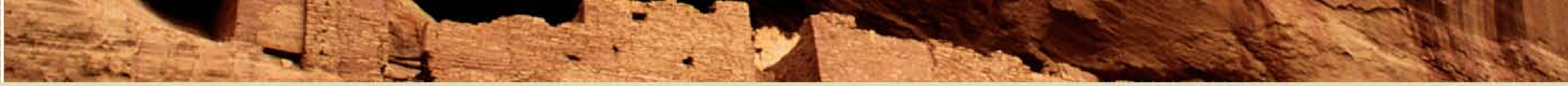
Deploying Registration-Free COM Controls

Dan Levene
dlevene@anasazisoftware.com
Chief Technology Officer
Anasazi Software, Inc.
Phoenix, Arizona USA

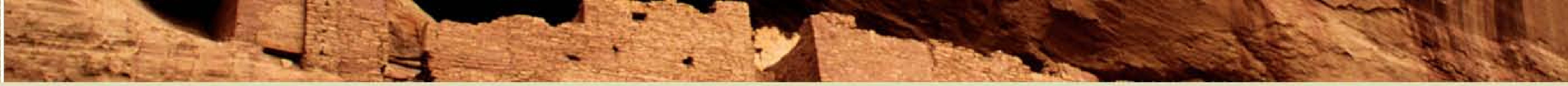
Deploying Registration-Free COM Controls

- **What is Registration-free COM?**
  - A method to deploy and use COM controls and automation objects without having to register them to Windows (via regsvr32.exe, commercial installers such as InstallShield or Wise, or custom installers).
  - Registration-free COM is an informal reference to Microsoft's Isolated Applications and Side-by-side Assemblies Solution documented in the MSDN Windows Development Library.
  - Available in all Windows operating systems from XP forward.
  - Specifically, what I am going to focus on are the techniques of deploying Isolated Applications.



Deploying Registration-Free COM Controls

- **Why Registration-free COM?**
  - Eliminate conditional locating of COM control files.
  - Eliminate the need for administrator privileges to install COM controls.
  - Simplify installation process architecture.
  - Increase the portability of installed applications.
  - Force application to use a resource from a specified location (relative to the executable's path) and from nowhere else.
    - If the manifest is incorrect, Windows fails the executable launch with a message.
    - If the resource is missing from the specified location, the app functionality that depends upon it will fail.
    - In other words, if installed correctly the app will work, if not, the app will fail positively—no in between mysterious behavior because a component was found in a directory unrelated to the intended installation.



Isolated Applications

- **From Microsoft's Introduction to Isolated Applications and Side-by-side Assemblies:**
  - **Purpose**
    - Isolated Applications and Side-by-side Assemblies is a Microsoft Windows solution that reduces versioning conflicts in Windows-client applications. With Windows, application developers can build isolated applications that are fully self-describing and unaffected by changes to the registry, other applications, or other versions of assemblies running on the system. Application authors and administrators can use manifests to manage the sharing of side-by-side assemblies after deployment on either a global or per-application basis. Customers benefit from isolated applications that are more stable and more reliably updated.

  - http://msdn.microsoft.com/en-us/library/dd408052(v=VS.85).aspx

ANASAZI SOFTWARE®

Isolated Applications

- **From Microsoft's Introduction to Isolated Applications and Side-by-side Assemblies (continued):**
  - **Where Applicable**
    - Isolated applications and side-by-side assembly sharing can be used to develop applications that safely share operating system assemblies. Developers can use this technology to correct DLL versioning conflicts caused by an incompatible version of a shared assembly.
    - If your application must consistently get the version of a component you have tested, it is possible to isolate your application so that it will always be run with the tested version of the component on the user's computer.

  - http://msdn.microsoft.com/en-us/library/dd408052(v=VS.85).aspx

Isolated Applications

- **Highlights of the Microsoft Solution's Documentation**
  - **Isolated Applications**
    - Total isolation is achieved when everything an application executable needs to run is in the directory where the executable is found
    - A totally isolated application can be deployed with a simple file system copy of the directory (xcopy).
    - A partially isolated application is one that insists on finding certain components in the specified directory while allowing other components to be located elsewhere and discovered through Windows normal search path.
    - Registration-free COM is a partial isolation solution targeted at COM controls.

ANASAZI SOFTWARE®

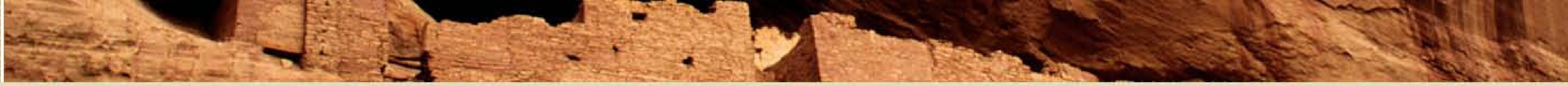
Side-by-Side Assemblies

- **Highlights of the Microsoft Solution's Documentation (continued)**
  - **Side-by-Side Assemblies**
    - A side-by-side assembly is an advanced method of storing and registering a set of version-compatible components under the Windows system installation directory. The purpose is to allow different version sets of components to be registered globally and referenced as necessary by individual applications.
    - Microsoft deploys a handful of standard side-by-side assemblies that are available to all executables. The most commonly used one is the Shell Common Controls version 6.0 (Comctl32.dll) assembly.
    - For some time now, Visual Dataflex application manifest files have included a dependency node reference to the Common Controls 6.0 assembly.

ANASAZI SOFTWARE®

Isolated Applications

- **Highlights of the Microsoft Solution's Documentation (continued)**
  - **Manifest Documentation**
    - Manifests are XML documents
    - Description of allowed elements and attributes.
    - Schema (careful, this is Microsoft's own schema format, not an xsd).
    - Editorial: This documentation is a bit sparse and does not describe the intricate workings of manifests in great detail, but it does provide a high level picture of the breadth and depth of functionality available through use of manifests.



Deploying Registration-Free COM Controls

- **How is Registration-free COM achieved?**
  - By including in the Application Manifest the information for all referenced COM objects that is typically stored in the registry when the COM control hosting the objects is installed and registered. (i.e., regsvr32.exe)
  - And, by deploying the .dll or .ocx file in the location the manifest specifies relative to the executable's location. Typically, this is simply the executable's directory.

ANASAZI SOFTWARE®

Deploying Registration-Free COM Controls

- **How is Registration-free COM achieved?**

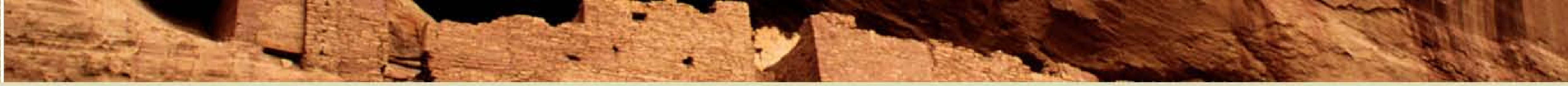
  On a machine that has the control conventionally registered (a developer's machine, typically):

  1. For a given imported COM package, append a <file> element to the manifest with a name attribute equal to the name of the dll or ocx file.
  2. For each "Set psProgID" line in a COM control's imported class package, locate the defined GUID in HKEY_CLASSES_ROOT\CLSID in the Registry.
  3. If found, create a sub-element <comClass> and populate its allowed attributes with corresponding information from the Registry.
  4. If there is an associated Typelib GUID in the Registry, create a sub-element of the <file> element named <typelib> and populate appropriately.
  5. Deploy the COM control (.dll, .ocx, etc.) in the same directory as the executable.

ANASAZI SOFTWARE®

Anasazi Manifest Modifier Utility

- Anasazi has created a utility that updates an application's manifest file with appropriate entries.
- The Manifest Modifier has two modes:
  - In its first mode, it examines .pkg files created by the COM Class Generator when importing COM controls in the Studio and creates a fragment manifest file with the same root name as the .pkg. In this fragment, it creates appropriate <file> entries that contain the same COM definition information that is typically placed in the Windows registry.
  - In its second mode, it merges selected .pkg.manifest files into a targeted application manifest file.

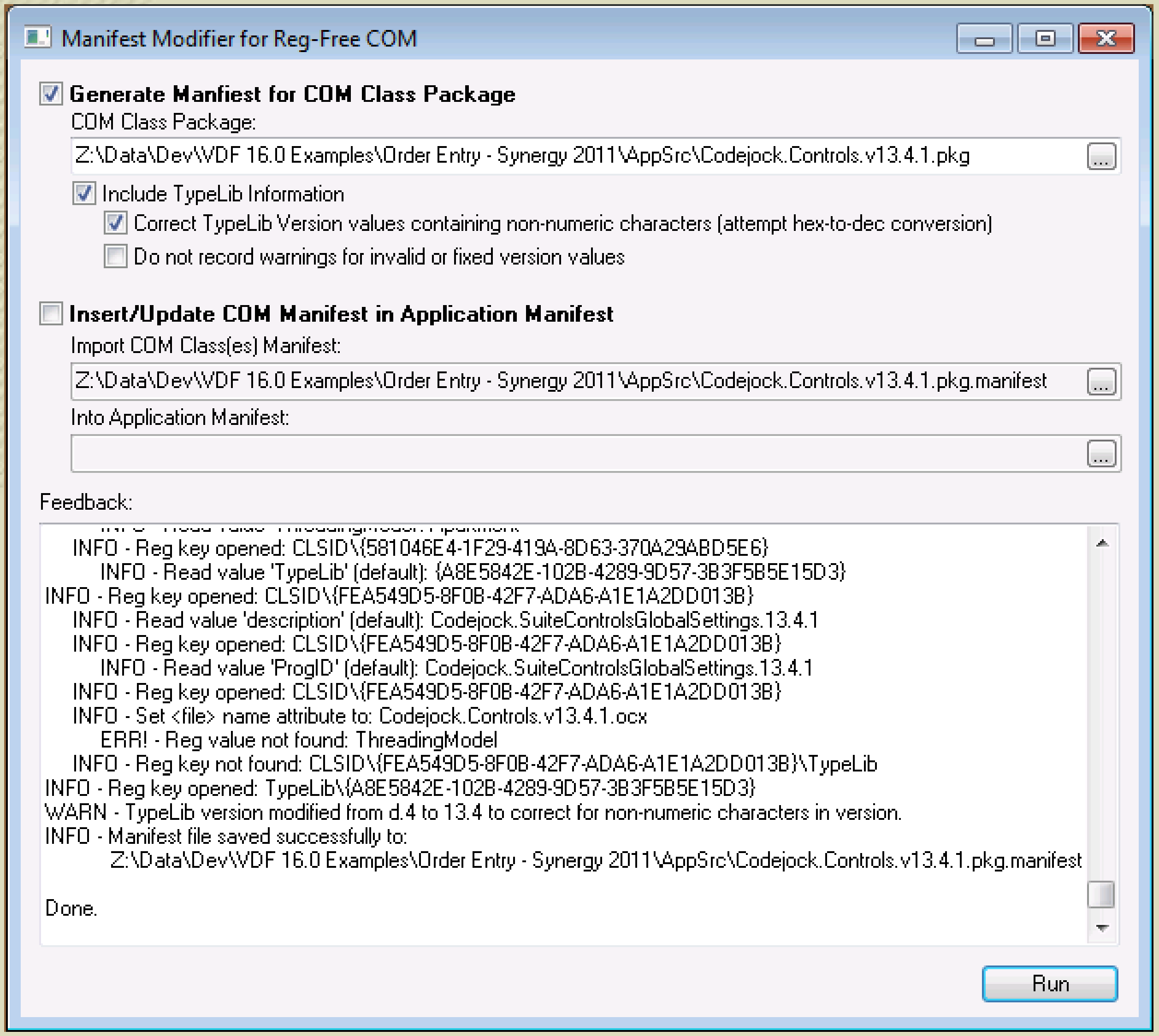


Anasazi Manifest Modifier Utility

- Any time a new or updated COM import package is created use the Manifest Modifier to:
  - Create a fragment manifest file in sync with the COM package.
  - Merge the fragment manifest into all application manifests that use the package’s COM controls. If the COM control’s entries already exist in the application manifest, they will be updated; otherwise, they will be appended as new.
- No need to regenerate or merge each time a VDF application is compiled. The VDF compiler/linker will simply update relevant information in the manifest and ignore the isolation entries.

ANASAZI SOFTWARE®

Anasazi Manifest Modifier Utility



Manifest Modifier for Reg-Free COM

[✓] **Generate Manfiest for COM Class Package**

COM Class Package:
Z:\Data\Dev\VDF 16.0 Examples\Order Entry - Synergy 2011\AppSrc\Codejock.Controls.v13.4.1.pkg

[✓] Include TypeLib Information

[✓] Correct TypeLib Version values containing non-numeric characters (attempt hex-to-dec conversion)

[ ] Do not record warnings for invalid or fixed version values

[ ] **Insert/Update COM Manifest in Application Manifest**

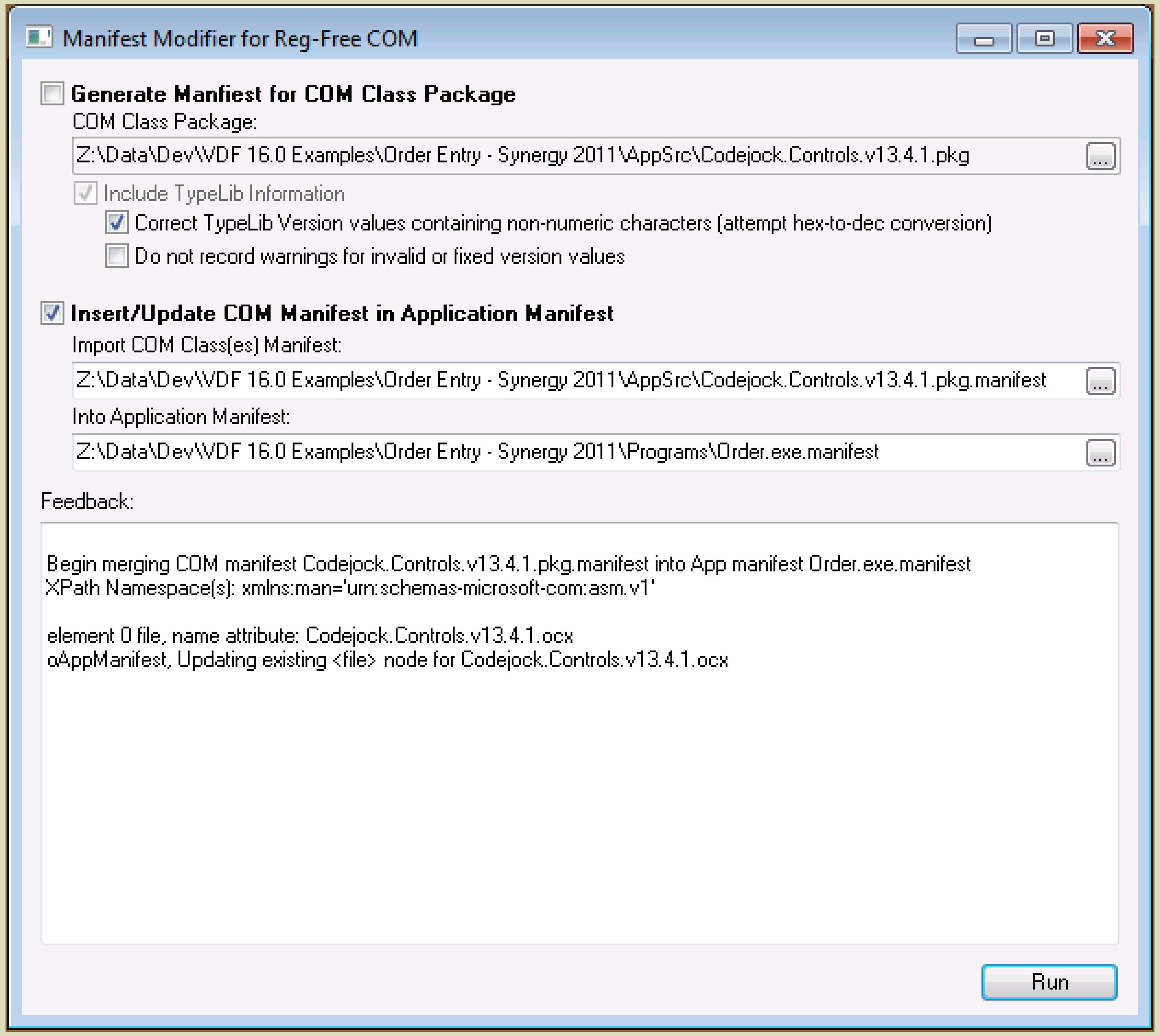
Import COM Class(es) Manifest:
Z:\Data\Dev\VDF 16.0 Examples\Order Entry - Synergy 2011\AppSrc\Codejock.Controls.v13.4.1.pkg.manifest

Into Application Manifest:

Feedback:

```
    INFO - Reg key opened: CLSID\{581046E4-1F29-419A-8D63-370A29ABD5E6}
        INFO - Read value 'TypeLib' (default): {A8E5842E-102B-4289-9D57-3B3F5B5E15D3}
INFO - Reg key opened: CLSID\{FEA549D5-8F0B-42F7-ADA6-A1E1A2DD013B}
    INFO - Read value 'description' (default): Codejock.SuiteControlsGlobalSettings.13.4.1
    INFO - Reg key opened: CLSID\{FEA549D5-8F0B-42F7-ADA6-A1E1A2DD013B}
        INFO - Read value 'ProgID' (default): Codejock.SuiteControlsGlobalSettings.13.4.1
    INFO - Reg key opened: CLSID\{FEA549D5-8F0B-42F7-ADA6-A1E1A2DD013B}
    INFO - Set <file> name attribute to: Codejock.Controls.v13.4.1.ocx
        ERR! - Reg value not found: ThreadingModel
    INFO - Reg key not found: CLSID\{FEA549D5-8F0B-42F7-ADA6-A1E1A2DD013B}\TypeLib
INFO - Reg key opened: TypeLib\{A8E5842E-102B-4289-9D57-3B3F5B5E15D3}
WARN - TypeLib version modified from d.4 to 13.4 to correct for non-numeric characters in version.
INFO - Manifest file saved successfully to:
         Z:\Data\Dev\VDF 16.0 Examples\Order Entry - Synergy 2011\AppSrc\Codejock.Controls.v13.4.1.pkg.manifest

Done.
```



Manifest Modifier for Reg-Free COM

[ ] **Generate Manfiest for COM Class Package**

COM Class Package:
Z:\Data\Dev\VDF 16.0 Examples\Order Entry - Synergy 2011\AppSrc\Codejock.Controls.v13.4.1.pkg

[✓] Include TypeLib Information

[✓] Correct TypeLib Version values containing non-numeric characters (attempt hex-to-dec conversion)

[ ] Do not record warnings for invalid or fixed version values

[✓] **Insert/Update COM Manifest in Application Manifest**

Import COM Class(es) Manifest:
Z:\Data\Dev\VDF 16.0 Examples\Order Entry - Synergy 2011\AppSrc\Codejock.Controls.v13.4.1.pkg.manifest

Into Application Manifest:
Z:\Data\Dev\VDF 16.0 Examples\Order Entry - Synergy 2011\Programs\Order.exe.manifest

Feedback:

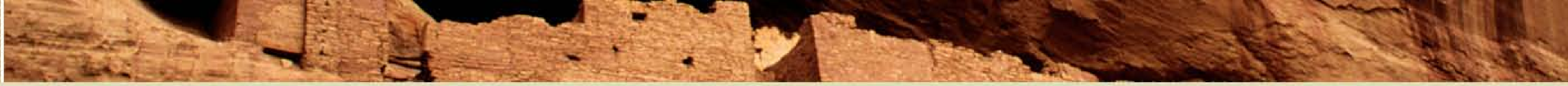
```
Begin merging COM manifest Codejock.Controls.v13.4.1.pkg.manifest into App manifest Order.exe.manifest
XPath Namespace(s): xmlns:man='urn:schemas-microsoft-com:asm.v1'

element 0 file, name attribute: Codejock.Controls.v13.4.1.ocx
oAppManifest, Updating existing <file> node for Codejock.Controls.v13.4.1.ocx
```

Final Comments on Registration-Free COM

- Not all COM controls can be registered in an application manifest.
  - Controls that are managed by Windows as a registered side-by-side assembly should not be referenced in a manifest as an isolated module. For instance, though not a COM control, Microsoft's Common Controls v6 must be referenced as a <dependency><dependentAssembly> not as a <file> isolated reference. Attempting to do so will cause Windows to fail application launch because it knows this file has a public side-by-side registration.
  - Not all developers of COM controls follow the rules to the letter when creating COM entries in the Registry. Creating the manifest <file> entries may fail or the entries may need to be altered manually in order to get the control to behave in isolation. Let trial and error be your guide.

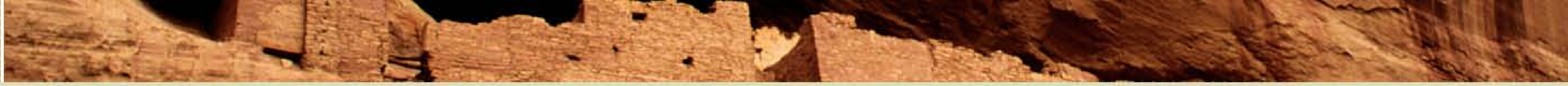
ANASAZI SOFTWARE®



Bonus: Embedded Manifest

- **Microsoft command line tool: MT.exe**
  - Included with Windows SDK or Microsoft Visual Studio development environment.
  - Many options for creating, merging and manipulating manifests
  - Syntax to embed manifest in executable:
    - MT.exe –manifest <MyApp.exe.manifest> -outputresource:<MyApp.exe>

  - Once embedded:
    - The .manifest file does not need to be deployed.
    - An external manifest is ignored and has no effect on the application’s behavior.
  - Caveat: Anasazi has not tested that an executable with an embedded manifest works in all operating system or deployment environments and does not embed their own manifests at this time. Be sure to test in any desired target environments.



Thank you

For further information, to obtain a copy of this presentation or the Manifest Modifier Utility,

Feel free to contact me:

Dan Levene

dlevene@anasazisoftware.com

Chief Technology Officer

Anasazi Software, Inc.

Phoenix, Arizona USA

480 598-8833

ANASAZI SOFTWARE®