



# 2008 ESRI Developer Summit

March 17–20, 2008 • Palm Springs, CA

## Developing .NET Applications for ArcGIS Engine

*Mike Rudden    Mary Harvey*

# Schedule

- **75 minute session**
  - 60 - 65 minute lecture
  - 10 - 15 minutes Q & A
- **Cell phones and pagers**

***Please!***  
*Turn **OFF** cell phones  
and paging devices*



- Please complete the **session survey** – we take your feedback very seriously!

# What's new in ArcGIS Engine 9.3...

- **Dynamic Display Improvements**
  - + Much enhanced SDK documents and samples
- **Significantly enhanced Editing API**
  - + Comprehensive SDK documents and samples
- **Also:**
  - Network Analyst Extension improvements
  - Error Reporting - New Crash Dump Analysis
  - SQL Server Express DVD included
  - Visual Studio 2008 supported

# Session Objectives

- Raise awareness for the new improved visualization capabilities of **dynamic display** and provide guidance on best practices when building applications
- Raise awareness of the new programmatic **editing** capabilities for creating business focussed applications



# 2008 ESRI Developer Summit

March 17–20, 2008 • Palm Springs, CA

## ArcGIS Engine Dynamic display

# Topics

- **Overview**
  - What is it
  - When to use it
- **Static Content**
  - Caching
  - Cache Management
- **Dynamic Content**
  - Dynamic Layers
  - Dynamic Events
  - Dynamic Drawing

# What is Dynamic Display?

- It is an alternative display mechanism specifically designed for dynamic GIS display environments.
- It moves intensive graphics rendering from the CPU to the graphics hardware:
  - Active Display
  - Uses OpenGL for rendering
  - Sub-second refresh rate
- ArcGIS Engine developer technology

# Dynamic Display Capabilities

Update many  
moving objects  
with  
Smooth navigation  
GIS data query

+

Performance  
Improvements  
Multi-threaded  
Caching

+

Further performance  
Improvements  
Cache management  
Text & Symbol  
Improvements  
Selection

9.2

> 9.2 sp3

9.3



# When to use it

- **When to use it:**
  - Smooth navigation
  - Displaying moving objects
  - Displaying real time or other feeds
  - Playback of recorded feeds
  
- **When not to use it:**
  - Creating high quality maps
  - Printing
  - Editing

**Controlled  
Environments**

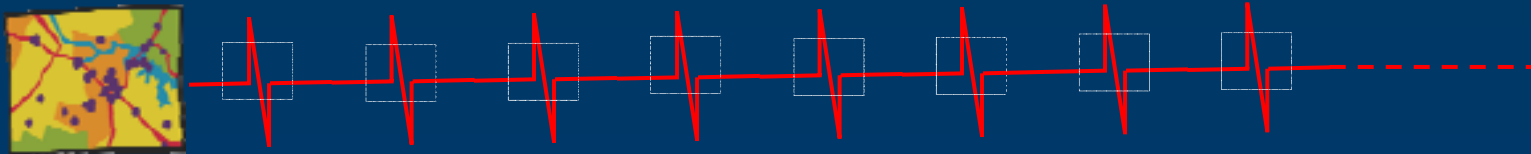
# Enable Dynamic Display

- **Good video card**
  - OpenGL 1.1 (or higher) compliant
  - 64MB VRAM

```
//enable dynamic display
IMap map = ...;
IDynamicMap dynamicMap = (IDynamicMap)map;
dynamicMap.DynamicMapEnabled = true;
```

- **Continuous draw cycle**

- **IDynamicMap.DynamicDrawRate** (30 msec default)



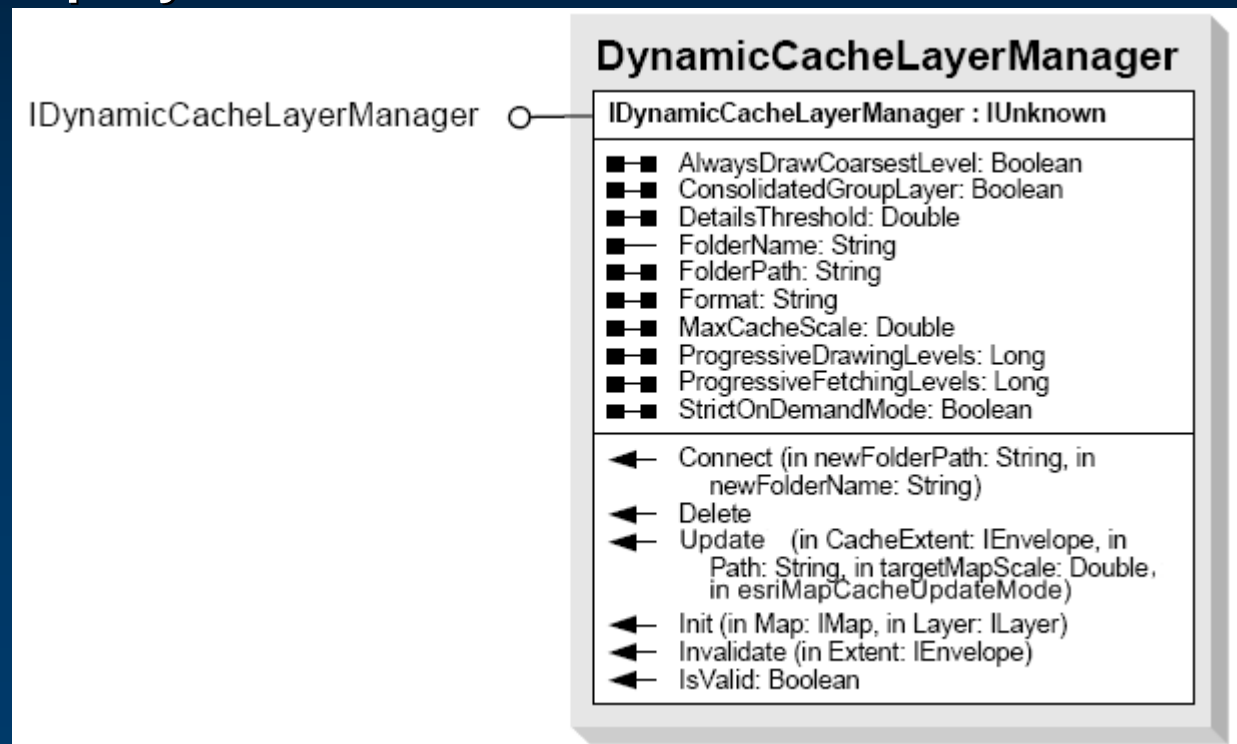
# Static Content

- Handled on a background thread
- Rendered as tiles
- (File & SDE) Geodatabase rasters and MapServer layers
  - direct read without caching
- Other static content:
  - Local cache created
  - Persist the cache manually:
    - Ensure map is in dynamic mode
    - Save layers - layer files (.lyr) or map document (mxd)
    - Reuse the cash in subsequent sessions
  - Or programmatically - IDynamicCacheLayerManager

**Demo**

# Cache Management

- **IDynamicCacheLayerManager** interface
  - Pre-generate, connect to, update, invalidate cache
  - Cache format (PNG, JPEG)
  - Progressive drawing, fetching levels
  - Consolidate group layer tiles



# Creating, deleting or updating the cache

```
//create, update or delete cache
string sCacheFolderPath = @"C:\myCache";
IDynamicCacheLayerManager dcm = new
    DynamicCacheLayerManagerClass();

IMap map = axMapControl1.Map;
ILayer layer = map.get_Layer(0);
if (layer is IDynamicLayer)
    return;

dcm.Init(map, layer);
dcm.Update(null, sCacheFolderPath, map.MapScale,
    esriMapCacheUpdateMode.esriMapCacheUpdateRecreateAll);
```

# Connecting to a cache

```
//connect to the cache
IDynamicCacheLayerManager dcm = new
    DynamicCacheLayerManagerClass();

string newFolderPath = @"C:\ReConnect";
string newFolderName = @"C:\MyLayer_GUID";

IMap map = axMapControl1.Map;
ILayer layer = map.get_Layer(0);
if (layer is IDynamicLayer)
    return ;

dcm.Init(map, layer);
dcm.Connect(newFolderPath, newFolderName);
```

# Dynamic Content - Dynamic Layers

- Custom layer implements **IDynamicLayer**
- Will be visible in the map's layers list and TOC
- **DrawDynamicLayer** method
  - Contains the drawing commands
  - Immediate phase : Directly drawn to display
  - Compiled phase : Stored in a display list
- **DynamicLayerDirty**
  - Indicates if layer wants to redraw
- **DynamicRecompileRate**
  - Indicates when layer wants to recompile its display list
  - Only for Compiled phase

# Dynamic Display Draw Cycle

For each dynamic layer -

- Is **DynamicLayerDirty** for Immediate phase?
  - Yes : invoke **DrawDynamicLayer** of all dynamic layers in Immediate phase
  - No : continue
- Is **DynamicLayerDirty** for Compiled phase && **DynamicRecompileRate** elapsed?
  - Yes : invoke **DrawDynamicLayer** of current layer only in Compiled phase and of all dynamic layers in Immediate phase
  - No : continue

# Creating A Dynamic Layer

- Create a custom layer
  - Implement `ILayer, IGeodataset`
  - Implement `IDynamicLayer`

Or

- Create a custom layer
  - Inherit from the `BaseDynamicLayer` class
  - Override `DrawDynamicLayer` method

# Dynamic Content – Dynamic Map Drawing

- Listen to the **IDynamicMapEvents**
  - **BeforeDynamicDraw**
  - **AfterDynamicDraw**
- Will not be visible in the map's layers list and TOC.

# Dynamic Display Drawing API

- OpenGL API
- Dynamic Display Drawing API

## 1. Create a glyph (marker, line, fill, text)

- Using **IDynamicGlyphFactory**

## 2. Apply properties to the glyph (color, scale, rotation,...)

- Using **IDynamicSymbolProperties2 \***

## 3. Draw the glyph

- Using **IDynamicDisplay, IDynamicDrawScreen, or IDynamicCompoundMarker2 \***

**\* new interfaces at 9.3**

**Demo**

# Tips

- **Use Snippets and SDK**
- **Static Content**
  - Use Geodatabase Raster & Map Server layers
  - Rasters with pyramids
  - Pregenerate local cache
  - Remove unused layers & set scale thresholds
  - Use annotation in preference to labels
  - Use consolidated group layers
- **Dynamic Content**
  - Use Compiled phase for content which does not change frequently
  - Create DynamicGlyph once, then reuse
  - Delete DynamicGlyph when done



# 2008 ESRI Developer Summit

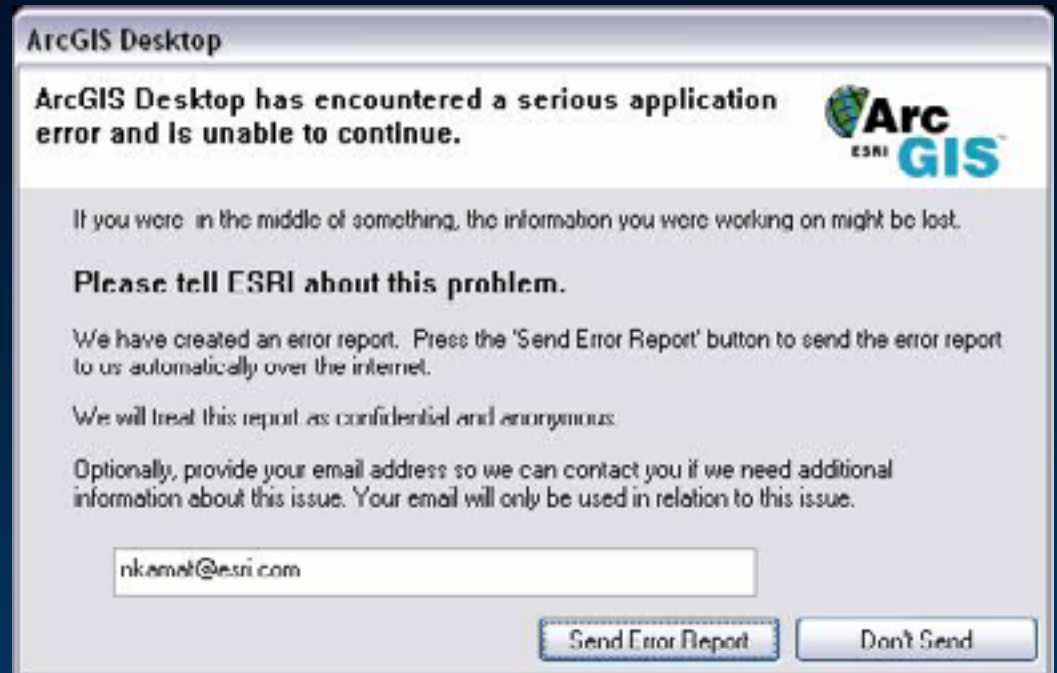
March 17–20, 2008 • Palm Springs, CA

## Error Reporting

### *Crash Dump Analysis*

# Crash Dumps: user view

- “Send Error Report”:  
automatically send  
to ESRI



- “Don't Send”:
  - .dmp files saved into user profile: \Application Data\ESRI\Error Reports
  - Can be emailed to [ArcGISErrorReport@esri.com](mailto:ArcGISErrorReport@esri.com)

# Crash Dumps: for developers

- Point Visual Studio 2005 at the new ArcGIS Symbol Servers
- Open Crash Dump file in VS and go to the function call (F5) where the error occurred (or hang occurs)





# 2008 ESRI Developer Summit

March 17–20, 2008 • Palm Springs, CA

## ArcGIS Engine Editing

# Topics

- **Editing capabilities**
- **Architecture**
- **Core programmatic editing**
  - new interfaces
- **Imposing business logic**
- **Extending the editing framework**

# ArcGIS Engine Editing Capabilities

**Editing support provided at Geodatabase level**

+

**Rich set of UI Editing components**

+

**Comprehensive programmatic editing capabilities**

**9.1**

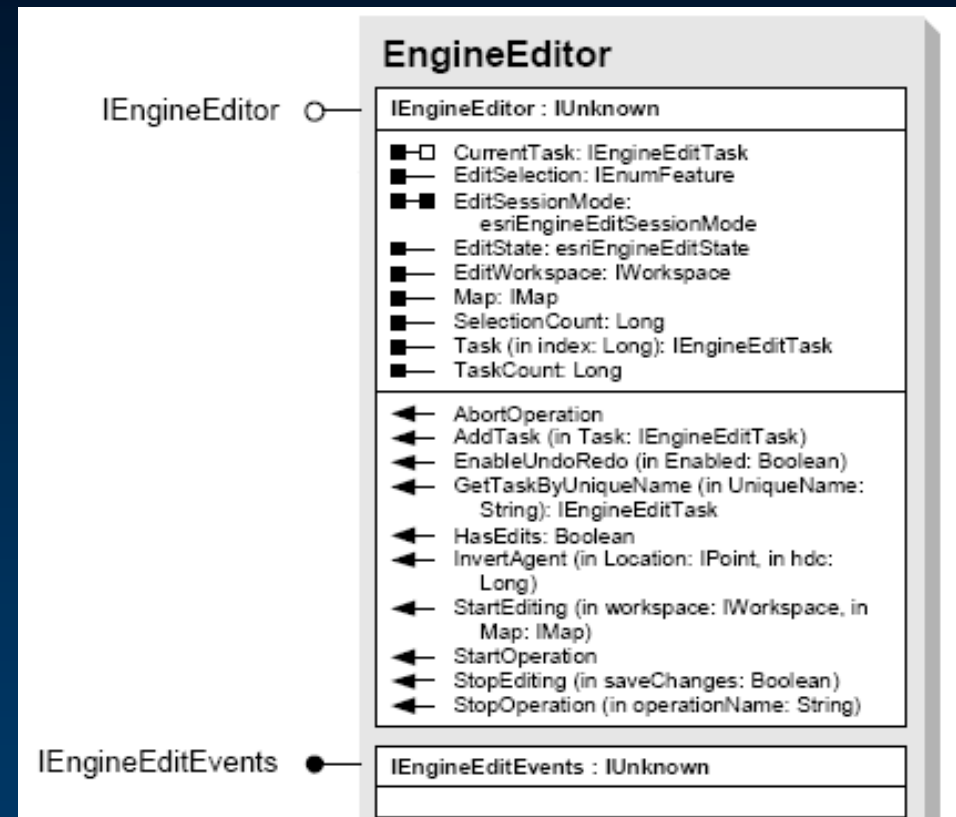
**9.2**

**9.3**



# Editing Architecture

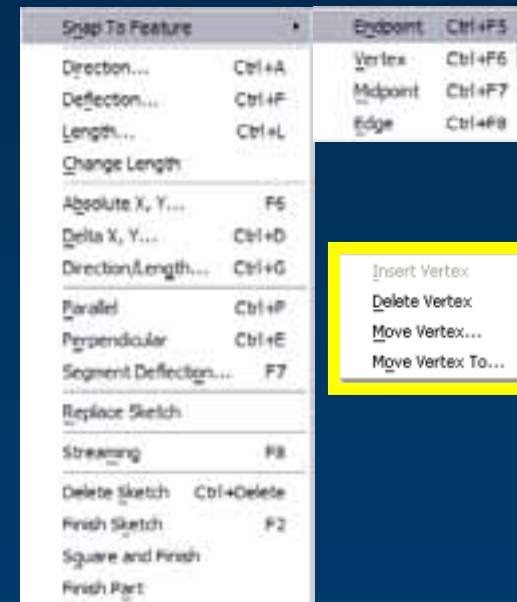
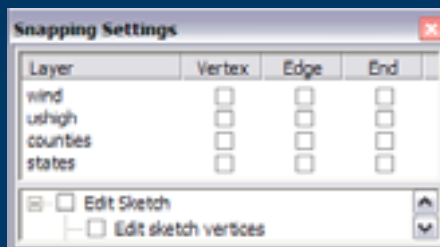
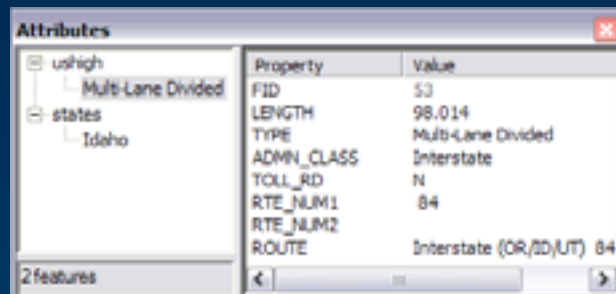
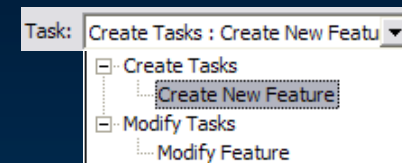
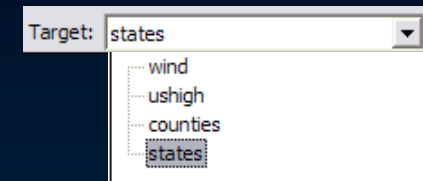
- Provided in the Controls library
- No access to Editor library (ArcMap)
- Primary object is EngineEditor singleton



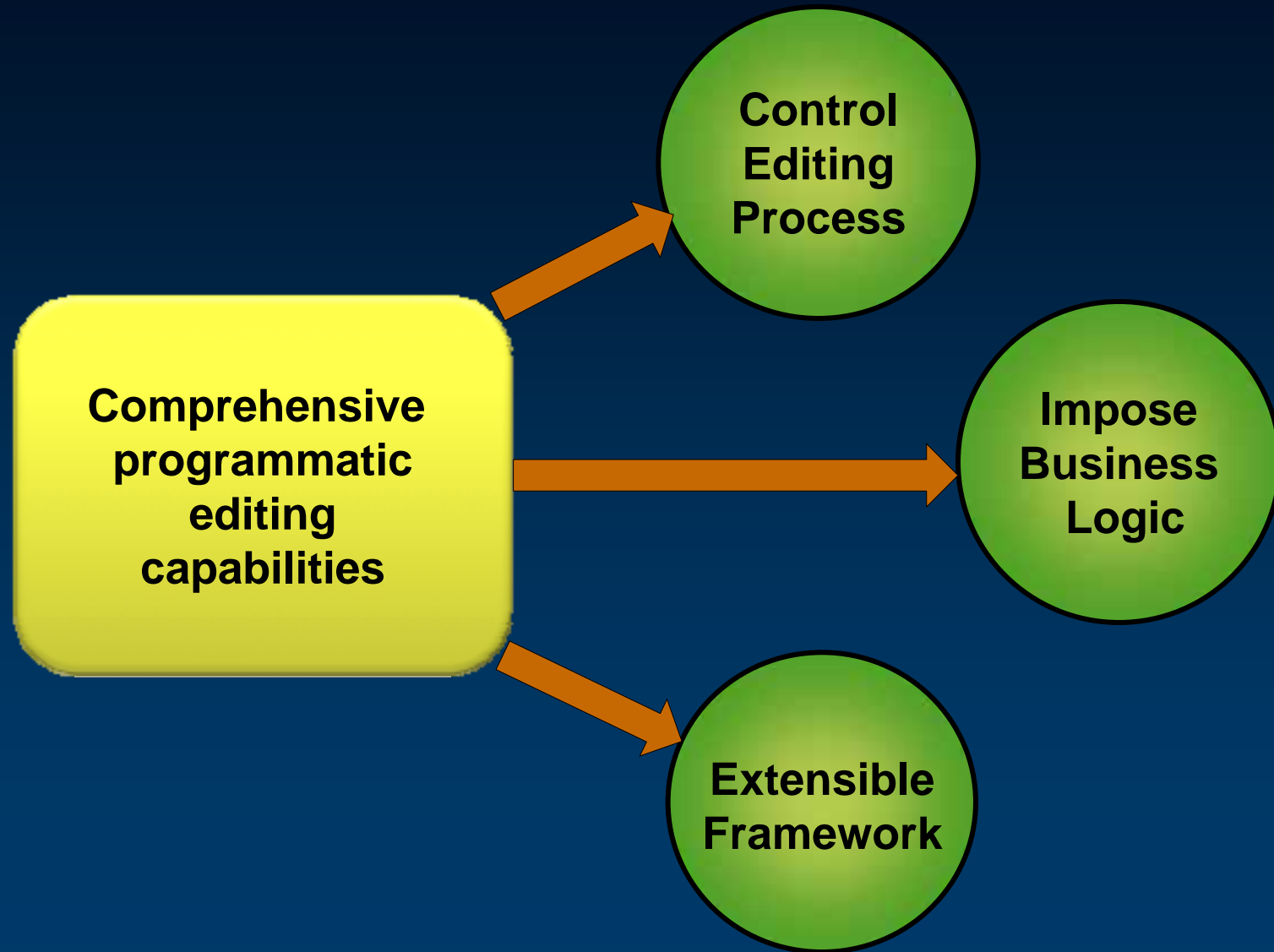
```
IEngineEditor engineEditor = new EngineEditorClass();
```

# Editing at 9.3: UI Components

- 43 Out-Of-The-Box
- Editor menu
- Sketch & edit tools
- Snapping, attribute, sketch properties dialogs
- Target layer, edit task controls
- Sketch, **edit vertex** context menus



# Editing at 9.3: Programmatic Access



# Core Editing: Interfaces



- **EngineEditor**
  - Manage edit session & edit tasks
    - **IEngineEditor**
  - Manage target layer
    - **IEngineEditLayers**
  - Manage sketch geometry
    - **IEngineEditSketch**
  - Manage snap agents and snapping settings
    - **IEngineSnapEnvironment**
  - Manage properties of an edit session
    - **IEngineEditProperties, IEngineEditProperties2**

# Core Editing: Common Steps

1. Get instance of EngineEditor
2. Set up the operation stack to allow undo/redo
3. Start editing
4. Set target layer
5. Get instance of desired edit task
6. Set the current edit task
7. Set the current tool (e.g. sketch tool)
8. ....Perform edits (within Edit & Sketch Operations)
9. Stop Editing

Demo

# ArcGIS Engine Editor Events

Impose  
Business  
Logic

- **EngineEditor**
  - Respond to edit events
    - **IEngineEditEvents**
- **Validate Edits**
  - Should edits be allowed
- **Monitor & Respond**
  - Set enabled state of custom tools
  - Show certain tools when editing starts

IEngineEditEvents	IEngineEditEvents : IUnknown
	← OnAbort
	← OnAfterDrawSketch (in Display: IDisplay)
	← OnBeforeStopEditing (in saveChanges: Boolean)
	← OnBeforeStopOperation
	← OnChangeFeature (in Object: IObject)
	← OnConflictsDetected
	← OnCreateFeature (in Object: IObject)
	← OnCurrentTaskChanged
	← OnCurrentZChanged
	← OnDeleteFeature (in Object: IObject)
	← OnSaveEdits
	← OnSelectionChanged
	← OnSketchFinished
	← OnSketchModified
	← OnStartEditing
	← OnStartOperation
	← OnStopEditing (in saveChanges: Boolean)
	← OnStopOperation
	← OnTargetLayerChanged
	← OnVertexAdded (in point: IPoint)
	← OnVertexDeleted (in point: IPoint)
	← OnVertexMoved (in point: IPoint)

Demo

# ArcGIS Engine Custom Commands



Extensible  
Framework

- Many out-of-box commands
- You can create your own:
  - Implement ICommand (and ITool for a custom tool)
  - Register class in 'ESRI Controls commands' component category
- Enclose edits within edit and sketch operations
  - Provides undo/redo capability for edits
  - Take care setting up OperationStack

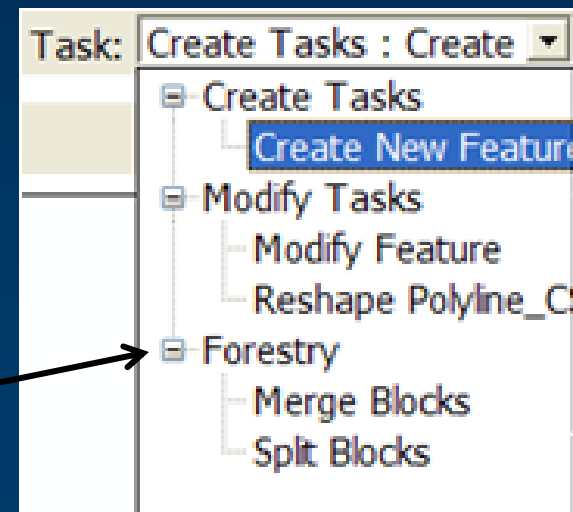
**Demo**

# ArcGIS Engine Edit Tasks

Extensible  
Framework

- Two out-of-the-box tasks provided:
  - Create New Feature
  - Modify Feature
- You can create your own:
  - Implement IEngineEditTask
  - Register class in 'ESRI Engine Edit Tasks' component category
- Use them to incorporate business logic

Create custom groups



Demo

# Customized Attribute Editing

- Extend out-of-the-box attribute editor
  - Implement IEngineFeatureInspector interface

Extensible  
Framework



- Bind a Geodatabase table to .NET Control
  - SDK Sample

Demo

# SQL Server Express with Engine Runtime

- Free, lightweight version of SQL Server 2005
- Will be included with ArcGIS Engine 9.3
  - Was already included with ArcGIS Desktop at 9.2
- Easy to 'SDE enable' using Post installation wizard
- Facilitate development of field based editing solutions which use:
  - Engine Runtime + Geodatabase Update Extension
  - New Editing Framework
  - Replication Capabilities
  - ArcGIS Server Services

# Questions...

- Slides and code will be available on EDN
- Please fill out session surveys!
- **Still have questions?**
  1. Tech talk, Demo Theatres, Meet the Team
  2. “Ask a Developer” link on web page