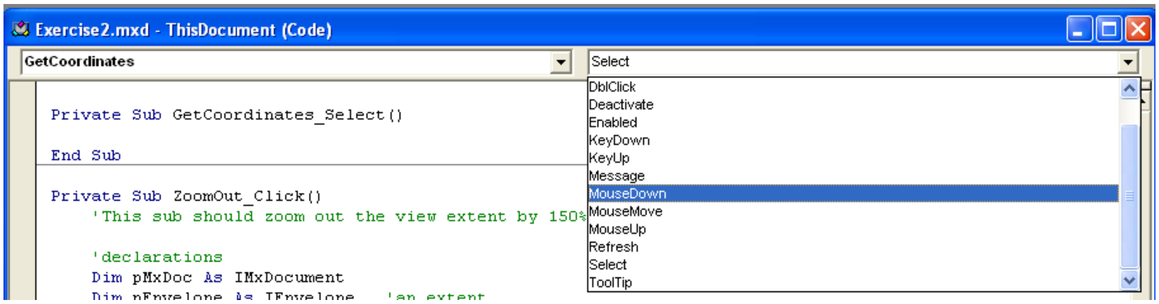


```
Private Sub ZoomOut_Click()
    'This sub should zoom out the view extent by 150%

    'declarations
    Dim pMxDoc As IMxDocument
    Dim pEnvelope As IEnvelope   'an extent

    'get the document
    Set pMxDoc = ThisDocument

    'zoom out
    Set pEnvelope = pMxDoc.ActiveView.Extent
    pEnvelope.Expand 1.5, 1.5, True
    pMxDoc.ActiveView.Extent = pEnvelope
    pMxDoc.ActiveView.Refresh
End Sub
```

10. Go back to ArcMap interface and try clicking the button to see what the effect is.
11. Go back to your code and change the values 1.5, 1.5 to different
12. Read through the code and see if you can logically understand what is happening.

Add a tool and code

13. Add a new tool to the toolbar you created. Open the customize window by right-clicking on the toolbar area and choosing ‘Customize’.
14. Click on the ‘Commands’ tab in the ‘Customize’ dialog. Choose ‘UIControls’ from ‘Categories’ panel. Click the New UIControl. Make sure ‘Exercise2’ is chosen in the ‘Save in:’ combo box. ***Do not choose ‘Normal.mxt***’.
15. In the ‘New UIControl’ dialog choose ‘UIToolControl’ and click Create.
16. Change the name of the control to ‘GetCoordinates’ (see step 22 in Exercise 1).
17. Drag the new tool to your created toolbar.
18. Change the tool icon to or another icon of your choosing (see steps 24 and 25 in Exercise 1).
19. When finished with above steps close the ‘Customize’ dialog.
20. Right-click on the new tool and choose ‘View Source’.
21. This will open up the VBA editor to the GetCoordinates_Click event sub in ThisDocument (in the Exercise2.mxd Project). Notice that this is a different event routine from the button. This event is triggered when the user selects this tool. We don’t want to put any code in this sub.
22. Instead in the right hand combo-box (Procedure) which lists the events associated with the tool choose ‘Mouse Down’.



```
Private Sub GetCoordinates_Select()

End Sub

Private Sub ZoomOut_Click()
    'This sub should zoom out the view extent by 150%

    'declarations
    Dim pMxDoc As IMxDocument
    Dim pEnvelope As IEnvelope    'an extent
```

23. Now in the GetCoordinates_Select routine copy and paste the following code.

```
'This sub should return the x, y coordinates in a message box

'declarations
Dim pMxDoc As IMxDocument
Dim pMap As IMap
Dim pPoint As IPoint
Dim dblX As Double
Dim dblY As Double

'get the document and map
Set pMxDoc = ThisDocument
Set pMap = pMxDoc.FocusMap

'get the point that the user clicked on
Set pPoint = pMxDoc.CurrentLocation
'get the x and y values
dblX = pPoint.x
dblY = pPoint.y

MsgBox "The coordinates of the point clicked are " & dblX & _
      ", " & dblY, vbInformation, "X, Y"
```

24. Go back to the ArcMap interface and use your tool and see what the results are.

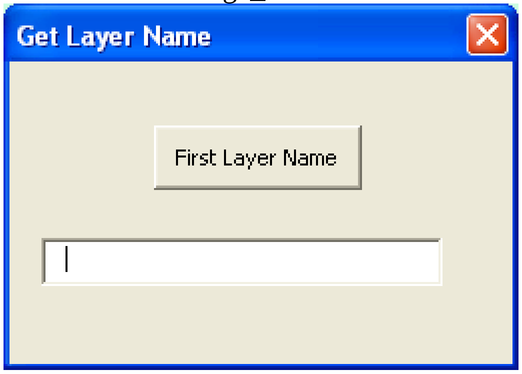
25. To report the numbers as integers to make it look nicer in the message box change the above MsgBox line to:

```
MsgBox "The coordinates of the point clicked are " & Int(dblX) & _
    ", " & Int(dblY), vbInformation, "X, Y"
```

Challenge Exercise 2

This challenge exercise will give you the opportunity to develop your own form for interaction with the user. The code will be provided for you.

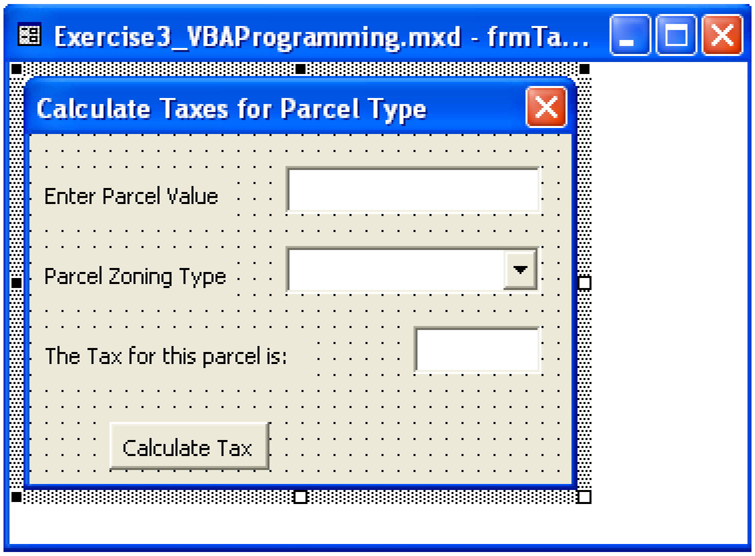
1. Design a form which looks like the following and have the button list the first theme in the map in the text box (form named frmLayerName, command button named cmdLayerName, and text box named txtLayerName). Add a button to your toolbar and open the form with the code in that buttons click event procedure. (Hint: You can get the code from Challenge_Exercise2.mxd in D:\VBAWshop\ArcMap\answers)



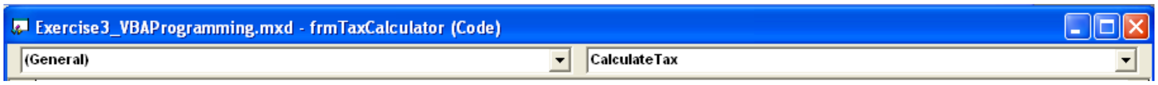
Exercise 3: VBA Programming Concepts

The purpose of this exercise is to reinforce the programming concepts introduced during the lecture. In this exercise some of the controls have already been created and much of the code has been written. You will need to try to read through the code and understand what is going on and fill in the places where code is missing. Places where you should add code are in comments and are identified by '******. This exercise is a hypothetical program to calculate tax amounts for parcels.

1. Open ArcMap and open the Exercise3_VBAProgramming.mxd in D:\VBAWshop\ArcMap, You should see a toolbar with a single button on it in the ArcMap interface .
2. Right-click on the above button and click 'View Source'.
3. The VBA editor will be opened up and you will see the code for the buttons click event (i.e. 'TaxCalculator_Click').
4. Add a line of code to open the form (Hint: use the show method on the frmTaxCalculator).
5. Open the form designer for the frmTaxCalculator by double clicking on the form in the left Project panel. You will see the form designer open up. There are four controls on the form. The first is a text box for entering the amount the parcel is worth. The second is the parcel zoning type.

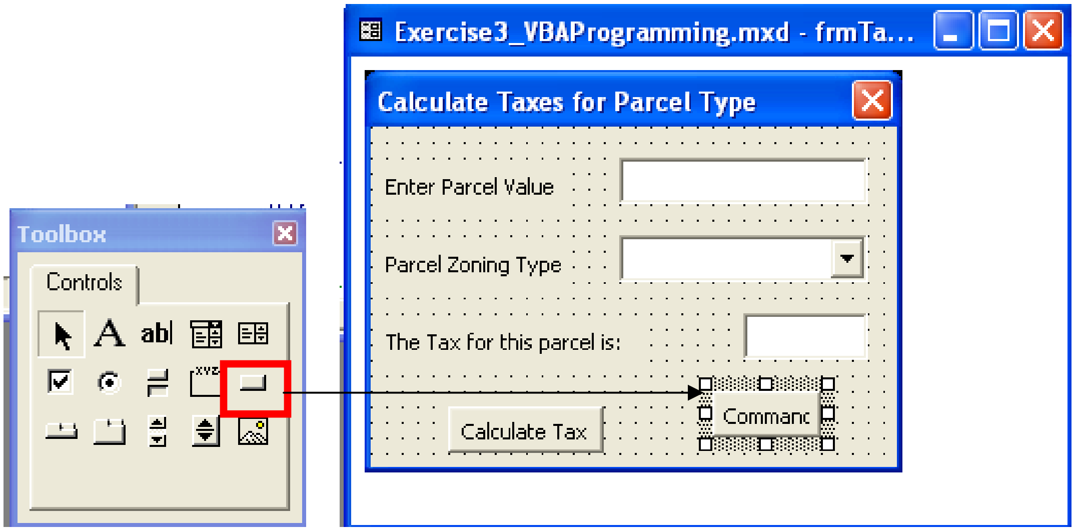

Exercise3_VBAProgramming.mxd - frmTa...
Calculate Taxes for Parcel Type
Enter Parcel Value
Parcel Zoning Type
The Tax for this parcel is:
Calculate Tax

6. Double-click on the form somewhere not on one of the controls. This will open up the code module for the form. You will see that the sub that is opened up in called 'UserForm Initialize'. This code occurs when the form is opening when the code you wrote in Step 5 is executed.
7. Read over the code in this sub and try to understand what is happening. Fill in the two lines of code indicated in the lines with comments starting with '******. If you have trouble filling in the lines ask for help or look at the code in 'Exercise3_VBAProgramming_Final.mxd' in D:\VBAWshop\ArcMap\answers.
8. Now click back on the form designer as shown in Step 6 and double-click on the 'Calculate Tax' button. This will open up the 'cmdCalculate_Click' code sub. Read over the code in this sub to understand what is happening.
9. Fill in the two lines of code necessary to complete this sub so it is functioning properly (indicated in comment lines starting with '******).
10. When finished filling in this sub choose '(General)' from the Object box at the top of the form code module. Then choose 'CalculateTax' from the Procedure box.


Exercise3_VBAProgramming.mxd - frmTaxCalculator (Code)

(General) | CalculateTax

11. This is a Function which calculates the tax amount based on parcel type passed and the parcel value. Read the code in the Function and try to understand what the purpose is. Write the code to finish the conditional logic to calculate the tax amount. The final logical condition should take the following form.
12. When you have finished filling in the code here you are ready to run the program. Go back to the ArcMap interface and click the button.
13. When the form opens up enter a parcel value such as 100000. Choose a parcel zoning type from the combo box. When you have entered these values click the 'Calculate Tax' button and you should see that the tax has been calculated and added to the bottom text box.
14. Now go back to form designer window. Add a new button control by choosing the 'CommandButton' control from the 'Toolbar' and drawing a rectangle next to the 'Calculate Tax' button on the form.


Toolbox
Controls
Exercise3_VBAProgramming.mxd - frmTa...
Calculate Taxes for Parcel Type
Enter Parcel Value
Parcel Zoning Type
The Tax for this parcel is:
Calculate Tax
Commanc

15. You now have a new command button on the form. Make sure it is selected (as above with handles around the new button). In the Properties window on the left side set the '(Name)' to cmdExit. Set the 'Caption' to 'Exit'. Drag the new command button to the right location on the form.
16. Double click the command button to open the code module and the sub for 'cmdExit_Click'.
17. Add in a comment line indicating that this will hide the form. Now add one line of code which hides or closes the form (Hint: use Hide method on the form).

Challenge Exercise 3

This challenge will demonstrate the idea of trying to check for errors in getting values from the users. You will have to check for problems and report problems to user.

1. Try running the program again but instead of typing a number in the 'Enter Parcel Value' text box enter a set of letters. Then click the 'Calculate Tax' button.
2. You will get an error message saying 'Run-time error '13': Type Mismatch'. Click 'Debug' on this error dialog. You will see that the code module will open up and the line causing the error is highlighted in yellow.
3. Click the ■ reset button in the VBA environment to stop the code from running.
4. You now know the line that is causing the problem in the 'cmdCalculate_Click' event procedure.
5. Change the code in here so that if the user enters a non-numeric value for the parcel value an error message will occur and the program will exit. (Hint: use an If…Then conditional logic, IsNumeric function to check if the string returned can be converted to a number, a message box to tell if it can't, and Exit Sub to leave the sub when error).

Exercises 4: ArcObjects Overview

1. Open an internet browser (Firefox or IE).
2. Go to http://edndoc.esri.com/arcobjects/9.2/welcome.htm
3. In the right panel at the bottom click on ArcObjects Library Reference
4. You will see a list of category folders. Click on the ArcMap folder. Then click on the 'ArcMap Library Object Model Diagram'.
5. Answer the following questions based on this OMD.

a. How many classes are present on this OMD? __________________________
b. What is an example of a property that can be got and set on this class? __________________________
c. Which method would be used to open a new map document on this class? __________________________

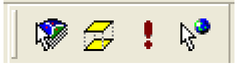
6. Follow steps 2 and 3 and then click on the Carto – Carto Library Object Model. Notice in the pdf that opened you will see that are 10 pages in this document. There are actually 10 separate OMD's.
7. Using the first page (Map and Page Layout) answer the following questions.
   a. Find the DeleteLayer method on the IMap interface. What kind of interface is required as input for this interface? ___________________
   b. What kind of relationship is represented between the Map class and the MxDocument class (this class is on another OMD)? ____________________
8. Navigate to the ArcMapUI OMD and answer the following questions.
   a. On the IMxDocument interface see which property returns the IMap interface? ___________________________________
   b. How many methods are there on the IMxDocument interface? ___________________

Challenge Exercise 4

1. Answer the following questions:
   a. Which method(s) would return an IWorkspace object from the WorkspaceFactory class (Hint: see the GeoDatabase OMD)? ____________________
   b. What are the class types of the FeatureDataset, GeoDataset, and DataSet classes (Hint: see the Geodatabase OMD? _________________________________
   c. What is the relationship between the Dataset class and the Workspace Class? __________________________________
   d. What is the relationship between the FeatureDataset class and the GeoDataset class? ____________________________

Exercuse 5: Using ArcObjects: Map Display

In this exercise some buttons have been added to a new toolbar in an ArcMap project. Some code has been written in the event procedures and you are expected to finish the code. These tools all deal with map display and manipulation of layers in a map.

1. Open the UsingArcObjects_Exercise5.mxd in D:\VBAWshop\ArcMap.
2. You should see that there is a toolbar with 4 buttons on it. 
3. Right-click on the first button (left-most button) and choose 'View Source'.
4. The VBA editor will open to the 'GetLayerNames_Click' event procedure.
5. You will see a comment line and some code. Read the code and try to understand what is happening. Inside the 'For I = …..' loop fill in two lines of code that sets the pLayer variable using a method from the IMap interface (pMap variable). Then use a MsgBox to report the name of the layers.
6. When you have finished the code go back to ArcMap and click the button to see if it works properly.

7. Now right-click on the second button and click ‘View Source’.
8. In the VBA Editor you will be in the ‘MoveBottomLayer_Click’ event procedure. Read the comments to understand what the purpose of the code is.
9. Fill in the code to finish the event procedure. First set the pMap variable using the same code you have seen multiple times (i.e. 2 lines of code).
10. Next set the intLayerCnt variable based on a method from IMap (see ‘GetLayerNames_Click’ event procedure).
11. Set the pLayer variable by setting it to the last layer in the map (Hint: the layers in maps are counted from 0 to the total number of layers -1).
12. Use the MapLayoutOMD.pdf in D:\VBAWshop\pd or use the ArcObjects Help (Hint: put your cursor on IMap and click F1) to find the right method on IMap to move the layer.
13. When finished with the code go back to ArcMap and test that the button works.
14. Next right-click on the ! button and choose ‘View Source’.
15. You will be in the ‘AddFeatureLayer_Click’ event procedure.
16. First read the comments and code that is there to get an idea of the purpose.
17. Add a line of code to open the workspace (the directory to open is D:\VBAWshop\Spatial)
18. Add a line of code to open the feature class (‘IowaRivers.shp’ in above directory).
19. Add code to set the feature class for the new feature layer.
20. Add code to add the feature layer to the map.
21. Go to ArcMap and test that the button works.

Challenge Exercise 5

In this challenge exercise you are asked to complete all of the code in a procedure. You will have seen or written all of the code that will be necessary. This is a button to zoom based on

1. Right-click on the button and click ‘View Source’.
2. You will see that the ‘ZoomToExtent_Click’ event procedure is open. There is a comment line at top telling you the general purpose. Then there are comment lines with ‘******’ to indicate where you should fill in lines of code to make the procedure work.
3. Complete the necessary code to finish the procedure and then test it by clicking the button in the ArcMap interface.

Exercuse 6: Using ArcObjects: Cursors, Selection Sets, Geoprocessing

This exercise will focus on working with records in tables and spatial attribute tables through cursors and selections sets. Also there will be an exercise to carry out a simple buffering of features.

1. Open the ‘UsingArcObjects_Exercise6.mxd’ in D:\VBAWshop\ArcMap.
2. You will see that there is a custom toolbar added to the map. ………
3. Right click on the button and choose ‘View Source’.
4. Read the code in the ‘SelectCountyPops_Click’ event procedure to get an idea of what is supposed to happen.
5. Fill in the lines of code to make the procedure work properly. These line include:

    a. Setting the map.
    b. Setting the feature layer
    c. Setting up a query filter. In this step you have to set up a query to select features from the feature layer. There are 3 fields which have population data by county (Tot_Pop – total population, MalePop – total male population in a county, FemalePop – total female population).
    d. Setting feature selection to select features
    e. Refresh the map.

6. When finished return to ArcMap and run the tool and see that it works.
7. Change the query in the code to select different counties. Try to use 'AND' and 'OR' in the query to make a more complex query.
8. Now right-click on the button and choose 'View Source'.
9. You will see a set of code with comments in the 'Buffer_Features' click sub. Read through the declarations, code, and comments to get an idea of what is going on.
10. Go through the lines commented with '********* to fill in the necessary code.
11. All of the code you need to fill in should have seen been demonstrated today. Use the OMD's and online help to help you figure out. In this procedure 2 procedures will be called to carry out functions. These are already completed routines and just need to be called.

Challenge Exercise 6

In this challenge exercise you will see the combination of many of the things you have learned into one exercise. A custom button code will open and populate a form, the program will accept the parameters from the form, and then carry out some spatial processing. You will be expected to fill in some lines of code but the more important lesson to learn is to see how all of the different aspects fit into the overall design. Code is written in the button click event, in the form control events, and a sub procedure in a stand-alone code module.

1. Open the 'Challenge_Exercise6.mxd' from D:\VBAWshop\ArcMap.
2. You will see that there is one button on a customized toolbar.
3. Right-click on the button and click 'View Source'.
4. You will see the 'Clip_Click' button event procedure code. Read over the comments in the code to get an idea of the purpose of the code.
5. Now open the form in the VBA editor. Click on each of the controls to see what there names are as this will be useful in finishing the code.
6. Double click on the 'Clip' button in the form. This will open up the form code module. Can you guess what the small amount of code is doing in this form module?
7. Open the 'Module1' code module by double-clicking on in Project panel on the left of VBA editor. Read the comments to understand what is happening in the 'ClipGP' procedure. The Geoprocessor object is a special object designed to work with scripting languages in ArcGIS. This provides an efficient way to carry out geoprocessing.
8. Return to the 'Clip_Click' event prodedure code in ThisDocument code module.
9. Fill in the rest of the code to make it work.
10. When finished try to use the tool to clip the statewide features using the 'BlackHawkCounty' layer.