# VB File I/O

FreeFile: Get an available file number

```
FreeFile filenumber  'Integer: 1~255
```

Open: enable I/O to a file

```
Open filename For mod [Access access] As filenumber
```

- Mod
  - ASCII: Append, Input, Output
  - Binary: Random, Binary
- Access
  - Shared, Lock Read, Lock Write, and Lock Read Write

Write/Read

- ASCII Mode
  - Print/Write, Input
- Binary Mode
  - Put, Get

Close

- Close filenumber
- Close [All]

# Write/Read

- Binary
  - **Put** [#]*filenumber*, [*recnumber*], *varname*
  - **Get** [#]*filenumber***,** [*recnumber*]**,** *varname*
- ASCII
  - **Write** #*filenumber*, [*outputlist*]

```
Write #1, "Hello World", 234 ' Write comma-delimited data.
Write #1, ' Write blank line, i.e., insert a newline char.
```

  - **Print** #*filenumber*, [*outputlist*]

```
Print #1, "Zone 1"; Tab ; "Zone 2" ' Print in two print zones.
Print #1, "Hello" ; " " ; "World" ' Separate strings with space.
Print #1, Spc(5) ; "5 leading spaces " ' Print five leading spaces.
Print #1, Tab(10) ; "Hello" ' Print word at column 10.
```

  - **Input** #*filenumber, varlist*

```
Input #1, MyString, MyNumber ' Read data into two variables.
```

"Write #" puts delimiters and quotation marks in the output.

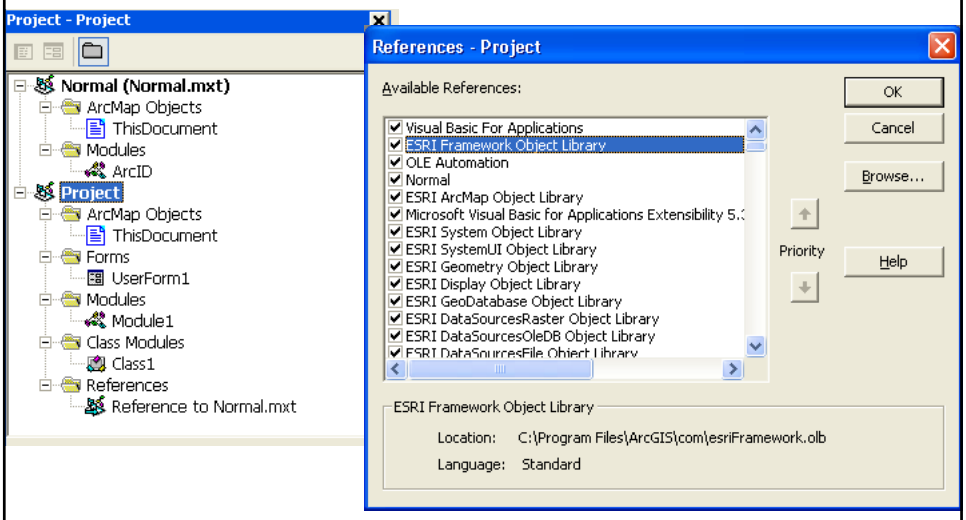
# Example

```
Dim MyString, MyNumber

Open "TESTFILE" For Input As #1 ' Open file for input.

Do While Not EOF(1) ' Loop until end of file.
    Input #1, MyString, MyNumber ' Read data into two variables.
    Debug.Print MyString, MyNumber ' Print data to the Immediate window.
Loop

Close #1 ' Close file.
```

# Random Access Example

```
Type Record ' Define user-defined type.
    ID As Integer
    Name As String * 20
End Type

Dim MyRecord As Record, Position ' Declare variables.

' Open sample file for random access.
Open "TESTFILE" For Random As #1 Len = Len(MyRecord)
' Read the sample file using the Get statement.
Position = 3 ' Define record number.
Get #1, Position, MyRecord ' Read third record.
Close #1 ' Close file.
```

# ArcObjects




# Objects & Variables

- Variables and variable types
  Dim response As Integer

- Objects and classes
  Dim d1 As Dog  'declare an obj variable that could refer to an object
  Set d1 = New Dog  'point the obj variable to a new object
  '(or Dim d1 As New Dog), but never Dim anything as New
  d1.Name = "Sparky"
  d2.Bark

Dog

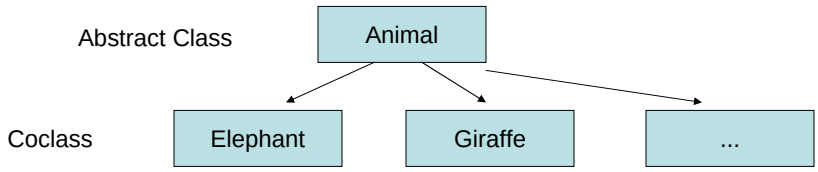
| Name: String |
| --- |
| Bark |

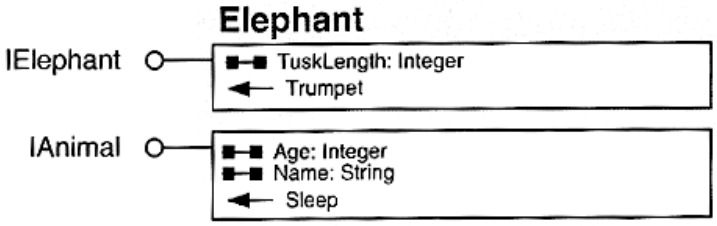
- d1 is an object instantiated using Dog class
  (Dog class is defined in the Dog class module)

# Types of Classes

- A **CoClass** (concrete class) can directly create COM objects by declaring a new object (e.g., FeatureClass).
- An **abstract class** cannot be used to create new objects but is a specification for instances of subclasses (through type inheritance) (e.g., GeoDataset)
- A **Class** cannot directly create objects, but objects of this class can be created as a property of another class or instantiated by objects from another class (e.g., EnumInvalidObject in data conversion)

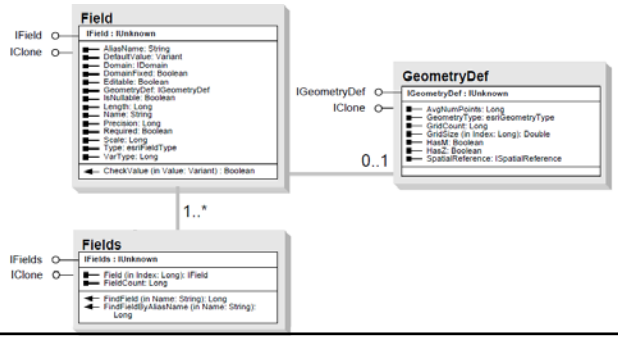
# Examples of Classes



# Types of Relationships (1)

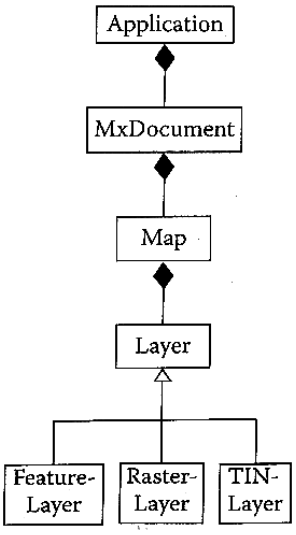
- **Associations** represent relationships between classes. They have defined multiplicities at both ends (i.e., how many instances of one class can be associated with the other class.)
  - **1** - One and only one (if none shown, '1' is implied)
  - **0..1** - Zero or one
  - **M..N** - From M to N (positive integers)
  - ***** or **0..*** - From zero to any positive integer
  - **1..*** - From one to any positive integer




# Types of Relationships (2)

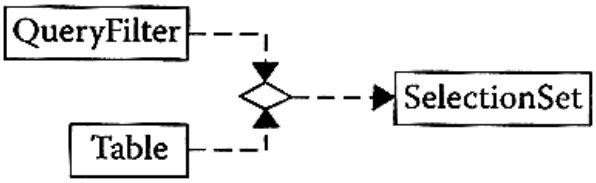
- **Composition** is a relationship in which objects from the 'whole' class control the lifetime of objects from the 'part' class (i.e., Map and FeatureLayer classes).

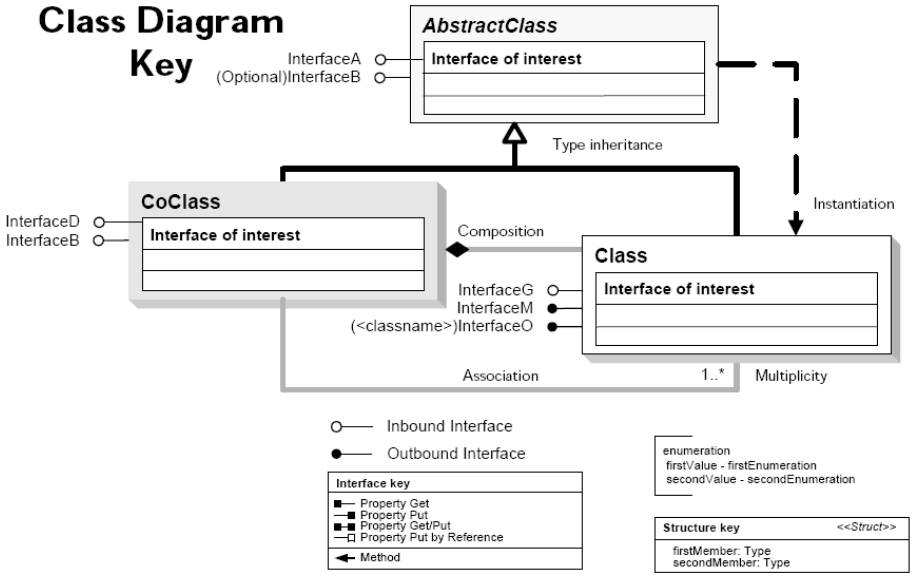
- **Type inheritance** defines specialized (sub)classes of objects which share properties and methods with the superclass and have additional properties and methods.



# Types of Relationships (3)

- **Instantiation** specifies that one object from one class has a method with which it creates an object from another class.
- An **N-ary association (Aggregation)** specifies that more than two classes are associated. A diamond is placed at the intersection of the association branches.



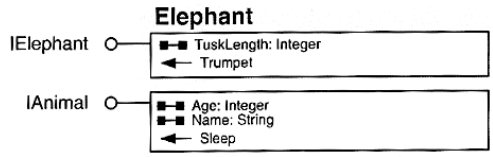
## COM (Component Object Model)

- A framework enables software components to communicate
- Works on Microsoft Windows OS
- Developers then could create re-usable software components (binary codes)
- Component objects always access other component objects through **interface** pointers (to preserve encapsulation of COM objects)
- ArcObjects follows COM architecture
- .NET is a newer framework and complement to COM.

## Interface

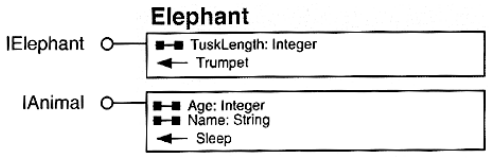
- An interface represents a set of externally visible/accessible operations or properties.
- One would never work directly with objects in ArcObjects.
- Interfaces can be either inbound or outbound. An inbound interface is the most common kind, the client makes calls to functions within the interface contained on an object. An outbound interface is one where the object makes calls to the client, a technique analogous to the traditional callback mechanism.

## Using a component through an interface



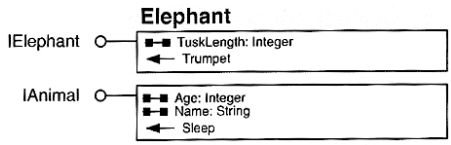

```
Dim pE As IAnimal
Set pE = New Elephant
pE.Name = “Dumbo”
pE.Sleep
Set pE = Nothing
```

## QueryInterface




```
Dim pA As IAnimal
Set pA = New Elephant
pA.Name = “Dumbo”
pA.Sleep

Dim pE As IElephant
Set pE = pA   ‘QI
pE.Trumpet
```

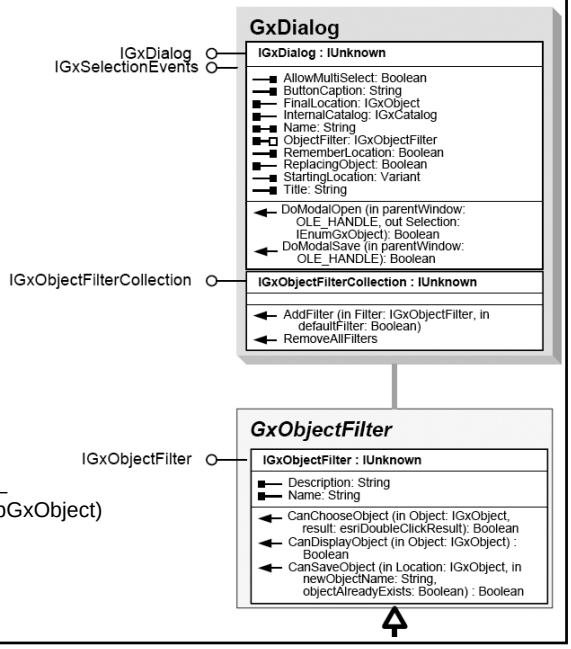
# QueryInterface: TypeOf

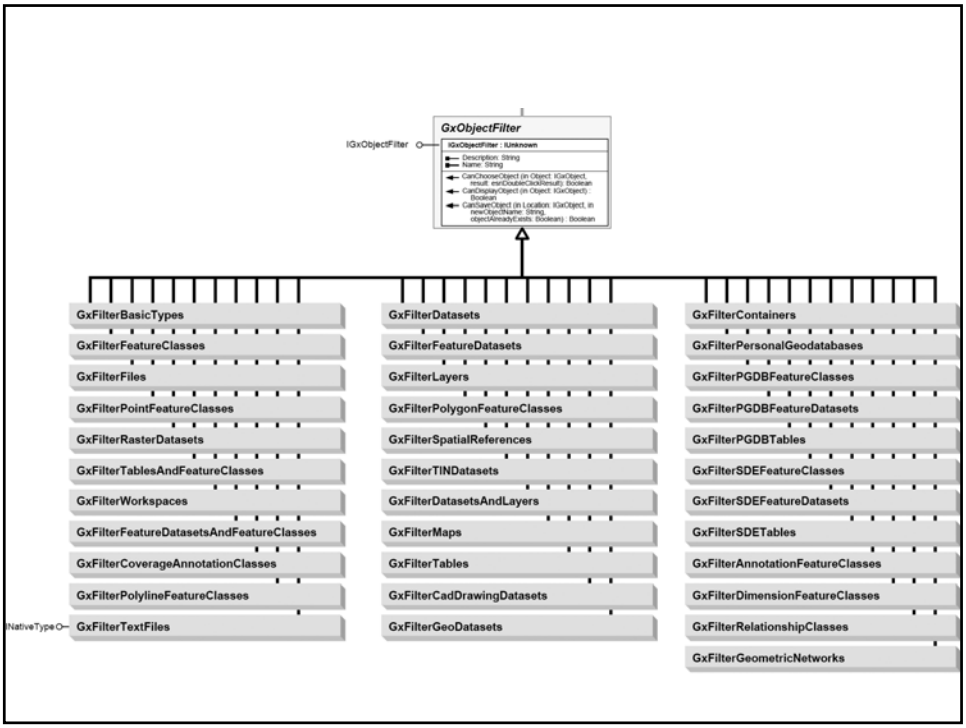



```
Dim pA As IAnimal
Set pA = New Elephant

If TypeOf pA Is Elephant Then MsgBox “I’m an elephant”
If TypeOf pA Is IAnimal Then MsgBox “I have an IAnimal interface”
If TypeOf pA Is IElephant Then MsgBox “I have an IElephant interface”
```

# GxDialog Coclass

```
Dim pGxDialog As IGxDialog
Set pGxDialog = New GxDialog

Dim pFilter As IGxObjectFilter
Set pFilter = New GxFilterFiles

Dim pGxObject As IEnumGxObject
Dim bObjectSelected As Boolean

With pGxDialog
   .AllowMultiSelect = False
   .ButtonCaption = "Select"
   .Title = "Select Input File"
   Set .ObjectFilter = pFilter
   bObjectSelected = .DoModalOpen _
        (ThisDocument.Parent.hWnd, pGxObject)
End With
```

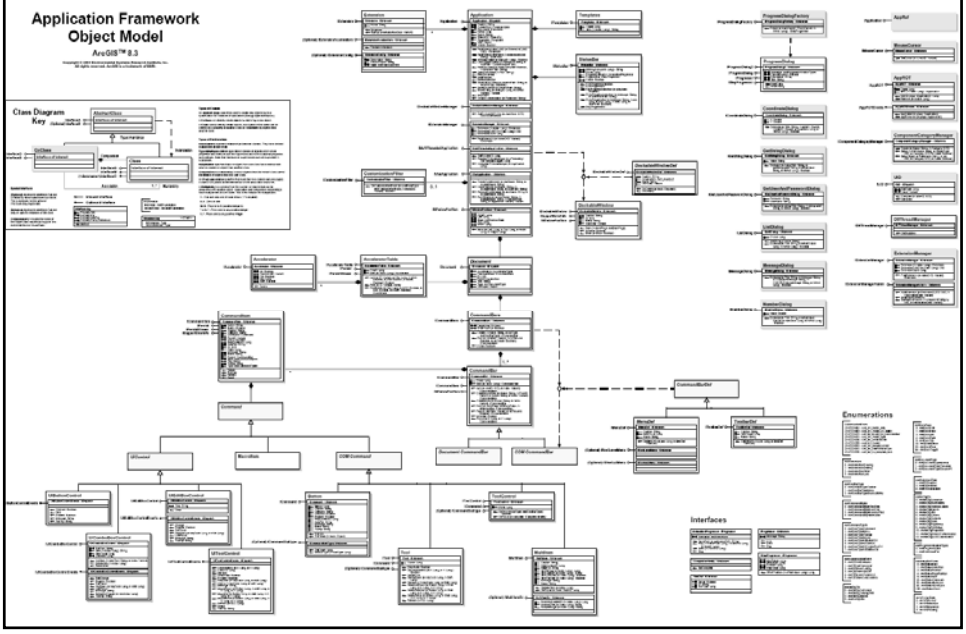





# Example

```
Private Sub cmbSelectInput_Click()
    Dim pGxDialog As IGxDialog
    Set pGxDialog = New GxDialog

    Dim pGxObject As IEnumGxObject
    Dim bObjectSelected As Boolean

    Dim pFilter As IGxObjectFilter
    Set pFilter = New GxFilterFiles

    'initialize and open mini browser
    With pGxDialog
        .AllowMultiSelect = False
        .ButtonCaption = "Select"
        .Title = "Select Input File"
        Set .ObjectFilter = pFilter
        bObjectSelected = .DoModalOpen(ThisDocument.Parent.hWnd, pGxObject)
    End With

    If bObjectSelected = False Then Exit Sub

    'display the selected dataset to ArcMap
    Dim pGxDataFile As IGxFile
    Set pGxDataFile = pGxObject.Next
    txtInput.Text = pGxDataFile.Path
    cmbViewInput.Visible = True
    If Len(Trim(txtOutput.Text)) > 0 Then cmbSort.Enabled = True Else cmbSort.Enabled = False
End Sub
```

## ArcObjects Object Model Diagrams




## ArcMap/ArcCatalog GUI

- Macros – Public Subs in Modules
- Menus – Commands & UIControls
- UIControls

# Types of UIControls

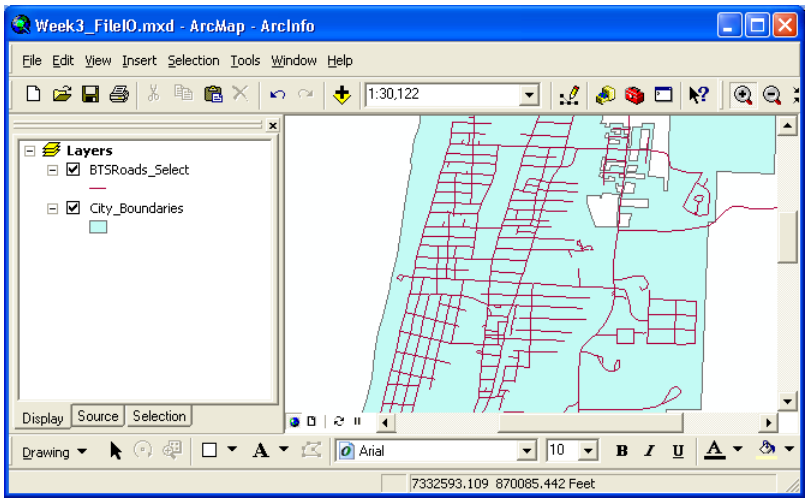
UIButton Control:
- To start, end, or interrupt an action or series of actions.

UITool Control:
- To perform some type of interaction with the display. You can respond to mouse and key events.

UIComboBox Control:
- It combines the features of a text box and a list box. A UIComboBoxControl provides a set of choices from which a selection can be made.

UIEditBox Control:
- To display information entered by the user.

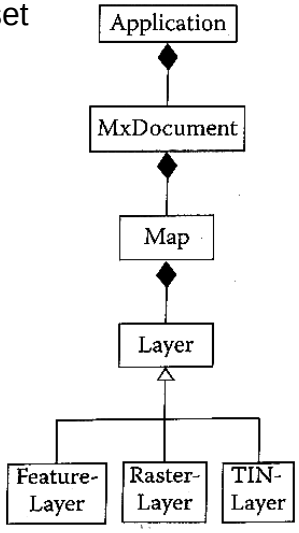

- Application and ThisDocument Objects



# ArcMap Application & MxDocument

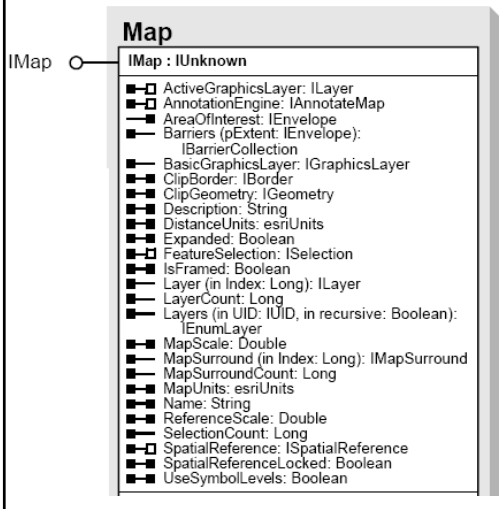
- Application & ThisDocument Objects: preset variables that ArcGIS provides.

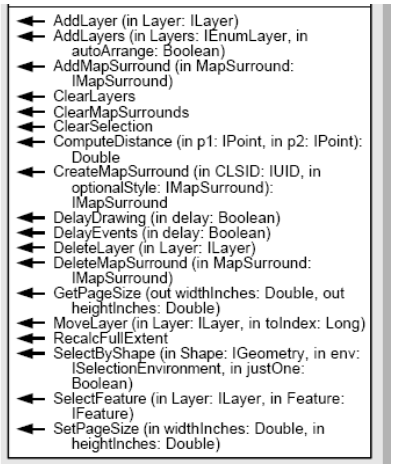
```
Dim pMxDoc As IMxDocument
Set pMxDoc = ThisDocument
pMxDoc.ActiveView.Refresh

Dim pMap As IMap
Set pMap = pMxDoc.FocusMap

Dim pFLayer As IFeatureLayer
Set pFLayer = New FeatureLayer
Set pFLayer.FeatureClass = pGxDataSet.Dataset

pMap.ClearLayers
pMap.AddLayer pFLayer
```




# Map Interface

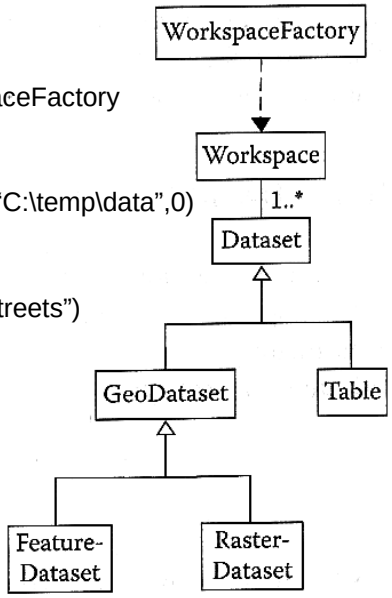
(ArcMap Object Model.pdf)



# ArcObjects Datasets

```
Dim pWkspFactory As IWorkspaceFactory
Set pWkspFactory = New ShapefileWorkspaceFactory

Dim pWksp As IFeatureWorkspace
Set pWksp = pWkspFactory.OpenFromFile("C:\temp\data",0)

Dim pFClass As IFeatureClass
Set pFClass = pWksp.OpenFeatureClass("streets")
```




# WorkspaceFactory Interface
## (Geodatabase Object Model 1.pdf)