VB and the CLR

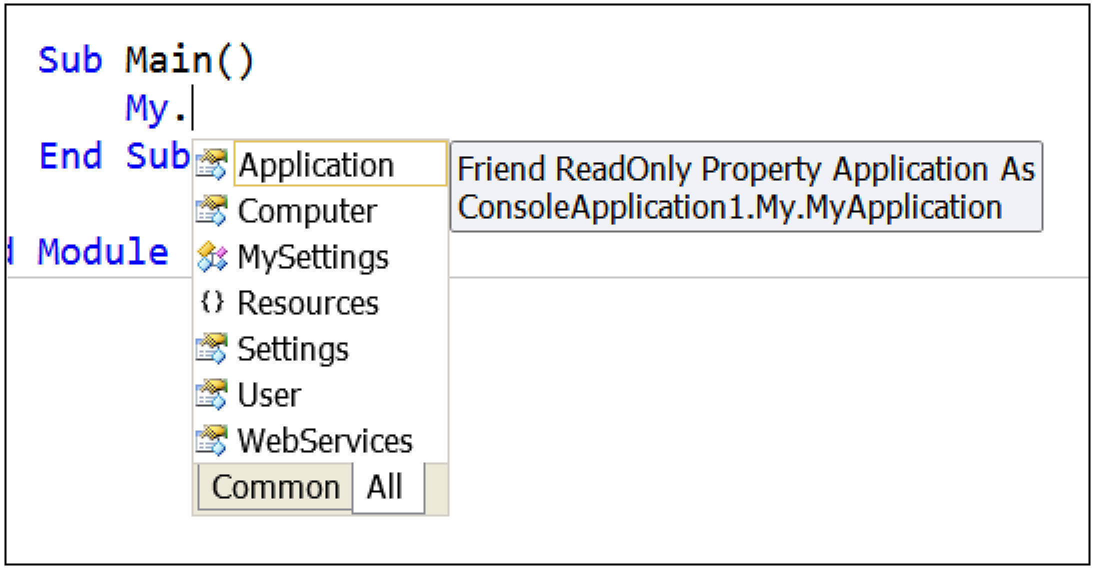
Best Friends Forever

Overview

- Working with the file system
- Garbage collection
- Threads
- COM Interoperability

My Namespace

- Framework class libraries are very large
- My namespace provides “speed dial” for commonly used functionality



```
Sub Main()
    My.
End Sub

d Module
```

Friend ReadOnly Property Application As ConsoleApplication1.My.MyApplication

Working with the File System

- **Use properties and methods from My.Computer.FileSystem**
  - Equivalent FCL types are in System.IO namespace
- **Commonly used methods:**
  - FileExists: check to see if a file exists
  - ReadAllText: read all the text in a file
  - GetDirectoryInfo: get a DirectoryInfo object for a specific folder
    - Folders property gets information about child folders
    - Files property get information about files in the folder
  - GetFileInfo: get a FileInfo object for a specific file

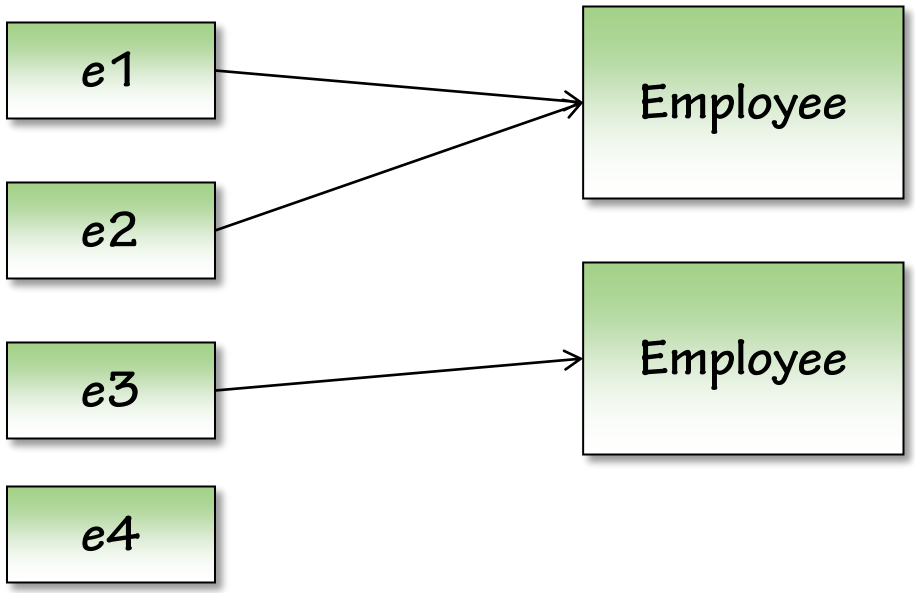
Working with the File System

```
Dim fileSystem = My.Computer.FileSystem

Dim docsFilePath = fileSystem.SpecialDirectories.MyDocuments
Dim docsFolder = fileSystem.GetDirectoryInfo(docsFilePath)

For Each folder In docsFolder.GetDirectories()
    Console.WriteLine(folder.Name)

    For Each file In folder.GetFiles()
        Console.WriteLine(vbTab & file.Name)
    Next
Next

Dim demoFilePath = docsFilePath & "\Demo Documents\Lorem Impsum.txt"
If fileSystem.FileExists(demoFilePath) Then
    Dim contents = fileSystem.ReadAllText(demoFilePath)
    Console.WriteLine()
    Console.WriteLine(contents)
End If
```

Reference Types

- Objects are created using the New operator
- Variables store a reference to an object
- Reference is stored on the Stack, object is stored on the Heap
- Assignment copies the reference
- Multiple variables can point to the same object
- Variables may not have a reference (be set to Nothing)

```
Dim e1 As New Employee()
Dim e2 As Employee = e1
Dim e3 As New Employee()
Dim e4 As Employee = _
    Nothing
```



Garbage Collection

- **Garbage collector cleans up unused objects from the Heap**
  - Only when the application is running low on memory
  - Objects live on even after they are no longer being used
- **Garbage collection mechanism can be a problem if objects are using unmanaged resources**
  - These are generally resources provided by the operating system
    - Memory, file handles, database connections, etc
  - The resources remain in use until the object is garbage collected

```
Sub OpenFile()
    Dim fs As New FileStream("C:\file.docx", FileMode.Open)
    ' ...
End Sub
```

Dispose

- **Objects that use unmanaged resources implement a Dispose method**
  - Frees up unmanaged resources
  - Users of the object should call Dispose when they are finished with the object

```
Dim fs As FileStream = Nothing
Try
    fs = New FileStream("file.docx", FileMode.Open)
    ' ...
Finally
    If fs IsNot Nothing Then
        fs.Dispose()
    End If
End Try
```

```
Using fs As New FileStream("file.docx", FileMode.Open)
    ' ...
End Using
```

Threads

- **System.Threading**
  - Low level API for starting, stopping, and joining threads
  - Here Be Dragons!
- **System.Threading.Tasks**
  - High level API for concurrent and asynchronous programming

```
Dim nums(499) As Integer
For index = 0 To nums.Length - 1
    nums(index) = index
Next

Parallel.ForEach(nums, _
    Sub(num) Console.WriteLine(num))
```

COM Interop

- **COM = Component Object Model**
  - Object model used by many languages and runtimes pre-.NET
- **VB can consume COM components**
  - Classic Visual Basic Active X DLLs for example
  - Just add a reference and Visual Studio will create the required code
- **VB can create COM components**
  - That can be consumed by Classic Visual Basic applications for example
  - Use the COM Component item template
    - Not available in the Express versions of Visual Studio

Summary

- Working with the file system
- Garbage collection
- Threads
- COM Interoperability