Using the ArcGIS Engine Controls in .NET

Mike Rudden and Mary Harvey
Overview

• Extend with custom components

• Persisting Application State

• Tips & Tricks
ArcGIS Engine SDK Overview

- **APIs**
  - COM, **.NET**, Java and C++

- **Components**
  - Controls library: **2D Controls**, 3D Controls, Framework Controls, Controls Commands
  - ArcObjects libraries: Carto, Geodatabase, Geometry...

- **Tools**
  - Documentation
  - **VS2005 Integration**
ArcGIS Engine 9.2 Visual Studio 2005 Integration

• **Solution / Project Level**
  – ArcGIS Code Converter
  – Project Templates

• **Project Level**
  – Add Class Wizard
  – Add ArcGIS Reference
  – ArcGIS Toolbox Reference

• **Class Level**
  – Component Category Registrar
  – ArcGIS License Initializer
  – Snippets
ArcGIS Engine 9.2 Licensing

- You must ‘explicitly’ license an application:
  - Using the “ArcGIS License Initializer” wizard
  - Using the LicenseControl
  - N.B. ‘Shutdown the application’ checkbox

Demo
Extending ArcGIS Engine

• The framework can be extended by writing custom COM components:
  – Commands / Tools
  – Menus / Palettes / Pop-up Menus
  – Extensions
Implementing Custom COM Components

1. Create a .NET project
2. Create a class
3. Reference the appropriate assemblies
4. Implement an interface
5. Compile and register DLL
6. Register with a component category

• Base classes provided in ESRI.ArcGIS.ADF.BaseClasses
  – BaseCommand, BaseTool, BaseMenu...

```csharp
public override void OnCreate(object hook)
{
    if (hook == null)
        return;
}
```
The HookHelper Class

- The hook may be many different objects:
  - MapControl, ToolbarControl, MxApplication...

- HookHelper takes care of the type of hook

- IHookHelper provides useful properties...

<table>
<thead>
<tr>
<th>IHookActions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IHookActions : IUnknown</td>
<td></td>
</tr>
<tr>
<td>DoAction (in pUnknown: IUnknown, in Action: esriHookActions)</td>
<td></td>
</tr>
<tr>
<td>DoActionOnMultiple (in pArray: IArray, in Action: esriHookActions)</td>
<td></td>
</tr>
<tr>
<td>DoActionWithName (in pUnknown: IUnknown, in Name: String, in Action: esriHookActions)</td>
<td></td>
</tr>
<tr>
<td>DoActionWithNameOnMultiple (in pArray: IArray, in pNamesArray: IStringArray, in Action: esriHookActions)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IHookHelper</th>
</tr>
</thead>
<tbody>
<tr>
<td>IHookHelper : IUnknown</td>
</tr>
<tr>
<td>ActiveView: IActiveView</td>
</tr>
<tr>
<td>FocusMap: IMap</td>
</tr>
<tr>
<td>Hook: IDispatch</td>
</tr>
<tr>
<td>OperationStack: IOperationStack</td>
</tr>
<tr>
<td>PageLayout: IPageLayout</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IHookHelperEvents</th>
</tr>
</thead>
<tbody>
<tr>
<td>IHookHelperEvents : IUnknown</td>
</tr>
<tr>
<td>OnHookUpdated (in hookEvent: esriHookHelperEvents)</td>
</tr>
</tbody>
</table>
HookHelperClass

```csharp
public override void OnCreate(object hook)
{
    if (m_hookHelper == null)
    
        m_hookHelper = new HookHelperClass();

    m_hookHelper.Hook = hook;

    m_hookHelper.ActiveView......

• Also GlobeHookHelper...

Demo
```
HookActions

• Performs common actions on simple geometries.
  – Is action supported on geometry
  – Do action on geometry using esriHookActions constants:
    • esriHookActionsFlash
    • esriHookActionsPan
    • esriHookActionsZoom ...

• GlobeHookHelper also implements IHookActions

Demo
Creating ToolbarMenus

• **Types**
  - Add to ToolbarControl
  - Popup menus
  - Context menu to work with TOCControl
  - Sub menus

• **Steps**
  - Create ToolbarMenu
  - Add Commands or Tools or IMenuDef
  - Share CommandPool
Creating Toolbar Palettes

- **Types**
  - Add to ToolbarControl
  - Popup palette

- **Steps**
  - Create ToolbarPalette
  - Add Commands or Tools or IPaletteDef
  - Share CommandPool

- **3 out-of-the-box palettes provided:**
  - Erazer
  - Pen
  - Highlighters

Demo
Implementing COM extensions

- Extensions can be used to share data between software components

- Interfaces: IExtension, IExtensionConfig

- Type Library: esriSystem

- State of commands is controlled by state of extension

Public Class ZoomExtension
  Implements IExtension
  Implements IExtensionConfig

Demo
Runtime Customization & Persistence

• ToolbarControl CustomizeDialog

• Persistence
  – Saving the ToolbarControl state
  – Saving MapDocuments
  – Saving LayerFiles
The CustomizeDialog

- Modeless dialog that allows you to:
  - Add and remove commands from the ToolbarControl at runtime
  - Browse existing ESRI commands, tools, menus, palettes and toolbars
  - Drag and drop items or double-click

- Need to enable programmatically

- Set ToolbarControl into Customize mode

Demo
Persisting Customizations

• Saving and loading ToolbarControl contents

• Writing map documents & layerfiles
  – Open Document:
    ```csharp
    m_MapDocument = new MapDocumentClass();
    m_MapDocument.Open(sFilePath,"");
    ```
  – Save Document:
    ```csharp
    m_MapDocument.Save(m_MapDocument.UsesRelativePaths, true);
    ```

Demo
Tips and Tricks

• Synchronizing MapControl and PageLayoutControl
• DPI Issue – 96 v 120 DPI
• Dynamic Display
• Singletons
• Geoprocessing
• Migrating Existing applications to 9.2
MapControl & PageLayoutControl Synchronization

How to synchronize the MapControl and PageLayoutControl

Summary
A common task for an ArcGIS Engine developer is to synchronize (in the same application) the map used by the MapControl with the focus map of the PageLayoutControl. This topic discusses the methods used to achieve this synchronization.

Development licensing
Engine Developer Kit
Deployment licensing
ArcView
ArcEditor
ArcInfo
Engine Runtime

Synchronizing the MapControl and PageLayoutControl
A common task for an ArcGIS Engine developer is to synchronize (in the same application) the map used by the MapControl with the focus map of the PageLayoutControl. For example, an application may have one tab containing a MapControl and one tab containing a PageLayoutControl, both displaying the same map data. Any changes made to a map in one control must be reflected in the other control when its tab is activated.

Consider the following differences between the ArcMap desktop application and the ArcGIS Engine controls:
- The ArcMap application has two views: page layout view and data frame view (also known as map view). These two views share the same map object. For example, if some features are selected in map view, the same feature selection will be visible on the map in page layout view. However, the map object can only be active in one view at any one time; the active one is either a PageLayout or a map.
- The ArcGIS Engine MapControl and PageLayoutControl do not automatically know of each other’s existence within an application; they do not share the same map object. For example, if the map extent is changed in the MapControl, the new map extent will not be reflected in the focus map of the PageLayoutControl.

The synchronization that ArcMap provides can be implemented with the MapControl and PageLayoutControl as follows:
- Listen to the events fired by a map object when it is changed in the MapControl and PageLayoutControl and make the same change in the other control. For example, the ViewEvents_DocumentCreated event is fired when a new layer is added to the map in the MapControl. When the event is fired, the same layer needs to be programmatically added to the focus map within the PageLayoutControl. This method of synchronization is not recommended due to the sheer number of events that need to be listened to for trapping all possible changes that can be made to a map object.
- If a change is made to the focus map of the PageLayoutControl, the ObjectCopy object can be used to copy (deep copy) the map object and the resulting map copy can be set into the MapControl. Each map contained within the PageLayout encapsulated by the PageLayoutControl resides within a separate MapFrame, and therefore has its MapFrame property set to True. A Map contained within the MapControl does not reside within a MapFrame.
DPI Considerations
Dynamic Display

• **Supports High Performance Drawing in 2D**
  – Leverages OpenGL (No experience required)
  – ArcObjects API for drawing to Dynamic Display

• **Developers Create ‘Dynamic’ Layers**

• **Map handles the drawing of Dynamic Layers**

• **Why use Dynamic Display?**
  – Useful for drawing many objects on the display quickly
  – Improved performance

**Demo**
Singletons

• Only ever 1 instance of a singleton per thread

• CommandsEnvironment
• EngineEditor
• EngineInkEnvironment
• EngineNetworkAnalystEnvironment
• MyPlaceCollection

Demo
Geoprocessing

- ArcGIS 9.2 provides an ESRI.ArcGIS.Geoprocessor assembly
  - Contains a managed Geoprocessor class
  - Used to execute GP tools

- Each system Geoprocessing toolbox has its own managed assembly
  - Contains classes representing each GP tool in the toolbox

- Use the Geoprocessor class to set up and execute the GP tools

Demo
Migrating Existing Applications to 9.2

• New Controls library
  – esriControls.olb

• New Primary Interop Assemblies (PIA’s)
  – ESRI.ArcGIS.AxControls
  – ESRI.ArcGIS.Controls

• Contains
  – All ArcGIS Engine Controls
  – All out of box commands
    – menus
    – palettes
    – multi-items
Summary

• Extend with custom components
• Persistence
• Tips & Tricks
Questions?

Mary Harvey and Mike Rudden
Other Sessions

4:30pm – 5.45pm Tuesday

• Leveraging the Geoprocessing Framework in ArcGIS Engine in .NET

1:00pm – 2:15pm Wednesday

• Creating Editing Applications with ArcGIS Engine in .NET

2:45pm – 4.00pm Wednesday

• ArcGIS Engine and ArcGIS Desktop Panel Discussion

4:30pm – 5:45pm Wednesday

• Network Analysis in ArcGIS Engine and ArcGIS Desktop