Agenda

- ArcGIS as a System
- SDK requirements at 10, 10.1, and 10.2
- Product name changes
- .NET 4.0 support
- 64 bit support
- Publisher Policy Files
- Component Registration
- VBA
- ArcGIS Runtime for...
ArcGIS 10 — A Complete System

Easier
More Powerful
and Everywhere

• Discover
• Create
• Manage
• Visualize
• Analyze
• Collaborate

Cloud
Enterprise
Local

Web
Mobile
Desktop
## Product Name Changes

- **Changes in the esriProductCode enumeration**

<table>
<thead>
<tr>
<th>10.0</th>
<th>10.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>esriProductCodeArcView</td>
<td>esriProductCodeBasic</td>
</tr>
<tr>
<td>esriProductCodeArcEditor</td>
<td>esriProductCodeStandard</td>
</tr>
<tr>
<td>esriProductCodeArcInfo</td>
<td>esriProductCodeAdvanced</td>
</tr>
</tbody>
</table>

- **Changes in the esriLicenseProductCode enumeration**

<table>
<thead>
<tr>
<th>10.0</th>
<th>10.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>esriLicenseProductCodeArcView</td>
<td>esriLicenseProductCodeBasic</td>
</tr>
<tr>
<td>esriLicenseProductCodeArcEditor</td>
<td>esriLicenseProductCodeStandard</td>
</tr>
<tr>
<td>esriLicenseProductCodeArcInfo</td>
<td>esriLicenseProductCodeAdvanced</td>
</tr>
</tbody>
</table>
ArcGIS for Desktop .NET SDK requirements

10.0

- Visual Studio 2008 sp1
- VS Express 2008
- Visual Studio 2010
- .NET Framework – 3.5 sp1
- ArcGIS Requirements
  - ArcGIS Engine Runtime
  - ArcObjects .NET SDK
  - ArcGIS Engine Developer Kit License
ArcGIS for Desktop .NET SDK requirements

10.1

- Visual Studio 2010
- Visual Studio 2010 Express
- .NET Framework – 3.5 sp1
- ArcGIS Requirements
  - ArcGIS Engine Runtime
  - ArcObjects SDK
  - ArcGIS Engine Developer Kit License
What is needed to Develop Engine Applications?

With 10.2

- Visual Studio 2012
- Visual Studio 2010
- Visual Studio 2010 Express
- .NET Framework – 3.5 sp1

- ArcGIS Requirements
  - ArcGIS Engine Runtime
  - ArcObjects .NET SDK
  - ArcGIS Engine Developer Kit License
.NET Framework

- .NET Framework 3.5 is min requirement
  - Guaranteed to be installed with Engine or Desktop

- .NET Framework 4.0 is fully supported
  - Need to handle distribution
  - Desktop developer Add-ins install without Admin requirements

- Do NOT embed interop types
64 Bit Support

- ArcGIS Desktop and Engine are 32 bit applications
  - Run as 32 bit applications on a 64 Bit OS
- For Standalone applications
  - Set platform to x86 in Visual Studio Configuration Manager
  - Default is “Any CPU”
- At Version 10 ArcGIS applications are Large Address Aware
  - On 64 Bit OS processes can take up to 4 gigabytes of RAM if available
No Publisher Policy Files at ArcGIS 10

- Policy files specify assembly redirection
  - Installed to the GAC
  - Included in previous versions of ArcGIS

- Desktop applications include an Application Configuration file
  - Redirects 9.3.x Assemblies to 10
  - There for migration assistance only
  - Recommended to recompile
Component Registration
Component Registration

Understanding component registration

- Prior to 10, ArcGIS knew which customizations to load based on which categories a component was registered in

- Example Custom ArcMap Command
  - Implement ICommand
  - Register COM component
  - Register command in MxCommands Category
  - ArcMap reads and loads all components from the MxCommands Category
Component Registration at ArcGIS 9.x

Understanding component registration

MxCommands

COM

ICommand

ArcMap.exe

My.dll
Component Registration at ArcGIS 10.x

Understanding component registration

- ArcGIS 10 no longer reads component categories from the registry

- Component category information is read from configuration files (*.ecfg)
Component Registration at ArcGIS 10

Understanding component registration at 10

ECFG
MxCommands
ArcMap.exe
My.dll
Registry
COM
ICommand
EsriRegasm
Component Registration

How to register custom components at 10.x

- Components are still COM based
  - Must register
- Use ESRIRegAsm.exe to register category information
  - Creates your *.ecfg file in the appropriate folder

Engine:

%CommonProgramFiles%\ArcGIS\Engine10.2\Configuration\CATID

Desktop:

%CommonProgramFiles%\ArcGIS\Desktop10.2\Configuration\CATID
Component Registration

Adding registration to existing projects

• Adding ESRIRegAsm.exe step to your .NET projects
  - Unload and Edit project
  - Add custom build step
  - This registers your component on build and un-registers on clean

• Visual C++ developers add custom Post-Build Event
  “$(CommonProgramFiles)\ArcGIS\bin\esriregasm.exe” /p:Desktop
  “$(TargetPath)”
Component Registration

How to deploy custom components

- To Deploy little change needs to be made
  - Option 1:
    - If you call RegSvr32.exe or RegAsm.exe
    - Call ESRIRegAsm.exe instead
  - Option 2:
    - Register like you always have with COM and copy a pre-generated *.ecfg file to the appropriate location

Engine:
%CommonProgramFiles%ArcGIS\Engine10.2\Configuration\CATID

Desktop:
%CommonProgramFiles%ArcGIS\Desktop10.2\Configuration\CATID
Migration

Tips for migration workflow

- ArcObjects SDK includes the Code Migration Analyzer
  - Provides warnings to assist with the migration process
- Update debug start action to new ArcGIS executable locations
- Fix any changes to Assemblies (specific version false)
- Target x86
- Add new component registration step
- Standalone applications add the Binding code
- Update Desktop License
Migrating Custom Components
VBA Migration
VBA Migration

Status for 10.2

- Free VBA Compatibility Setup
  - Request License
  - No Support or SDK
  - Available to allow existing solutions to still work during migration

- Esri urges you to migrate VBA applications to a supported development environment

- Where should a VBA developer go?
VBA Migration

Add-ins

• Why Migrate to Add-ins?
  - A lot of code can be copy/pasted in a VB.NET application
  - Types of customizations are the same or similar
  - Simple deployment model
Add-ins

• No COM registration required!
• Can use Java, .NET, or Python
• Create Add-in using a Wizard
  - Config.esriAddins.xml file and a class
  - New items can be created through wizard or through XML
• Compile Add-In
  - Creates an *.esriaddin file
  - Xcopy deployable
  - Installation is as simple as double-clicking
Add-Ins

- Buttons
- Tools
- Combo Boxes
- Multi-Items
- Menus
- Context Menus
- Toolbars
- Tool Palettes

- Dockable Windows
- Application Extensions
- Editor Extensions
- Editor Construction Tools
- SOE (Server)
Add-Ins

- Add-ins are installed on a per user basis
- No administrative permissions required
  - Can be digitally signed
  - Installation can be controlled by systems administrator
    - Block all Add-ins, block unsigned Add-ins, specify administrator folder, only allow Esri Add-ins
- Add-In can be placed on a network share and referenced through Add-In Manager
  - All updates will automatically be picked up next time ArcGIS is loaded
Migrate VBA to Add-In

```
Private m_pWork
Private m_s5erv
Private m_oInst
Private m_sUser
Private m_sPass
Private m_sAuth
Private wf As lng

Public Sub test()
Dim pMxDoc As IMxDocument
Dim pMap As IMap
Dim pActiveView
Set pMxDoc = Th.
Set pMap = pMxDoc.
Set pActiveView.
Dim player As Integer
Set player = pMxDoc.
Dim pFdataset As Table
Set pFdataset = pFdataset.
MsgBox pFdataset.
pFdataset.Delete.
MsgBox player.F
End Sub
Private Sub Log
```
Support issues
Debugging and .NET 4.0

- Update the *.exe.config file
  - C:\Program Files (x86)\ArcGIS\Desktop10.2\bin\ArcMap.exe.config

- Attach to process
IClassSchemaEdit4.AlterFieldName

- Allows developers to rename existing fields

```csharp
private static void renameField(ITable table, string oldField, string newField)
{
    ISchemaLock schemaLock = (ISchemaLock)table;
    IClassSchemaEdit4 classSchemaEdit = (IClassSchemaEdit4)table;
    try
    {
        schemaLock.ChangeSchemaLock(esriSchemaLock.esriExclusiveSchemaLock);
        classSchemaEdit.AlterFieldName(oldField, newField);
    }
    catch (Exception)
    {
        Console.WriteLine("Cannot get exclusive schema lock.");
        return;
    }
    finally
    {
        schemaLock.ChangeSchemaLock(esriSchemaLock.esriSharedSchemaLock);
    }
}
```
IGPToolBackground

- Can force the tool to run in foreground only
- Implement interface
- Return false
Some new interfaces at 10.1

- **ILabelClassDescription2.ExpressionType**
  - Starting at 10.1 you can get or set the expression type

- **IWorkspaceDomains3.AlterDomainWithName**
  - Starting at 10.1 you can rename domains using this

- **ICodedValueDomain2.SortByName or SortByValue**
  - Starting at 10.1 you can sort based on names or values
ArcGIS Runtime
ArcGIS Runtime

- New Architecture
- Easy to Deploy Solutions
  - Software, maps and data
- 64 bit Native Support
- Modern API
  - WPF and Java
  - OSx and QT (Coming)
- Reduced Memory and Disk Footprint
ArcGIS Runtime

- Not a functional equivalent to Engine
- Not an ArcObjects replacement
- New APIs
  - Code will need to be re-written
  - Should leverage general patterns for data sharing and distribution
Leveraging Packages

What are packages?

- Packages are a single file that contains a map or layer[s] and supporting data
  - Also can contain references to SDE data
- Easy to share
  - Single file
- ArcGIS 10.0 supports
  - Layer Packages (*.lpk)
  - Map Packages (*.mpk)
- ArcGIS 10.1 adds additional support for
  - Geoprocessing Packages (*.gpk)
  - Locator Packages (*.apk)
  - Tile Packages (*.tpk)
Leveraging Packages

Using Packages in ArcGIS for Desktop

- **Starting at ArcGIS 10 sp1**
  - Programmatically consume packages

- **IMapDocument.Open**
  - Map Packages
  - Layer Packages
  - Web Maps

- **ILayerFile.Open**
  - Layer Packages

- Just point to the path of the package
Working with Packages

Using Packages in ArcGIS for Desktop

- Opening Layer Packages

```vba
Dim layerFile As ILayerFile = New LayerFileClass
layerFile.Open("c:\Data\LayerPackages\USCities.lpk")
Dim layer As ILayer = layerFile.Layer
axMapControl1.AddLayer(layer)
```

- Opening Map Packages

```vba
Dim mapDocument As IMapDocument = New MapDocumentClass
mapDocument.Open("c:\Data\LayerPackages\MyMapPackage.mpk", ",")
axMapControl1.Map = mapDocument.get_Map(0)
```
Working with Packages

Using Online Content

- **Consume data on ArcGIS Online**
  - Pass in a URL with the id as the filename
    
    ```
    http://www.arcgis.com/home/item.html?id=a50c645f83bb4f5dbddd457df29d639
    ```

```csharp
private void addLayerFromOnlinePackage(string packageId)
{
    string lyrPackage = "http://www.arcgis.com/sharing/content/items/" + packageId + "/item.pinfo";
    ILayerFile layerFile = new LayerFileClass();
    layerFile.Open(lyrPackage);
    ILayer layer = layerFile.layer;
    axMapControl1.AddLayer(layer);
}
```
Working with Packages

Using IPackageFile at 10.1

• Add reference to:
  ESRI.ArcGIS.EngineCore

• Unpack using IPackageFile.Unpack

```csharp
private void unPack(string packagePath, string unpackToLocation)
{
    IPackageFile packageFile = new PackageFile();
    IStringArray stringArray = packageFile.Unpack(packagePath, ref unpackToLocation);
    axMapControl1.AddLayerFromFile(stringArray.get_Element(0));
}
Creating Packages for Runtime

- Enable Runtime tools
- Support ArcGIS Runtime
Leveraging packages in Runtime

• WPF example

```xml
<esri:Map x:Name="MyMap" UseAcceleratedDisplay="True">
    <!-- Local Tiled Basemap Layer -->
    <esri:ArcGISLocalTiledLayer ID="Topographic USA"
        Path="C:\TPKs\Topographic.tpk"/>

    <!-- Local Dynamic Layer -->
    <esri:ArcGISLocalDynamicMapServiceLayer ID="USA"
        Path="C:\MPKs\USCitiesStates.mpk"/>

    <!-- Local Feature Layer -->
    <esri:ArcGISLocalFeatureLayer ID="arcGISLocalFeatureLayer"
        Path="C:\MPKs\USCitiesStates.mpk"
        LayerName="States"/>

</esri:Map>
```
ArcGIS Runtime

- **New APIs**
  - Code will need to be re-written
  - Should leverage general patterns for data sharing and distribution
- **Not a functional equivalent to Engine**
- **Not an ArcObjects replacement**
Consume SOE in Runtime
Questions?