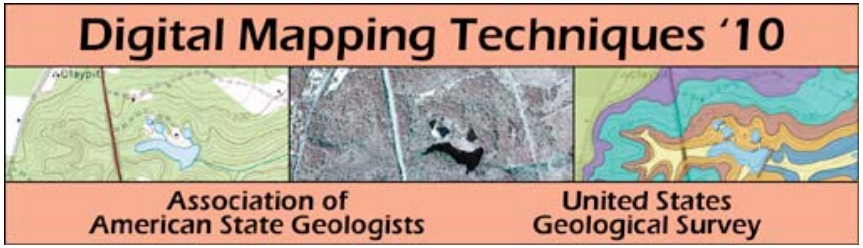


The following was presented at DMT'10
(May 16-19, 2010).

The contents are provisional and will be
superseded by a paper in the
DMT'10 Proceedings.

See also earlier Proceedings (1997-2009)
http://ngmdb.usgs.gov/info/dmt/

Automation in ArcGIS 10:
Understanding the changes taking place and options for migration of legacy code

Andrew L. Wunderlich
gibbon@utk.edu

Tectonics & Structural Geology Research Group

Department of Earth & Planetary Sciences
and Science Alliance Center of Excellence

University of Tennessee, Knoxville

Introduction

Important changes to customizations and automations are coming in ArcGIS 10

Microsoft has dropped support for VB6/ VBA and is emphasizing a shift to .NET-compliant languages

ESRI is removing the familiar VBA development environment from their products, discontinuing support, and promoting new alternatives

Customizing ArcGIS 10

- Still many forms of customization:
  - Layer files, styles, representations, and templates
  - Model Builder
    - simple geoprocessing
  - Python scripting (now w/ ArcPy)
    - advanced geoprocessing
    - automated map production
  - Coding with VB.NET and C#
    - Add-ins
    - stand-alone application development
    - application extensions

Major changes in ArcGIS 10…

Old options phasing out/removed:

- VBA/VB6, UIControl customizations

Advancements and/or improvements:

- Unified ArcObjects SDK for .NET
- Expanded Python support with ArcPy

New options:

- Development environments (Python window, MS Visual Studio)
- Customization interface (Add-ins)

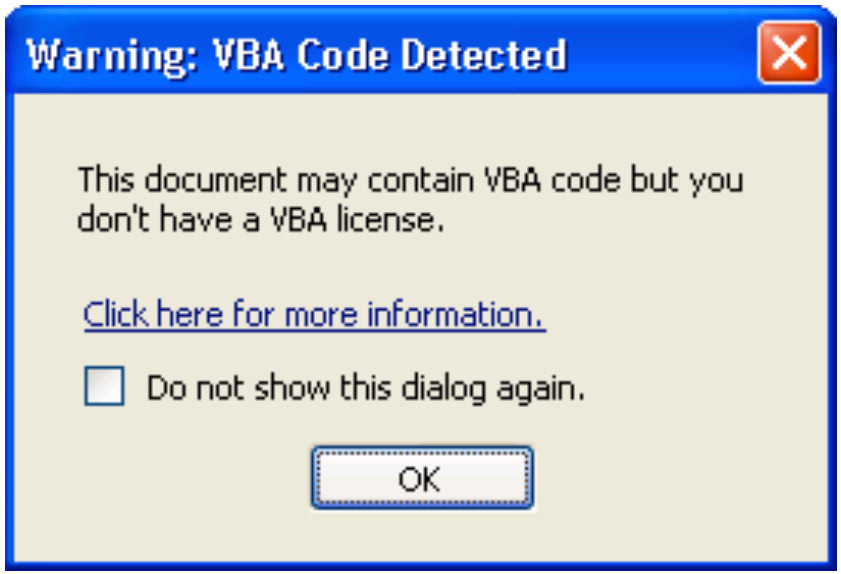
Out with the old…

VBA is being "depreciated":

- VBA ***will not be installed*** with ArcGIS Desktop 10 (but it can be installed separately)
- VBA license file must be requested from ESRI to continue running VBA code
- ESRI *strongly* recommends that no further development occur using VBA and that users migrate their customizations to supported languages (VB.NET, C#, Python, etc.)
- ESRI will no longer support VBA as a development platform after ArcGIS 10.0

What “depreciation” looks like…

- MXDs opened in ArcMap 10 that contain VBA code will trigger a warning message if the additional “VBA setup” is not installed and licensed…



…in with the (sort of) new!

- ESRI has fully embraced Python as the scripting language for ArcGIS 10
- Python code is interpreted at runtime (NOT compiled) and dynamically typed (auto-complete, syntax tooltips, etc.), in an environment similar to the old VBA editor.
- ArcPy site-package for Python 2.6:
  - "ArcPy provides access to geoprocessing tools as well as additional functions, classes, and modules that allow you to create simple or complex workflows quickly and easily."

    - ArcGIS Help

Python scripting in ArcGIS 10

- ArcPy site-package for Python 2.6 exposes:
  - Tools: all ***available*** tools. Depends on license level (e.g. ArcEditor, ArcInfo) and licensed extensions (e.g. Spatial Analyst)
  - Functions: no license dependence, general use: check existence of a dataset, class/dataset property retrieval, refresh the map, create cursors, etc.
  - Classes: parameter framework, used to create objects needed to feed a tool or function (e.g. SpatialReference class)
  - Modules: groups of special classes and functions for manipulation of maps and datasets (e.g. Mapping module)

Use Python w/ ArcPy for…

- Simple to complex geoprocessing workflows (SOME NEW ABILITY HERE)
  - Create an empty feature class, add fields
  - Query layer, export selection, buffer, clip, add a field, calculate attribute, etc., etc.
- Simple or complex map production workflows (LOTS OF NEW ABILITY HERE)
  - Open and manipulate MXD files
  - Fix broken data links
  - Create map books, automate export/print

Limitations of Python

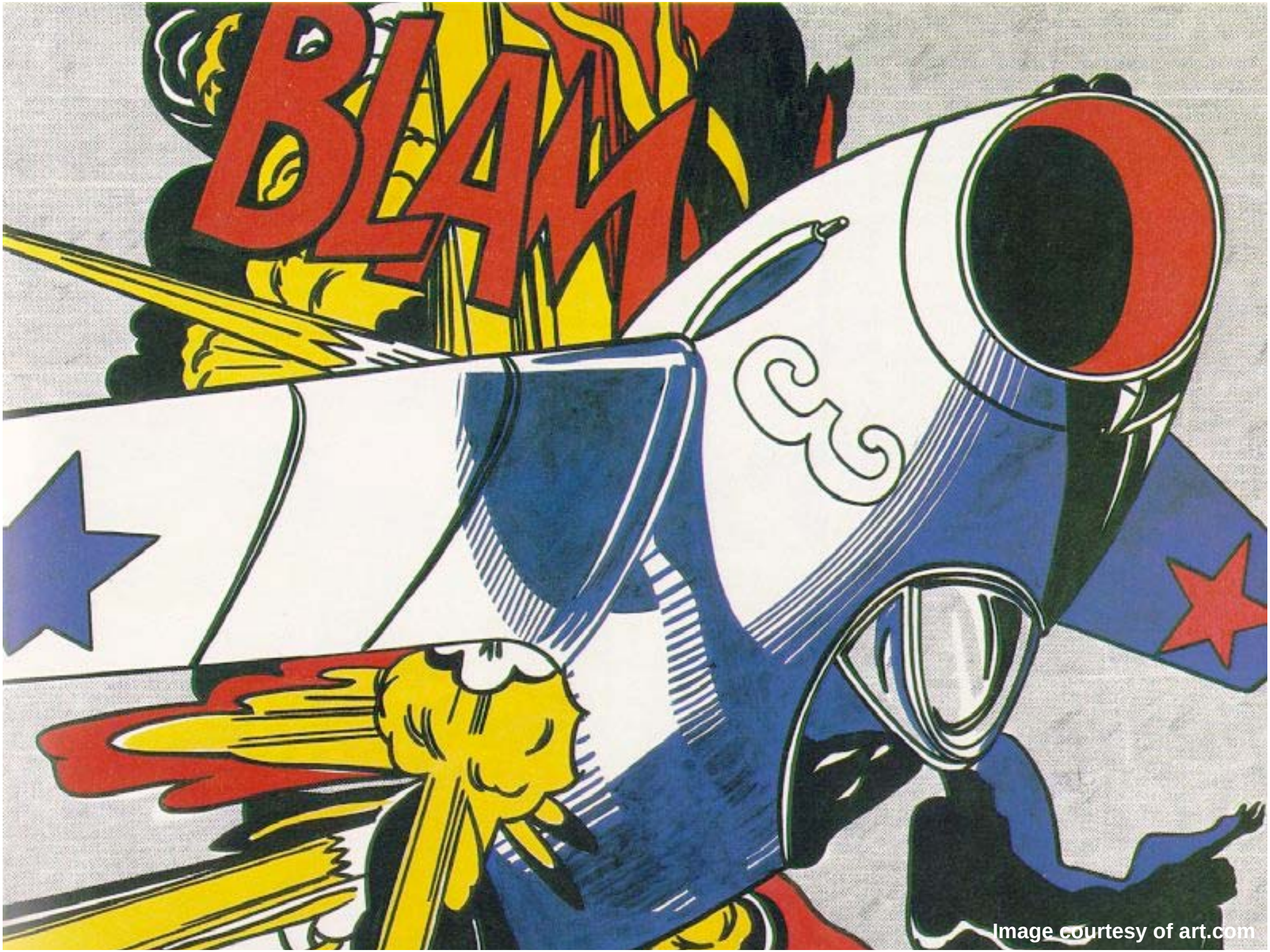
- Not all components of ArcGIS are exposed to Python (although it's much better than before)
- Editing and debugging is less sophisticated than more advanced programming environments such as Visual Studio
- BUT MOST IMPORTANTLY:
  - Cannot listen for and respond to ArcGIS application events
  - Cannot implement any Component Object Model (COM) interfaces including:
    - Buttons and tools
    - ComboBoxes
    - Dockable windows (UserForms)

I need my buttons!

Since VBA is no longer supported, all custom UIControls, including buttons, tools, combo boxes, and edit boxes, need to be rewritten in a compatible language…

What about my application extensions (DLLs) or stand-alone apps written in old VB?

DITTO…


BLAM
Image courtesy of art.com

Introducing Add-ins for ArcGIS

- The new add-in model provides you with a declaratively-based framework for creating customizations packaged within a compressed file
- Add-ins are easily shared between users: they do not require installation programs or COM registration
- Add-ins are added to a system by simply copying them to a well-known folder and removed by deleting them from this folder
- Add-ins can also be shared between users within an organization using a centralized network share.

-From the ArcGIS 10 Help

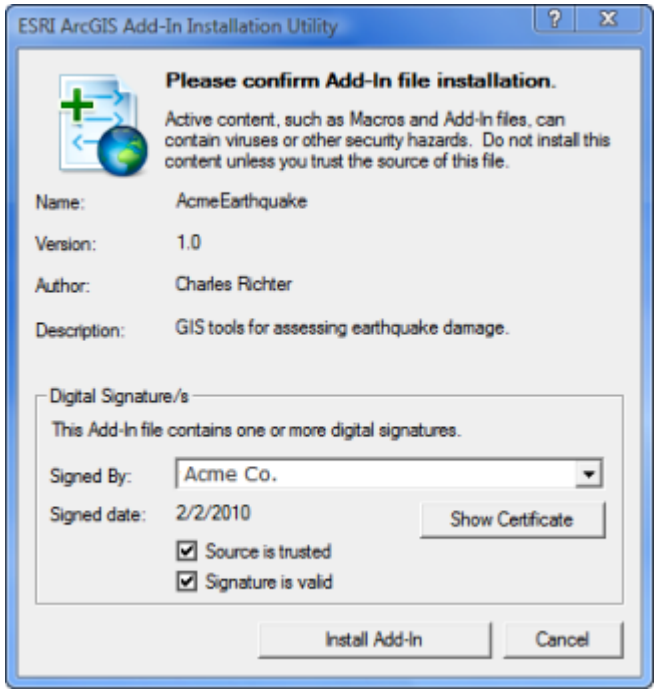
Add-ins …in with the ***really*** new!

- Written in .NET (VB or C#) or Java and accompanied by an XML configuration file
  - XML file describes the customization to ArcGIS
  - .NET or Java class provides custom behavior
- Add-Ins Wizard part of ArcObjects SDK to simplify development by integrating Add-in templates to Visual Studio new project wizard.
- Create custom buttons, menus, forms, application extensions, editor extensions, etc.

Adding Add-ins to ArcGIS

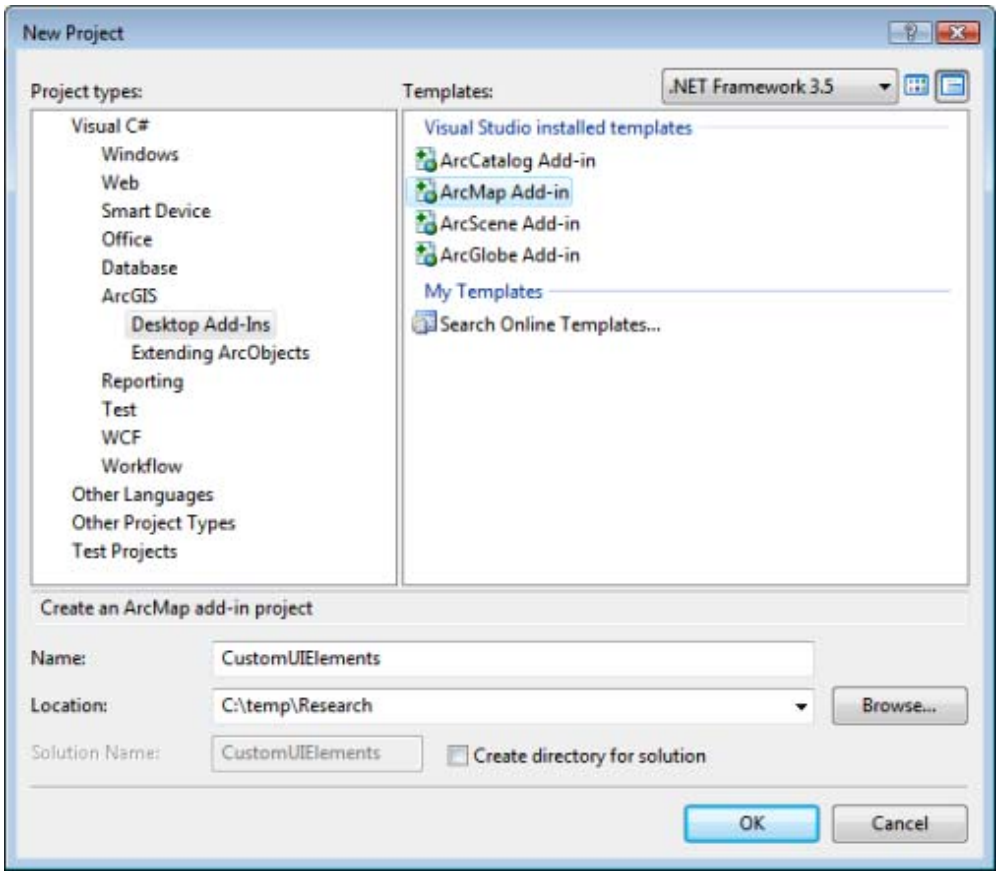
If the add-in file already exists, just double click it to start the ESRI Add-In Installation Utility:

- Add-in will be validated
- Copied to appropriate folder
- Made available in the “Customize” dialog…



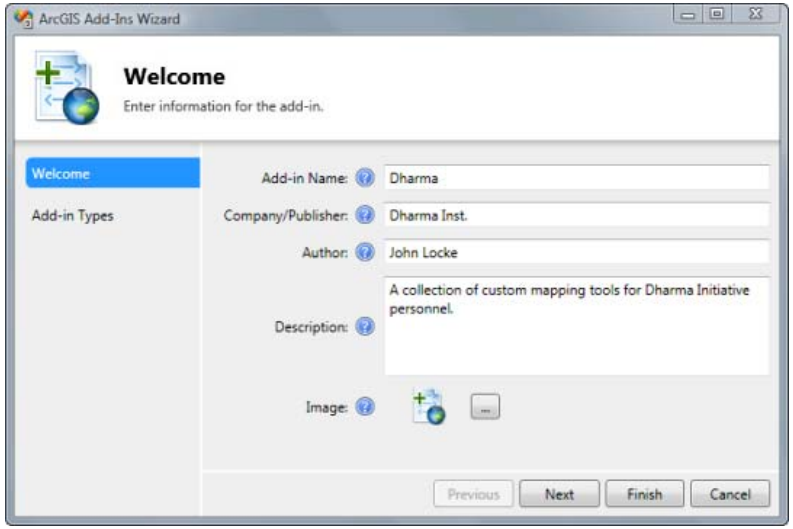
Creating Add-ins w/ Visual Studio

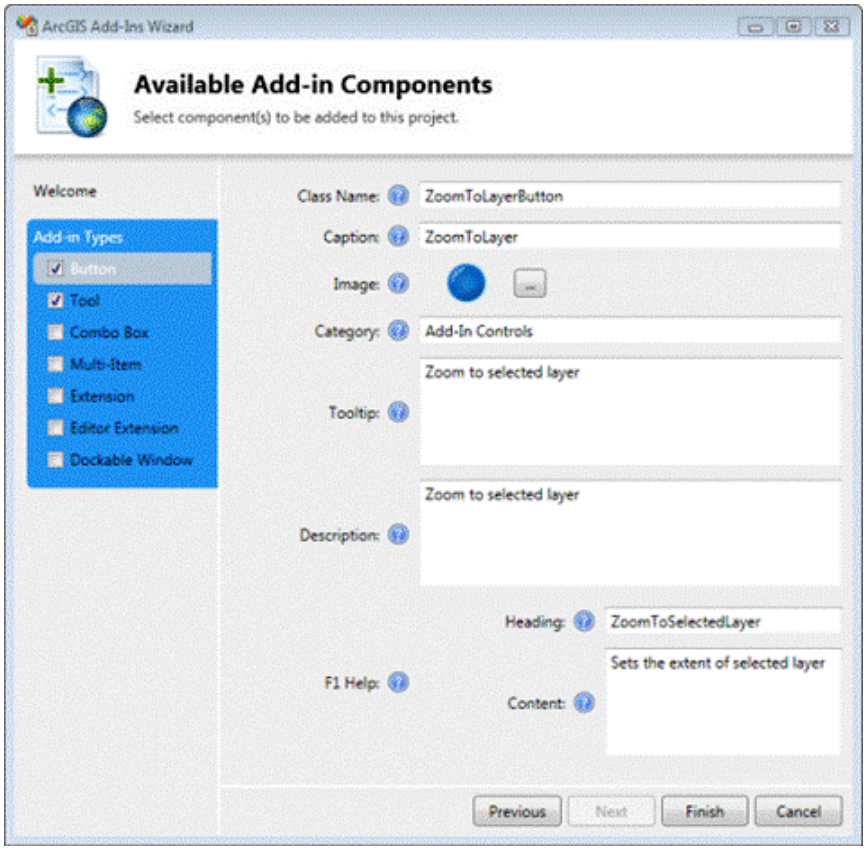
Start a new project and choose which application you want to create an add-in for:



Creating Add-ins w/ Visual Studio

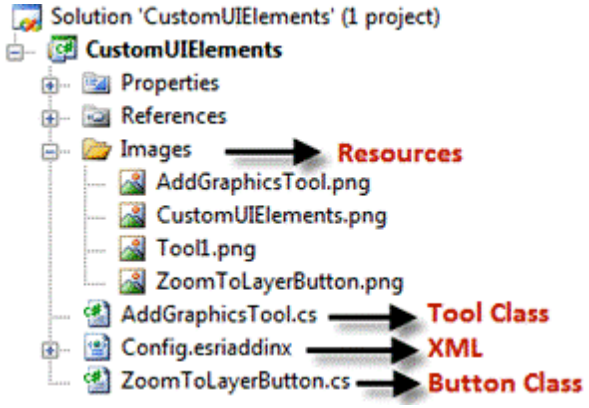
The ArcGIS Add-Ins Wizard creates appropriate XML file and managed classes required for the add-in based on user input:

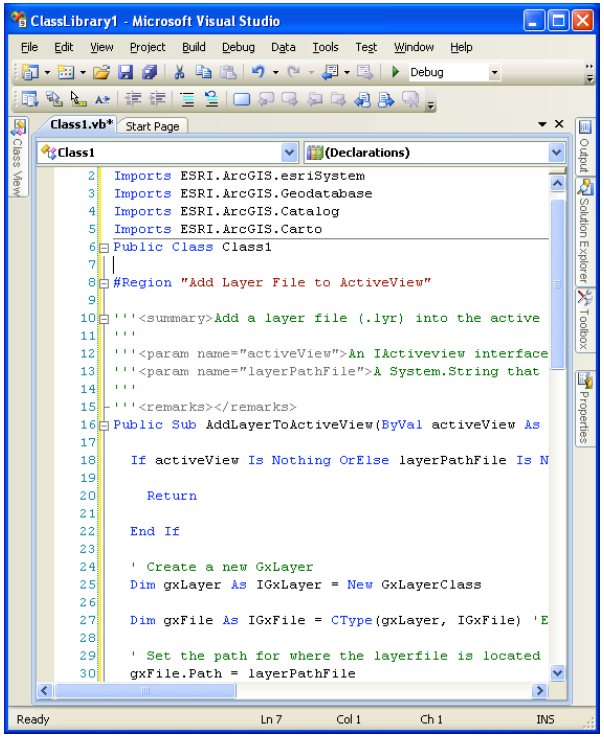




Creating Add-ins w/ Visual Studio

With info filled in, the Wizard generates the VS project structure, stores the XML config file, the picture resources. The project is ready to accept code in the “class” objects:





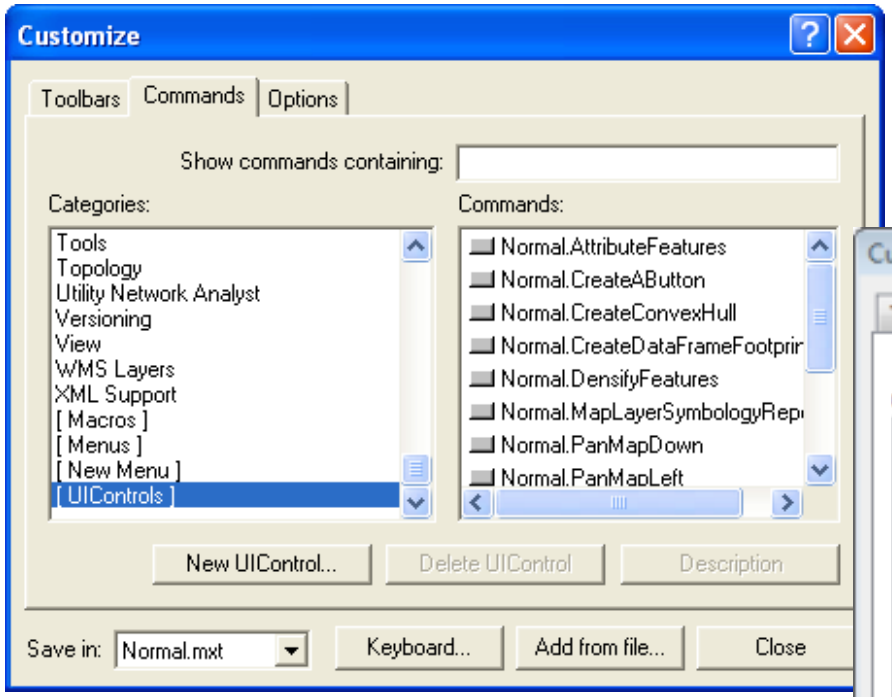
Creating Add-ins w/ Visual Studio

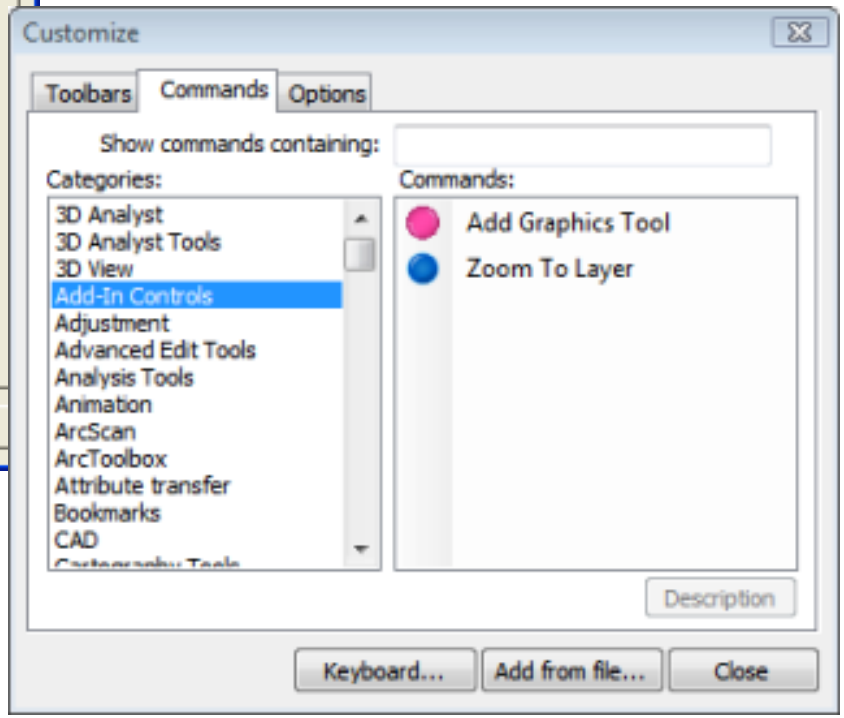
When finished with code:

- Build Solution:
  - If the project builds correctly, the necessary files for customization are bundled as an add-in file (Zip file, but with the .ESRIAddIn file extension) and installed using the ESRIRegasm utility
  - These steps are conducted automatically by a post-build MSBuild task included by the wizard in your project.
- The add-in is now ready to add to the application

Add-ins vs. Old UIControls Dialog

Old Customize dialog with “UIControls”



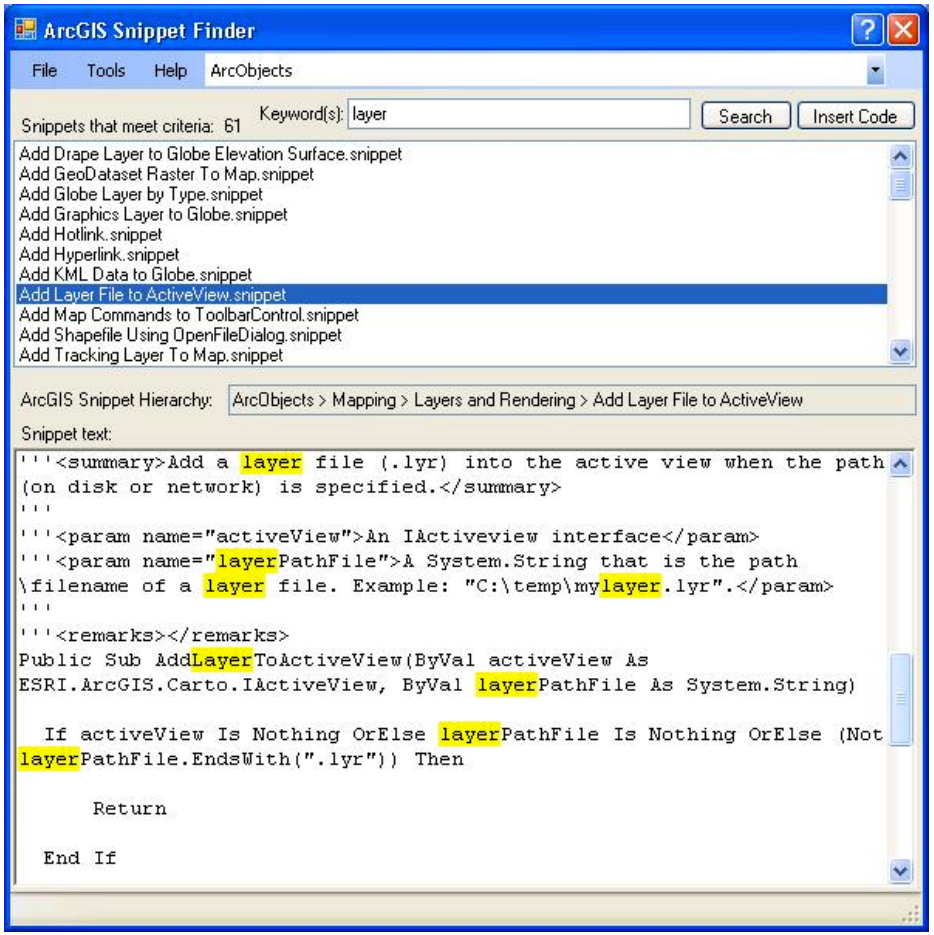


Commands are still added to a toolbar or menu by drag-and-drop

New Customize dialog with “Add-In Controls”

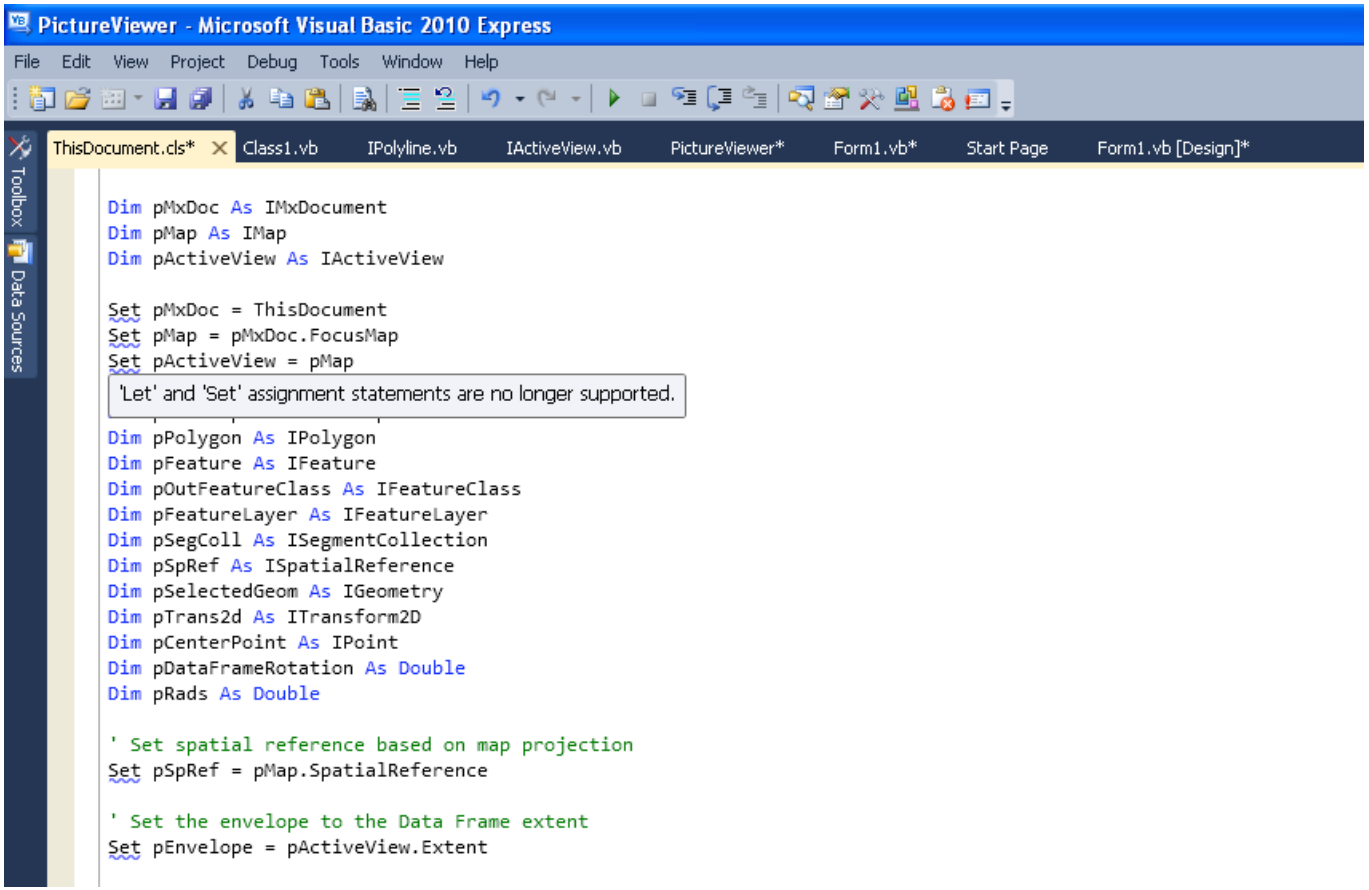
The hard part: migrating code

- Snippets: rebuild your automation using canned code
- Many common functions have been rewritten in .NET



The hard part: migrating code

- Open old code in VS and debug using context-sensitive helpers:

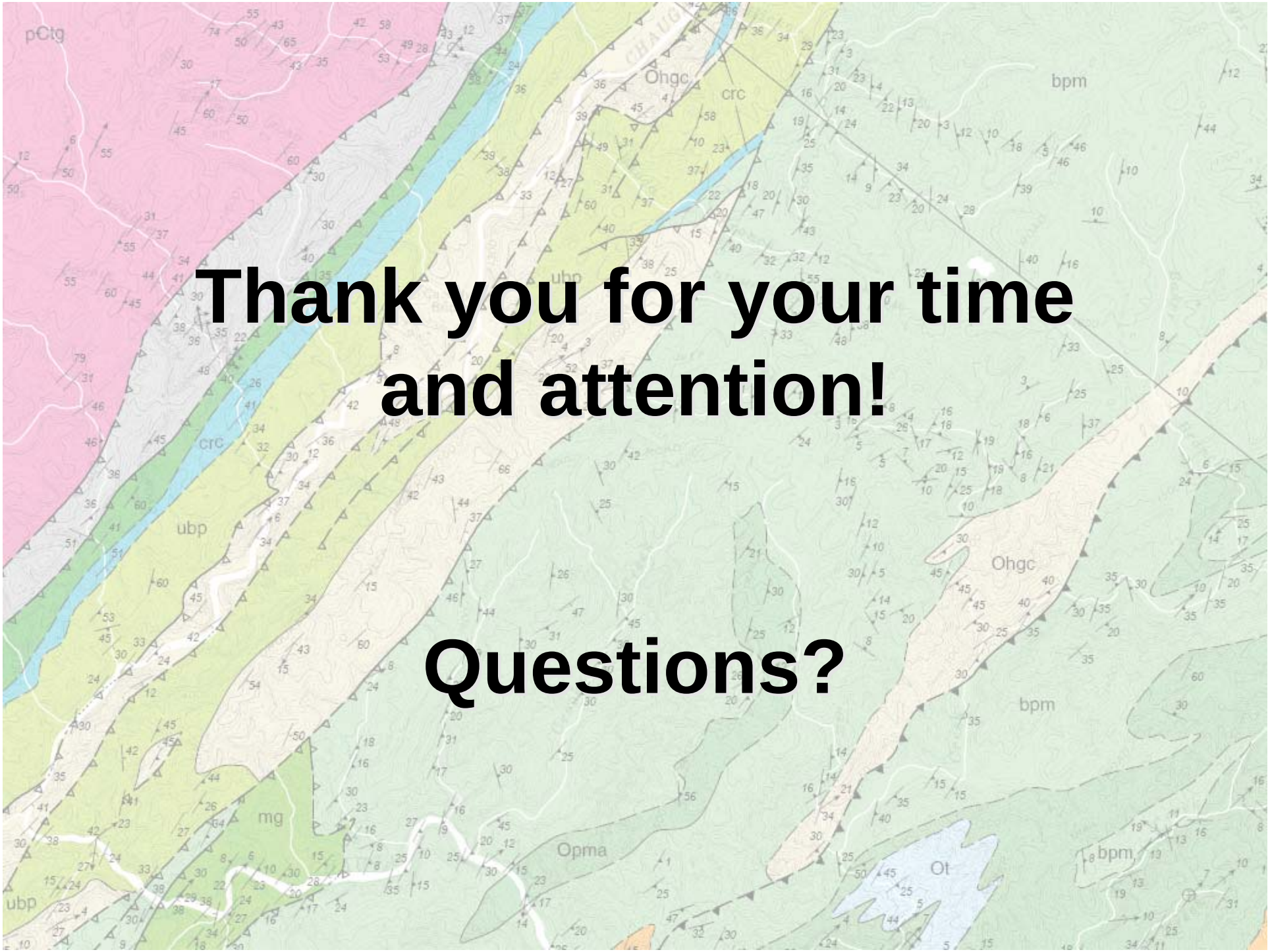


Resources for migration to .NET

- ArcGIS Desktop help (on the web)
- Visual Studio 2010 help (on the web)
- ArcGIS Resources Centers for .NET developers (on the web)
- The new ArcGIS Code Gallery (formerly ArcScripts on the web)

Final Thoughts…

- Try to use Python for geoprocessing automations:
  - Gentle learning curve
  - Lots of free resources for help
- Use Add-ins and take advantage of wizards and code snippets
- Plan the migration well, use the ArcGIS resources pages to help with the process
- Try to redo as little as possible, but understand that some things will have to be redone…



Thank you for your time and attention!

Questions?