



Developers Guide to 3D Visualization in ArcGIS 9.2

Nathan Shephard
Tamrat Belayneh

Contents

- **3D Visualization (Mapping) defined**
- **Globe Data Display Management & Architecture**
- **Caching and optimization**
- **Data Preparation and Usage Tips**
- **Customization Framework**
- **3D Rendering**
- **Publishing 3D Globes**
- **Developers Best Practice Guide**
- **Demos**
- **Question and Answers**

What is 3D Mapping?

- **Viewing spatial data in 3D: raster, vector, elevation**
- **Seamless transitions between global, local and street-level scale data**
- **Performing GIS analysis within a 3D context**
- **3D Mapping has applicability in fields such as:**
 - **Geology**
 - **Meteorology**
 - **Climatology**
 - **Hydrology**
 - **Utility Management**
 - **Disaster Management**
 - **Real estate, etc...**

3D GIS

Surface Data in 3D

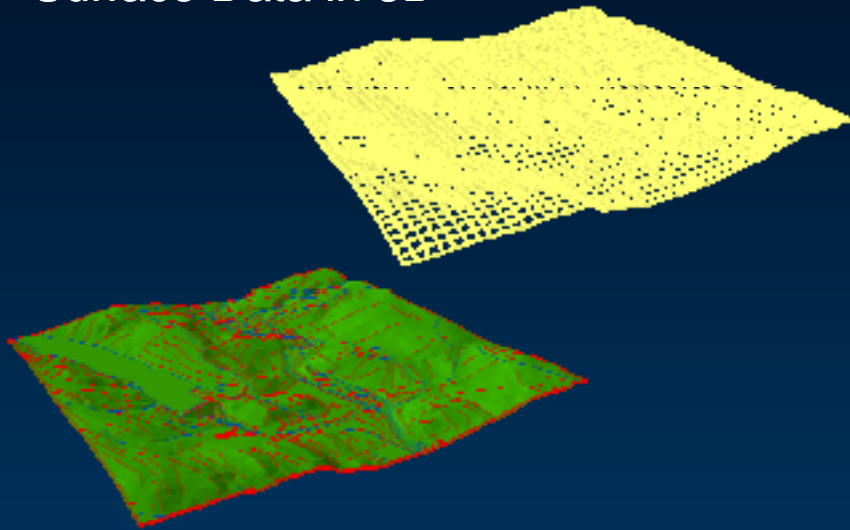
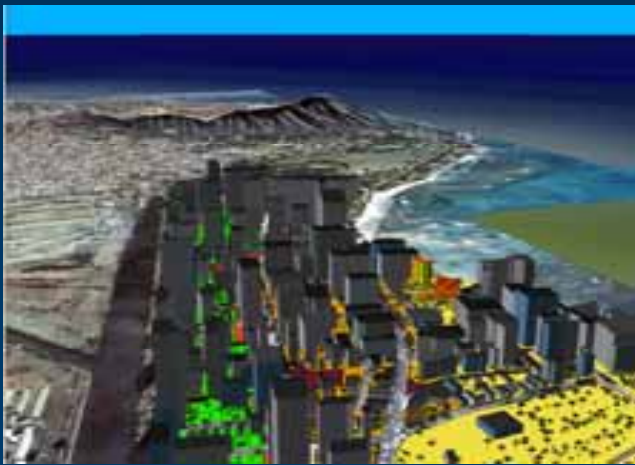
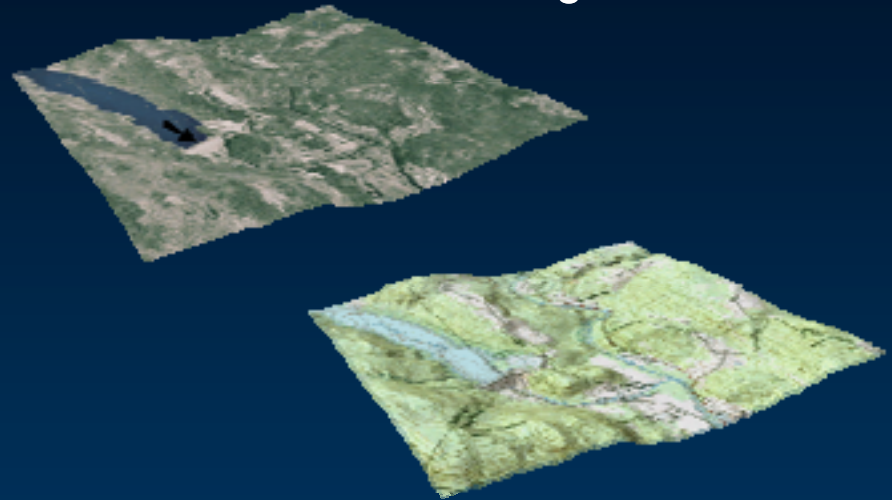
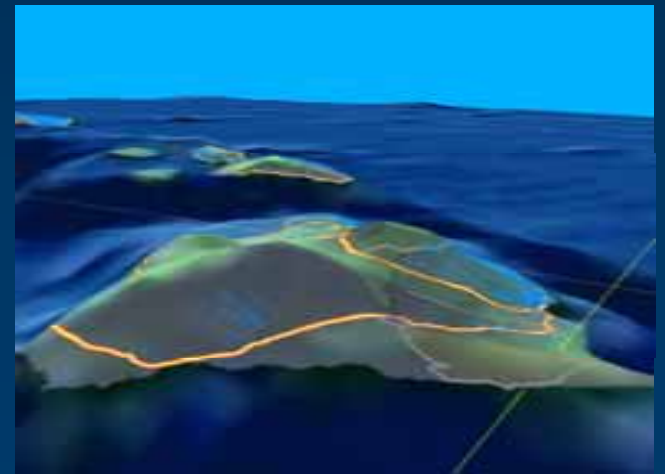


Image Data in 3D



Vector Data in 3D



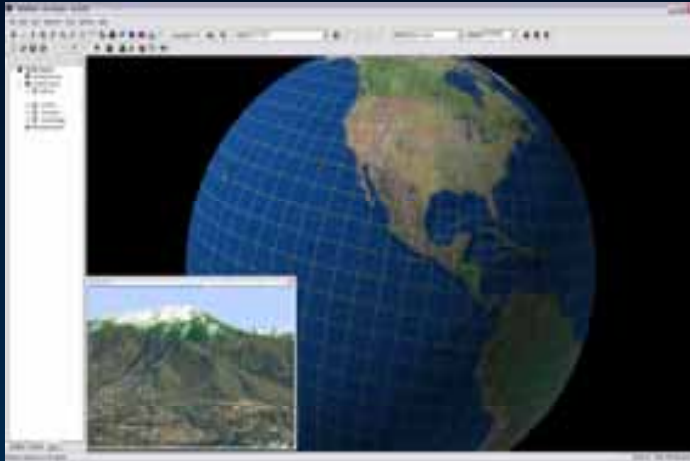
What can 3D Visualization do for you?

- **Accurately represent real-world GIS entities**
 - We live in a 3D world, our data should display that way
- **Create realistic 3D models (e.g. cityscapes)**
 - Tourism, Advertising
- **Visualize ‘what if’ scenarios**
 - Planned construction (buildings, roads, transmission lines, etc)
 - Decision support systems (evacuation, mock attacks, etc)
- **Gain insights into your data, including:**
 - 3D spatial relationships
 - Sizes / Scales
 - Visibility / Line of sight

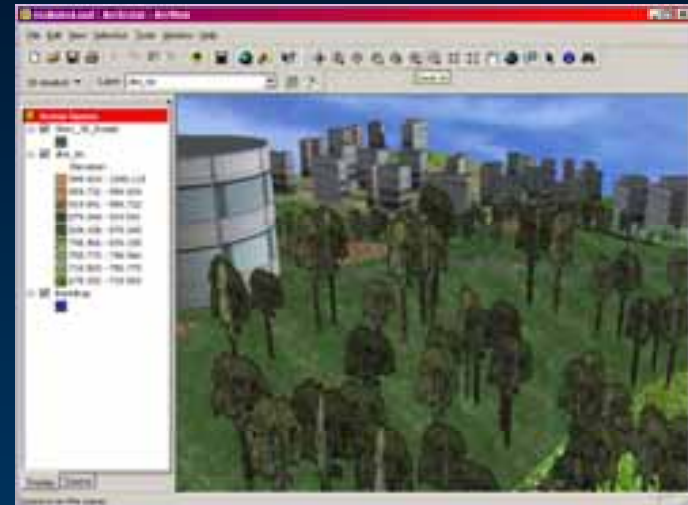
3D Visualization Solutions from ESRI

- **Out-of-the-box Desktop, Engine and Server solutions**
- **Desktop:**
 - ArcGIS 3D Analyst (ArcGlobe and ArcScene)
 - ArcGIS Explorer
 - ArcReader
- **ArcGIS Engine Solutions:**
 - Globe Control
 - Scene Control
- **ArcGIS Server Solution:**
 - Globe Server
 - Allows analysts to publish rich GIS web Services
 - No programming required

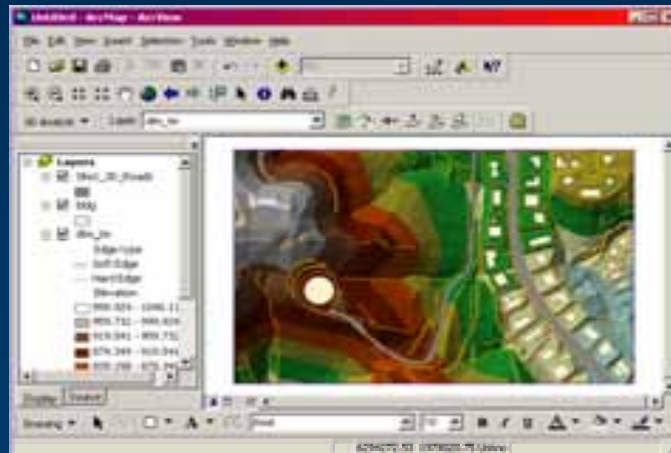
3D Mapping Environments in ArcGIS



ArcGlobe

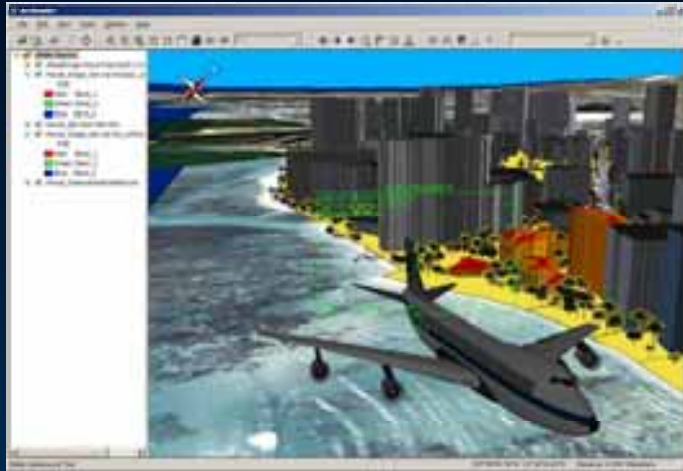


ArcScene



ArcMap

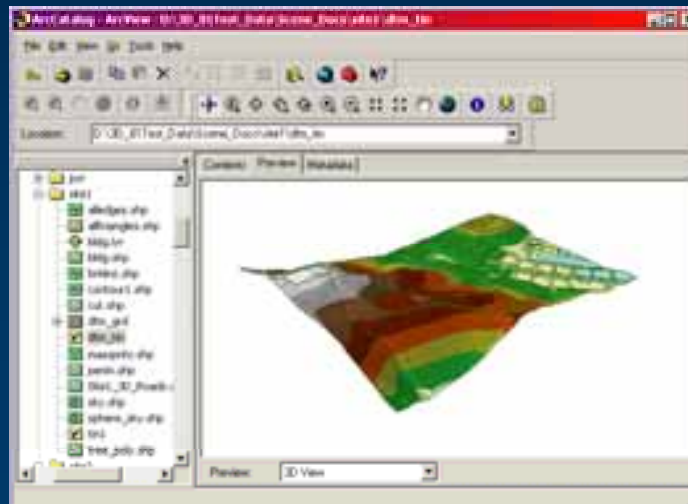
3D Mapping Environments in ArcGIS



ArcReader

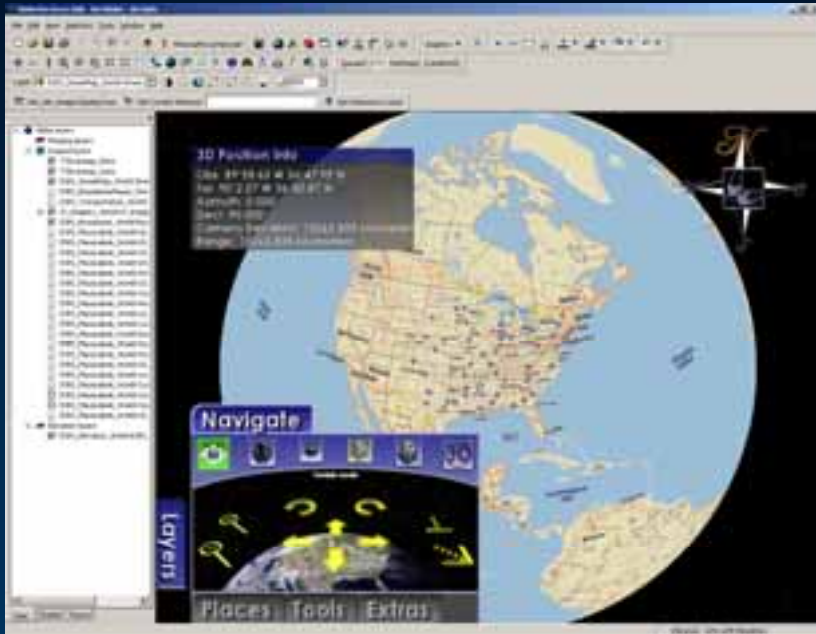


ArcGIS Explorer

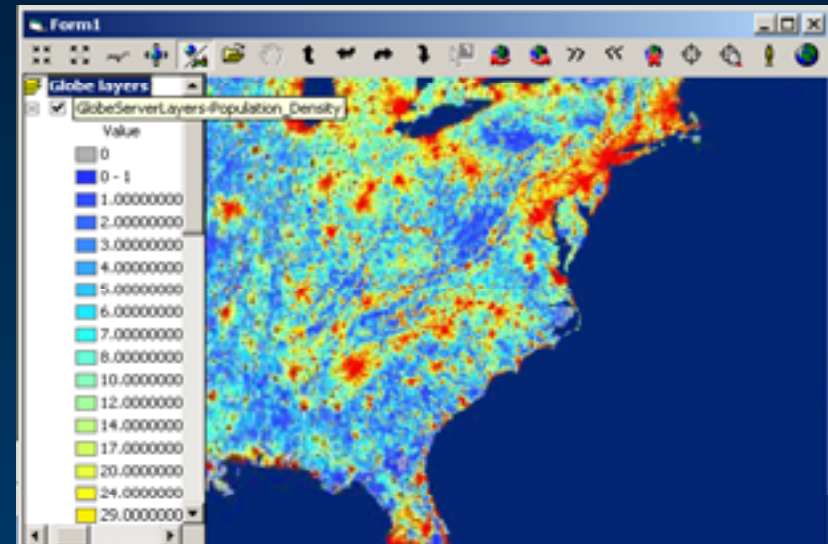


ArcCatalog

3D Mapping Environments in ArcGIS



Extensions to ArcGIS applications



Custom Application Built using Controls

3D Mapping Environments in ArcGIS

Example: stand-alone 3D Visualization by TouchTable Inc.



Globe Data Management Architecture

- All Globe based application share a fixed internal Spatial Reference - the Cube Projection- that organizes and optimizes display



**Cube
Projection in
2D view**

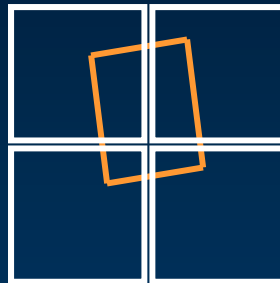
Globe Data Management Architecture

- Data is internally divided into **tiles** at different levels of detail (**scales**)

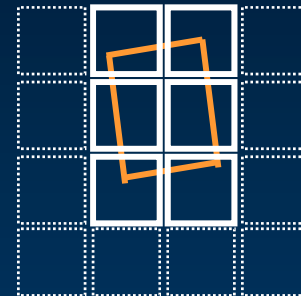
Data area



Level 0



Level 1



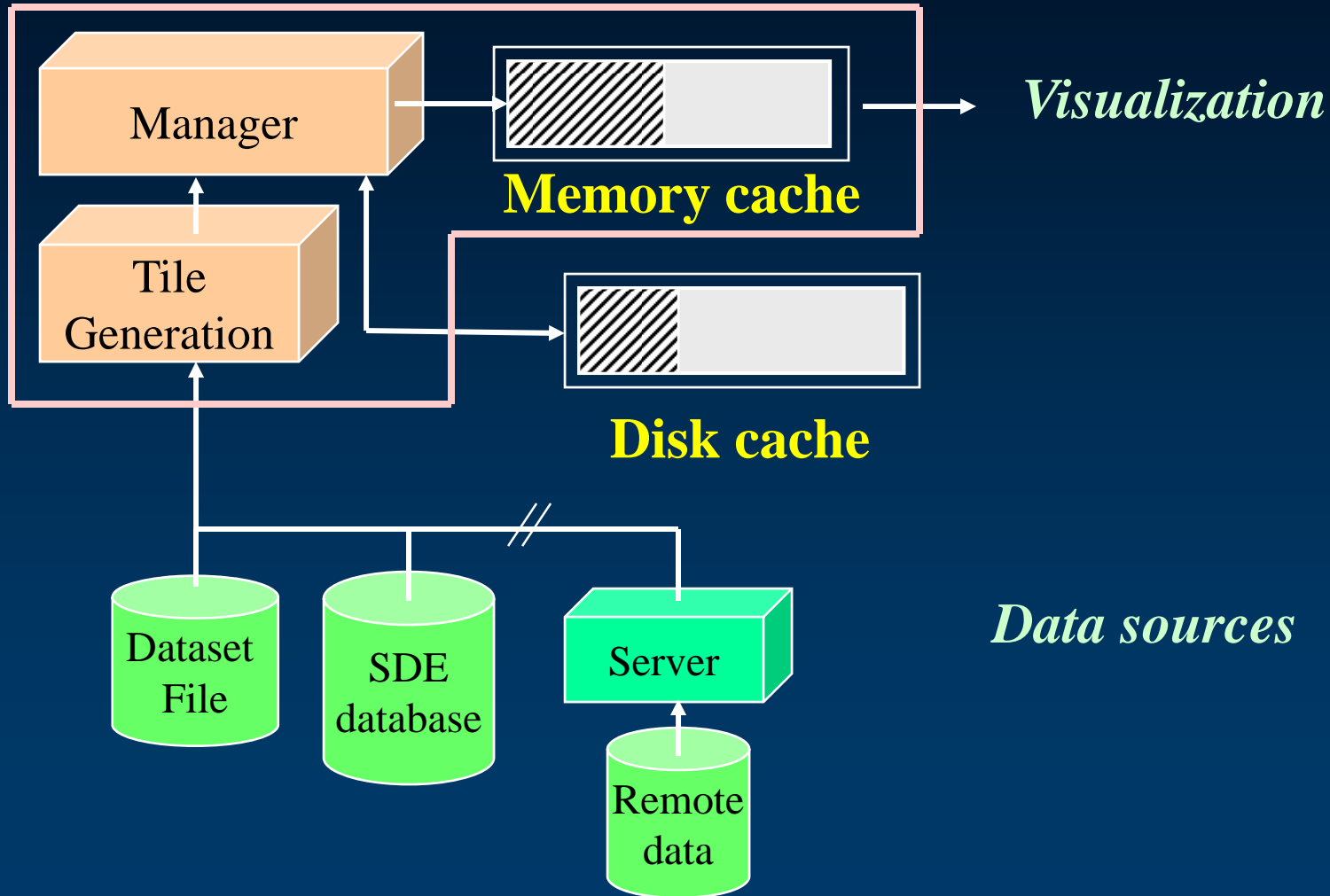
Level 2

- Results in a **multi-resolution** representation of the data
- Features as vectors have only one level of detail

Globe Data Caching Architecture

- To improve display performance, all Globe applications keep a cache of tiles for each **layer**
 - ArcGlobe, ArcGIS Explorer, ArcReader, GlobeControl apps
- Tiles are stored in the **memory cache** for immediate use
- Tiles are written to the **disk cache** for later use
- Tiles are swapped between the memory cache and the disk cache based on resources and the current view
- The architecture uses **two threads**:
 - One for rendering and navigation
 - One for tile creation, writing and fetching

Globe Data Caching Architecture

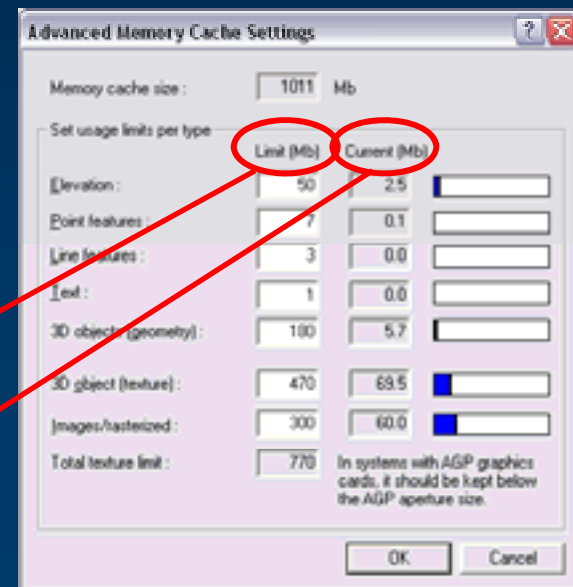
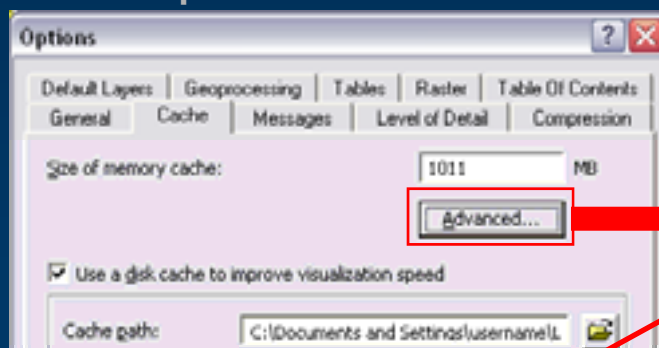


Globe Data Caching Architecture

Memory cache

- The memory cache is configured by data type
- The application will define a default size and configuration for the memory cache, but can be overridden for large or specialized 3D views
- Total memory usage should not exceed the physical memory (RAM)

Tools > Options



```
IGlobeAdvancedProperties2  
.GetTileMemory(type, limit, current);  
.SetTileMemory(type, limit);
```

Globe Data Caching Architecture

Disk cache

- Visualization tiles stored on disk for fast retrieval
- Each layer cache is **preserved** between sessions only if the layer is saved in a document or layer file
- The layer cache is **invalidated** if any display properties change (for example, symbology)
- There are two methods of generating the cache:
 - **On-demand**: tiles created as needed (default)
 - **Explicit**: all areas
 - **Partial**: only specified levels of detail (scales)
 - **Full**: all levels of detail (scales)

Globe Data Caching Architecture

On-demand caching (default)

- Indicated by **spinning globe** in ArcGlobe's status bar

```
IGlobeStatInfo.Statistic(esriTileRequestQueue)
```

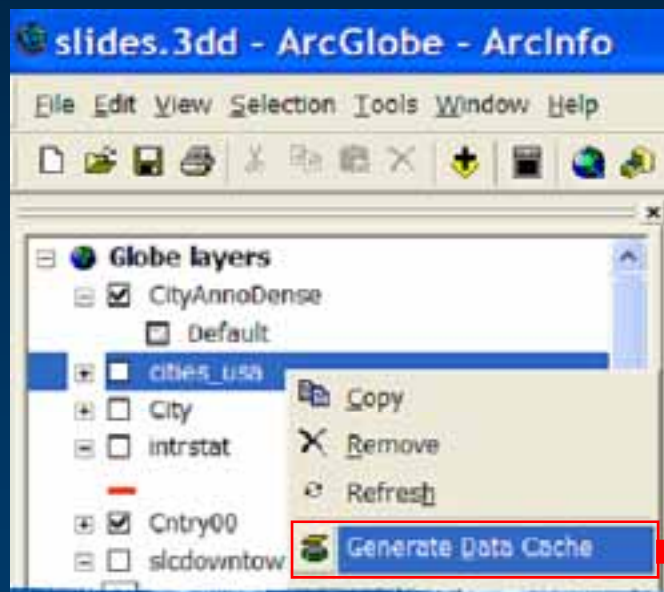
- It doesn't interrupt navigation (multi-threaded)
- The data source location, format, and projection impacts the caching performance (see Tips)
- Build a cache for an **area of interest** by using a camera animation to pre-visit the area
- **Disable disk caching** for layers with data that is constantly changing / updating

```
IGlobeLayerProperties2.UseCache = False;
```

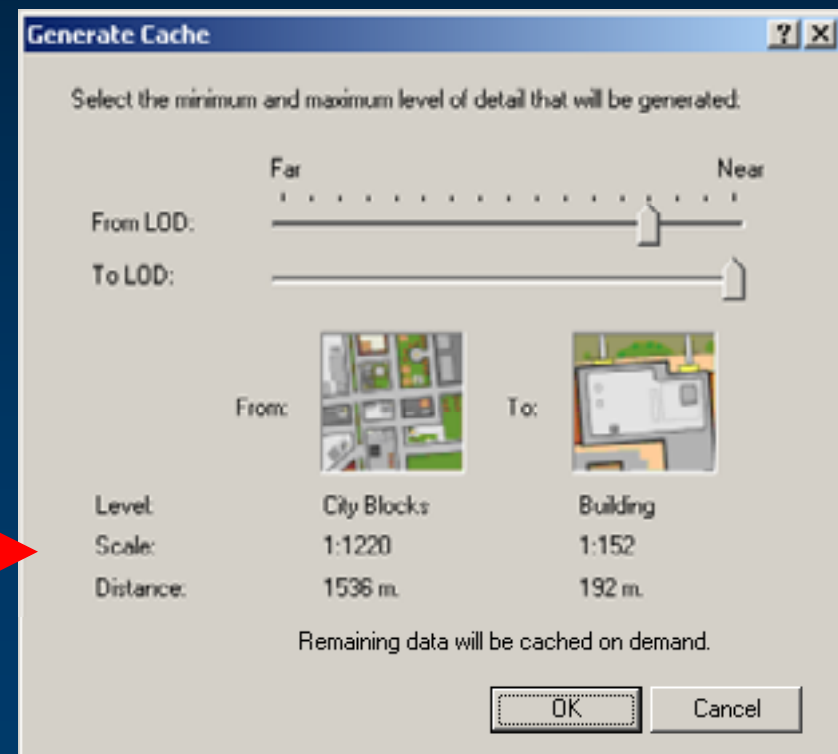
Globe Data Caching Architecture

Explicit caching

- Creates tiles for the layer for specific **levels of detail**
- May take a while to complete (minutes, hours, days)
- Avoids the performance cost of on-demand caching



```
IGlobeDisplayLayers.GenerateTiles  
(ILayer, FromLOD, ToLOD)
```



Data preparation tips

- Raster data:
 - Build **pyramids** when supported (UI prompt)
 - If the data extent is between 45° N and 45° S, project to **Geographic Coordinate System (GCS)**
 - Otherwise, project to **World Cube PCS**
 - Store rasters in **ArcSDE** – it improves caching performance and memory usage
 - Formats with **less compression** cache faster
 - Create a **Raster Catalog** when displaying many images that make a mosaic (eg: aerial photos)

Data preparation tips

- Feature data:

- For data that needs to be seen through a wide range of scales, use **multiple representations**

- Where possible, create separate feature classes
- Assign each representation a different visibility range

- **Simplify** geometry details that are not important

- Remove building interiors, if not needed
- Reduce the vertex count for polylines / polygons

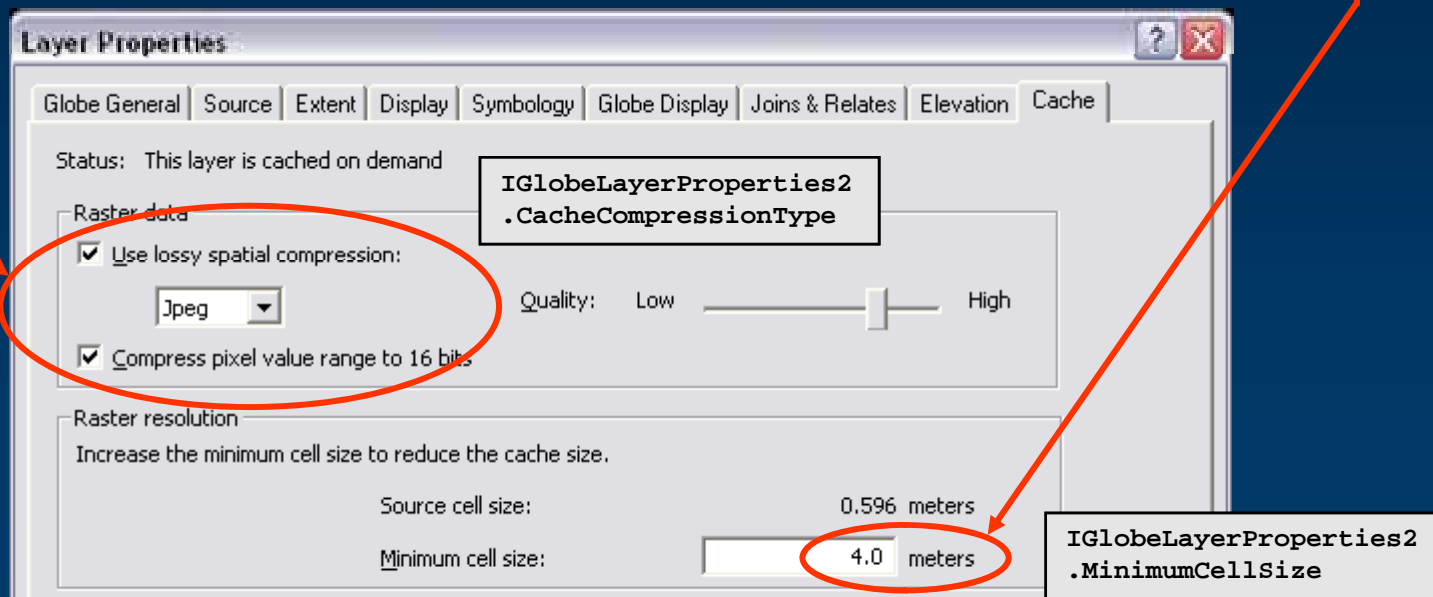
- Downscale or disable multipatch **textures**

- This can be done as a layer property or in the data

Layer Property setting tips

- Raster data

- If you don't need the full image resolution, the **cell size** can be increased to reduce cache size
- **Compression** can help manage cache size and performance
 - A JPEG cache type will create a smaller cache size
 - A DXT cache type will render faster (no uncompression step)



Layer Property setting tips

- Feature data:

- Features can be displayed as **vector** or **rasterized**

- Use vectors for 3D geometries (pipes, buildings, trees)
- Use rasters for draped features (roads, countries)

```
IGlobeLayerProperties2  
.IsDynamicallyRasterized
```

- Where possible, display features as **rasterized**

- It will perform faster and drape on the surface better

- Use a **visibility distance ranges**. A lot.

```
IGlobeLayerProperties2  
.MinimumDistance  
.MaximumDistance
```

Layer Property setting tips

- **Feature data (rasterized):**
 - Reduce the **level of detail** as much as possible
 - Each higher resolution LOD creates four times the number of tiles as the previous LOD
 - If many rasterized feature layers are used in the same area, use a **Group Layer Cache**
 - Generate a full or partial **layer cache**

```
IGlobeDisplayLayers.GenerateTiles  
(ILayer, FromLOD, ToLOD)
```

Layer Property setting tips

- Feature data as vector

- Set the **scale** (level of detail) carefully

- A default is calculated based on the data (extent, etc)
 - Too many features (vertices) in one tile will result in a *feature overflow* warning
 - Larger tiles mean smaller cache and better performance
 - Smaller tiles mean better per-tile visibility switching

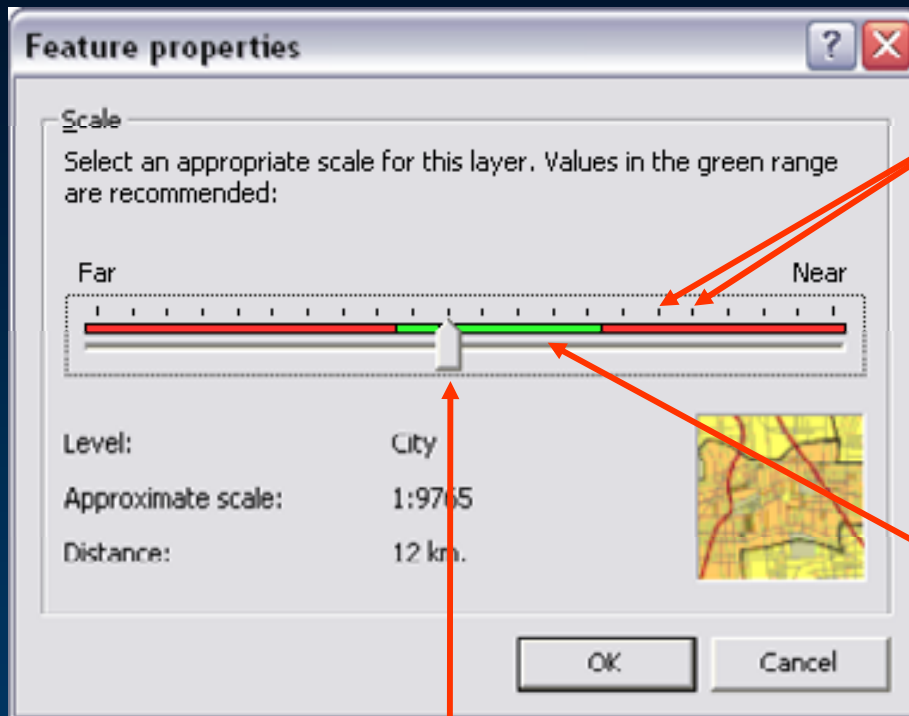
- **Hide the base** of extruded polygons

```
IGlobeHeightProperties2.BottomlessExtrusions
```

- Generate a full **layer cache**

- Vector layers only have a single level of detail and therefore make good candidates for a full cache

Setting the Scale for vector data



Each step down in the scale requires 4 times the number of tiles as the previous one

The green range shows the recommended settings based on a preliminary investigation of the data

```
IGlobeLayerProperties2  
.EstimateFeatureLODRange
```

The default or current value

```
IGlobeLayerProperties2  
.MaxFeatureLevelOfDetail
```

Group Layer Caches

- Consolidates **draped images** and **rasterized features** in a group layer into a single rendering pass

```
IGlobeHelperPub.SetGroupLayerCache ( parameters )
```

- The group layer will get its **own cache** of tiles
 - Optionally, child layers can also continue to keep their own cache
- Notes on consolidated group layers:
 - The group layer cache is invalidated easily
 - Coarser images will be magnified to blend with the others. Use the **Smooth texturing mode** for them
 - The **visibility distances** of the child layers is approximated
 - Elevation / Vector layers in the group are not affected

Group Layer Caches

- **Example 1: Background-based group layer**
 - Layers: States, Counties, Rivers, Towns
 - No child caches – 1 cache, 1 rendering pass
 - With child caches – 5 caches, 1 rendering pass
- **Example 2: Region-based group layer**
 - Layers: Flight Paths, Aerial Photo, Streets, 3D buildings
 - No child caches – 3 caches, 3 rendering passes
 - With child caches – 5 caches, 3 rendering passes

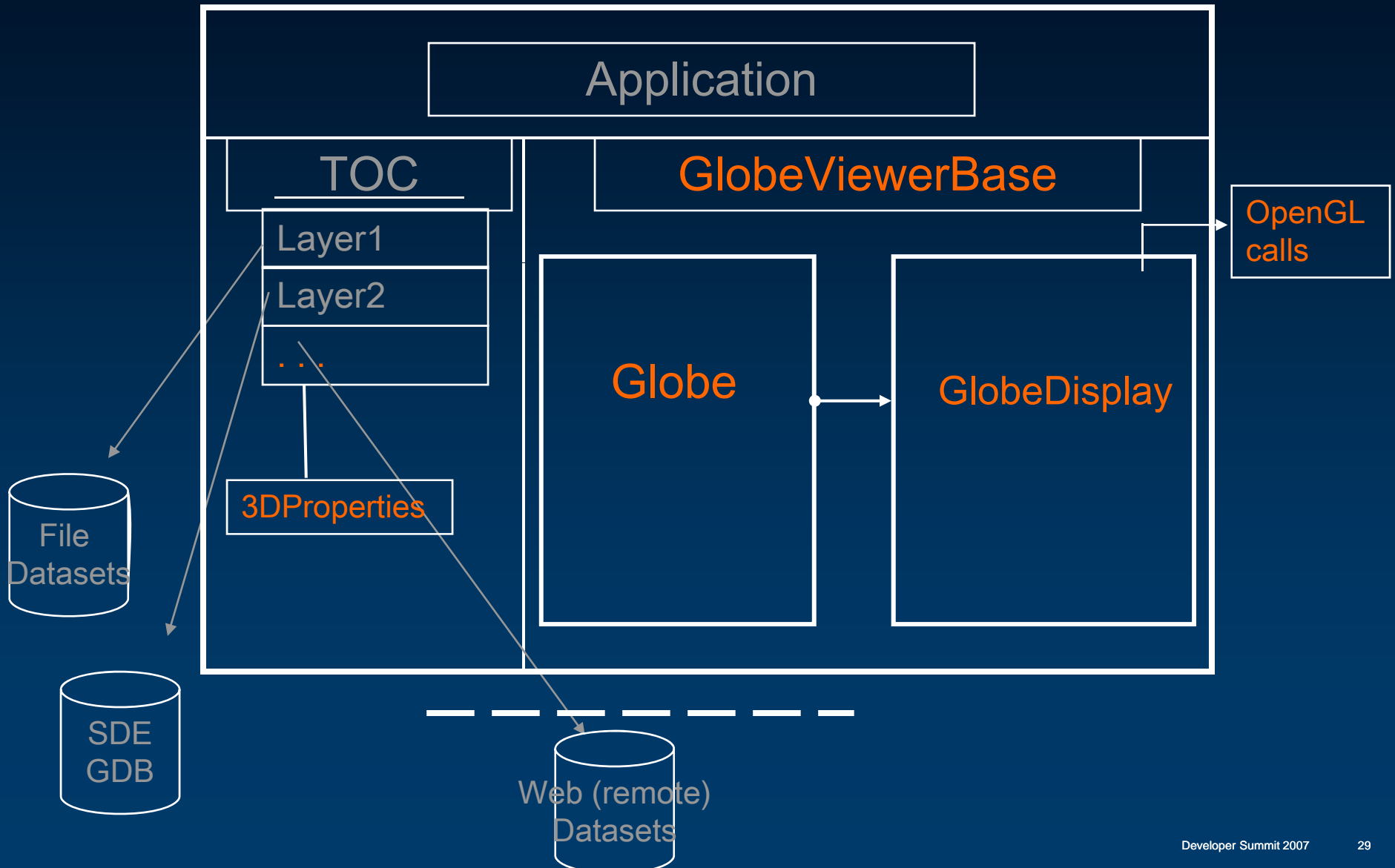
Lesson

- Group layer caches can improve performance, especially for mature, fully rasterized group layers.

Customization Framework

- All Globe based application share ArcGIS 9x customization framework:
 - **COM** components (ArcObjects), also accessible from **.NET** and **Java** APIs
 - *esriGlobeCore, esriGlobeCoreUI, esriArcGlobe*
 - As part of **3D Analyst**, shares the Multipatch (3D textured features), 3D Symbols, Video Exporters and Animation objects
 - Embedded Visual Basic for Applications (**VBA**)
- **Globe ActiveX Control**
- **Cross-platform development**
 - Available on Windows, Linux, and Solaris

Typical Globe Based Application Layout



3D Rendering

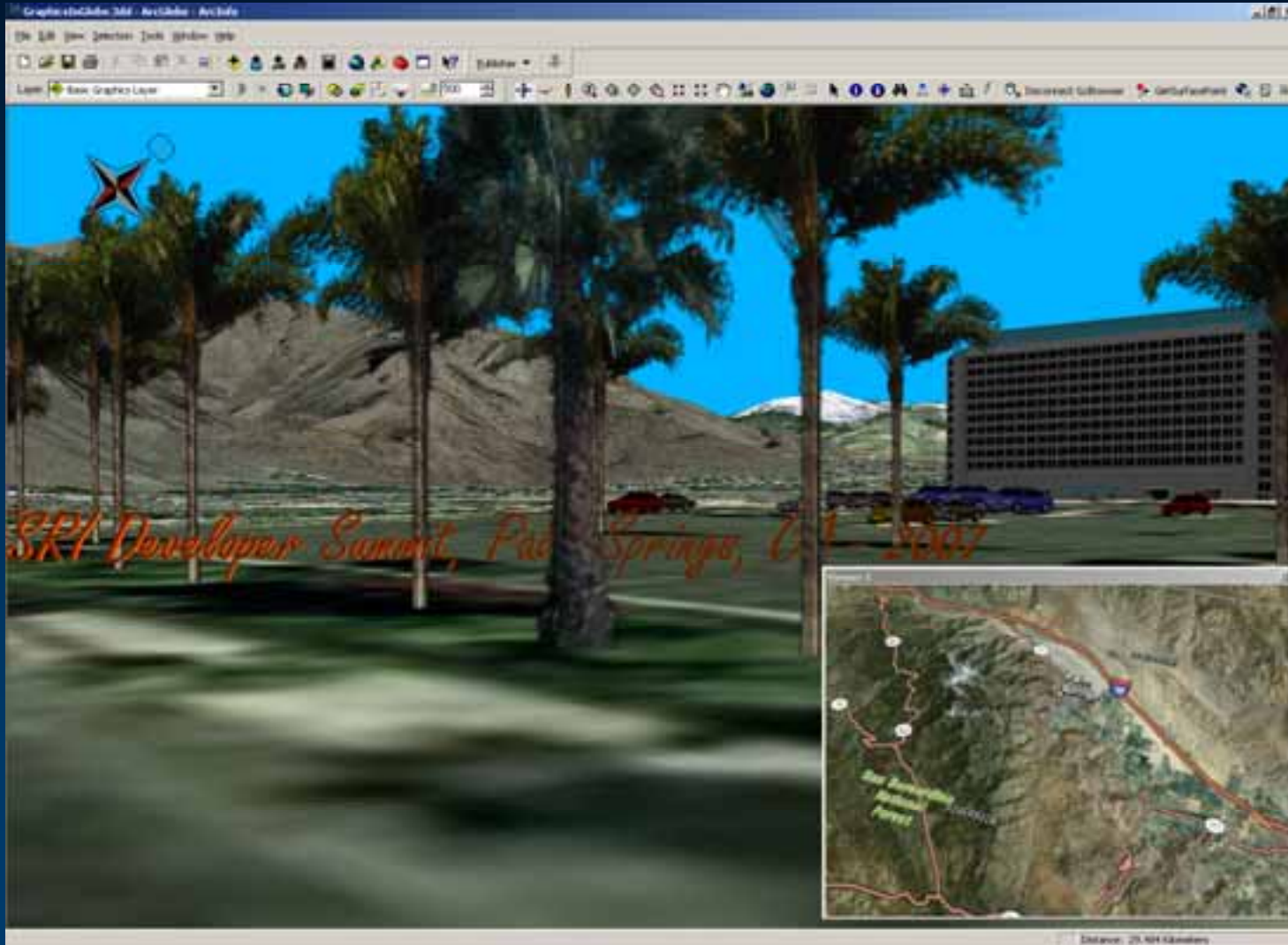
- **Out-of-the-box functionality**
- **Animation Framework**
- **Globe 3D Graphics**
 - **Point**
 - **Line**
 - **Polygon**
 - **Text**
 - **3D MarkerSymbols**

3D Rendering APIs

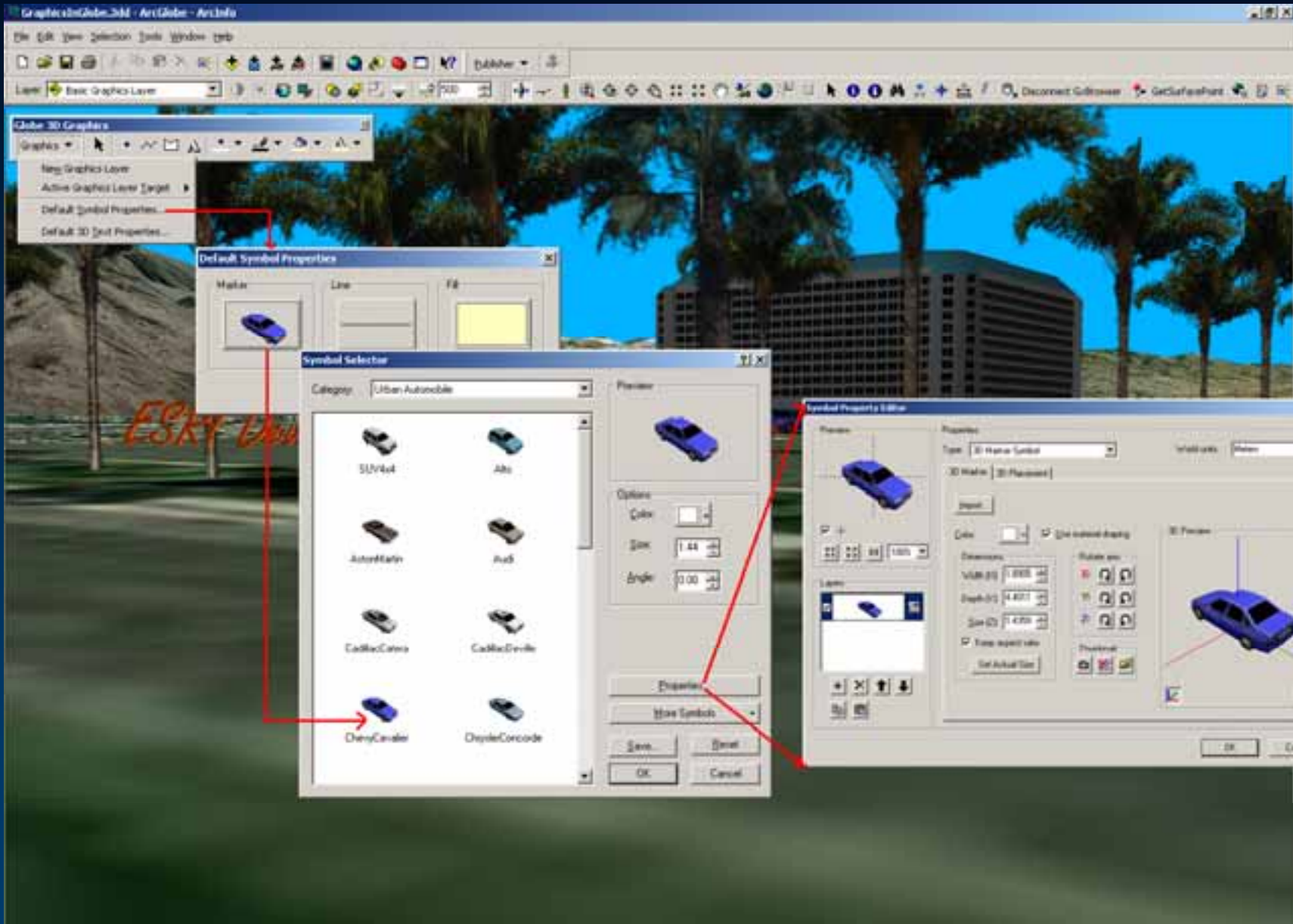
- **GlobeGraphics API**
 - GraphicsLayer
 - GraphicsElement
- **Globe API**
 - 3D Symbols
 - Coordinate conversion utility
 - CustomGlobeLayer
- **OpenGL API**
 - Globe framework provides mechanism to plug-in OpenGL calls

3D Rendering - Using GlobeGraphics

- Out-of-the-box functionality to create 3D Scenes



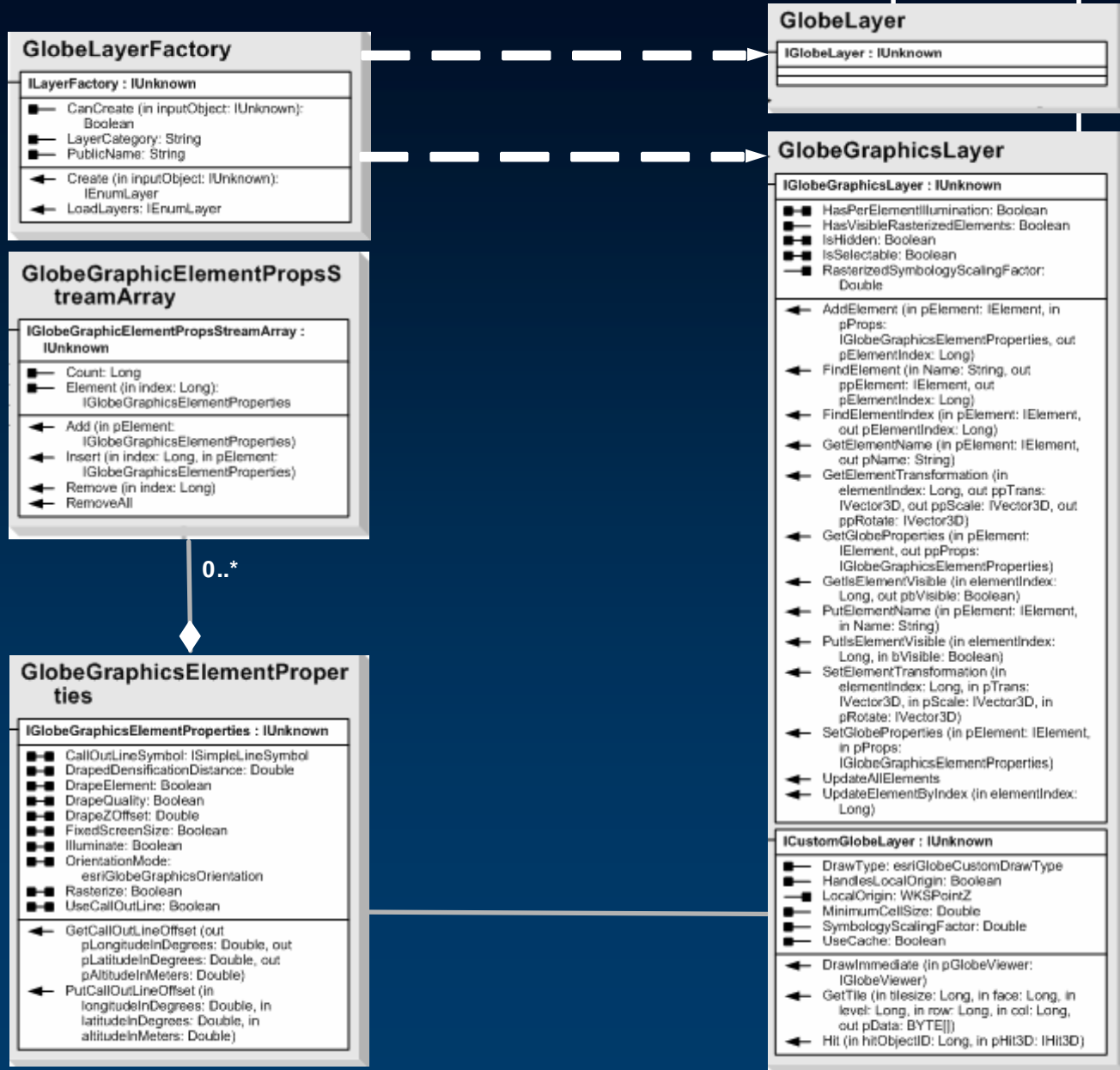
3D Rendering – GlobeGraphics Toolbar



GlobeGraphicsElement

- **Element types**
 - **PointElement**
 - **LineElement**
 - **PolygonElement**
 - **MultiPatchElement**
 - **TextElement3D**
- **Creating Element**
 - **Setting Element Geometry and Properties**
 - **Symbolizing Element**
 - **Adding Element to the Graphics Layer (Container)**

Globe Graphics Object Model



GlobeGraphicsLayer API usage (C++)

```
//Create a new graphics layer
m_ipGlobeGraphicsLayer.CreateInstance(CLSID_GlobeGraphicsLayer);
ILayer* (m_ipGlobeGraphicsLayer)->put_Name(L"MyGraphicsLayer");

//Add the new graphic layer to the globe
IGlobe* ipGlobe;
m_ipGlobeDisplay->get_Globe(&ipGlobe);
IScene* (ipGlobe)->AddLayer(ILayerPtr(m_ipGlobeGraphicsLayer), VARIANT_TRUE);
```

```
//Activate the new graphics layer
IScene* (ipGlobe)->ActiveGraphicsLayer(ILayerPtr(m_ipGlobeGraphicsLayer));
```

GlobeGraphics Element with 3D Symbols (C++)

//Create the element's geometry

```
IPoint* ipPoint(CLSID_Point);  
IZAware* (ipPoint)->put_ZAware(VARIANT_TRUE);  
ipPoint->PutCoords(position.longitude, position.latitude);  
ipPoint->put_Z(position.altitude);
```

//Create the element's color (red)

```
IRgbColor* ipColor(CLSID_RgbColor);  
ipColor->put_Red(255L);  
ipColor->put_Green(0L);  
ipColor->put_Blue(0L);
```

//Set the element's symbol

```
IMarkerSymbol* ipMarkerSymbol(CLSID_SimpleMarker3DSymbol);  
ISimpleMarker3DSymbol* (ipMarkerSymbol)->put_Style(esriS3DMSSphere);  
ISimpleMarker3DSymbol* (ipMarkerSymbol)->put_ResolutionQuality(1.0);  
ipMarkerSymbol->put_Size(700.0);  
ipMarkerSymbol->put_Color(IColorPtr(ipColor));
```

//Create the new marker symbol element

```
IElement* ipTrackElement(CLSID_MarkerElement);  
IMarkerElement* (ipTrackElement)->put_Symbol(ipMarkerSymbol);  
ipTrackElement->put_Geometry(IGeometryPtr(ipPoint));
```

//Add the graphic element to the graphics layer

```
IGraphicsContainer* (m_ipGlobeGraphicsLayer)->AddElement(ipTrackElement);
```

Demo

- **Creating and Using Graphics**
- **Designing a Globe Application**

3D Rendering - Using Globe API

- **Directly draw geometry using selected symbol through IDisplay, IDraw, IGlobeDisplay3**
- **Control the rendering as needed**
 - **Camera tracking moving object**
- **Use custom layers to render**
 - **Setting Element Geometry and Properties**
 - **Symbolizing Element**
 - **Adding Element to the Graphics Layer (Container)**

Custom Rendering using Custom Layers

- **Significantly improved at ArcGIS 9.2.**
- **No need to listen to GlobeDisplayEvents**
- **OpenGL can be directly called inside the method DrawImmediate()**
- **Provides Hit-Test mechanism for picking and selection in 3D**
- **Provides mechanism to get around floating point rendering precision in OpenGL**

Custom Rendering using Custom Layers

- Your custom layer needs to implement the following four interfaces at the minimum:
 - *ESRI.ArcGIS.GlobeCore.ICustomGlobeLayer*,
 - *ESRI.ArcGIS.Carto.ILayer*,
 - *ESRI.ArcGIS.Geodatabase.IGeoDataset*,
 - *ESRI.ArcGIS.Carto.ILayerExtensions*
- Optionally implement persistency if your layers need to be saved (recommended)
- Optionally implement *ILayerEffects* and create a *Feature3Dproperties* extension to support layer effects

Custom Rendering using Custom Layers

- The supported draw types are (at version 9.2)
esriGlobeCustomDrawRasterize,
esriGlobeCustomDrawByTile,
esriGlobeCustomDrawOpenGL,
esriGlobeCustomDrawVector,
esriGlobeCustomDrawOpenGLAndRasterize.
- To draw in OpenGL the draw type must be set to
esriGlobeCustomDrawOpenGL or
esriGlobeCustomDrawOpenGLAndRasterize
- Methods of IDisplay and IDraw can be mixed with
OpenGL calls when the DirectOpenGLDraw flag of the
IGlobeDisplay3 is set to True

Custom Rendering Using OpenGL

All Globe based application allow direct call to OpenGL API

When to use

- In Commands and Tools**
- Inside the BeforeDraw() and AfterDraw() events of IGlobeDisplayEvents**

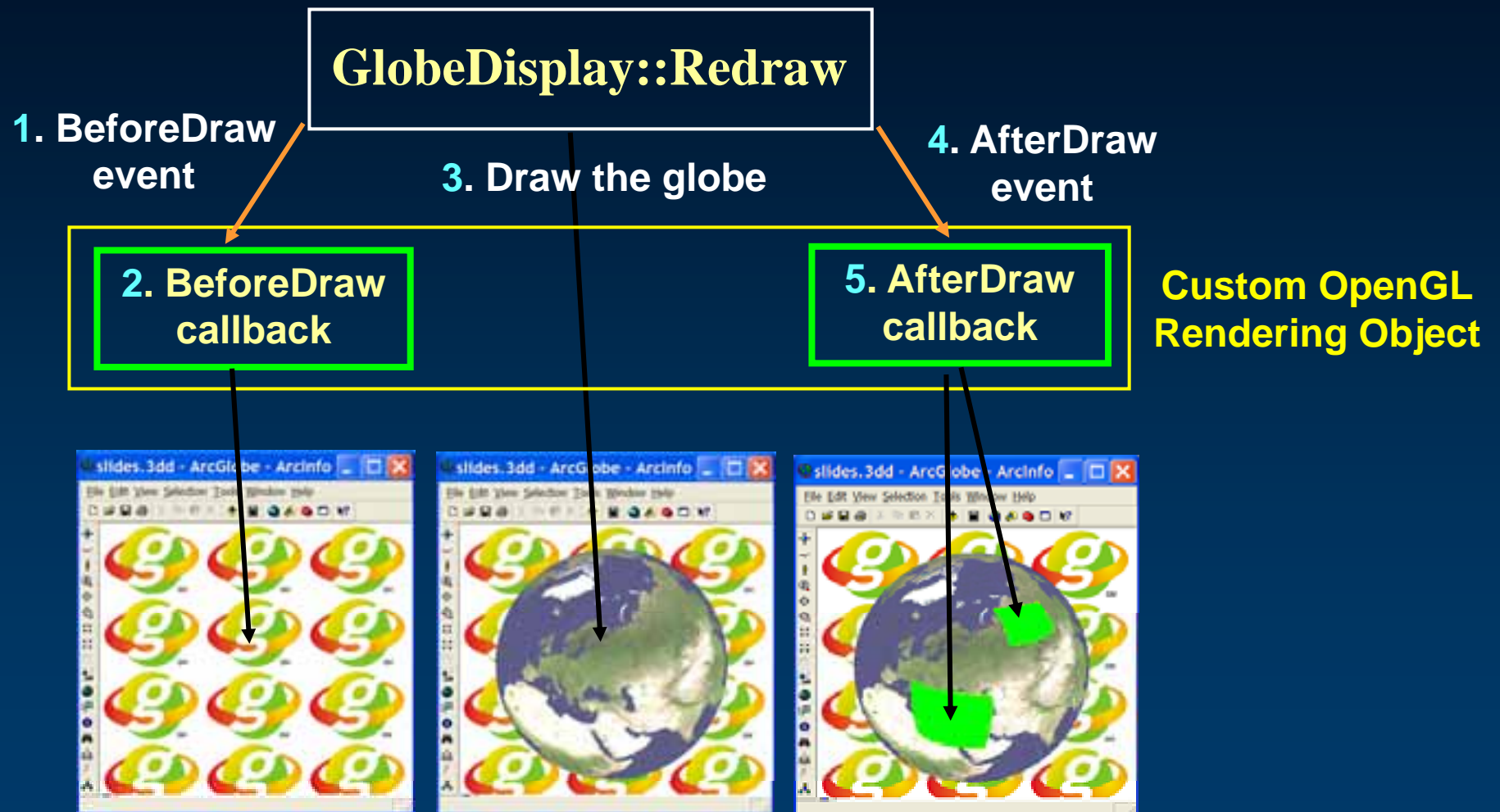
How to use

- Set the Display into Direct OpenGL mode**
- Set the symbol to the Display**
- Call the method to draw the geometry**
- Optionally call the OpenGL functions**
- Restore the Display mode**

Why use OpenGL

- **Add functionality not available on the UI**
 - Interactively displace/rotate graphic object
- **Render custom data sources**
 - Web Services, real-time feed, simulation
- **Add special effect to the globe display**
 - Simulate fire, smoke, rain, flood, wave, etc

Custom Rendering: how does it work?



Custom OpenGL Rendering (in C#)

```
public class MyCoolCommand : ESRI.ArcGIS.Utility.BaseClasses.BaseCommand
{
    private ESRI.ArcGIS.Controls.GlobeHookHelper m_globeHookHelper = null;
    private IGlobeDisplay m_globeDisplay = null;
    private IGlobeDisplayEvents_Event m_globeDisplayEvents = null;
}
```

```
public override void OnCreate(object hook)
{
    //...
    m_globeHookHelper = new GlobeHookHelperClass();
    m_globeHookHelper.Hook = hook;
    m_globeDisplay = m_globeHookHelper.GlobeDisplay;
    m_globeDisplayEvent = this.m_globeDisplay as IGlobeDisplayEvents_Event;
    m_globeDisplayEvent.AfterDraw +=
        IGlobeDisplayEvents_AfterDrawEventHandler(m_globeDisplayEvent_AfterDraw);
}
```

```
public void AfterDraw(ISceneViewer sceneViewer)
{
    //Use OpenGL API here
}
```

Server Based 3D Mapping

- **All geographic data is hosted on the server**
- **High performance globe services deliver data for viewing by 3D client applications**
 - ArcGIS Explorer, ArcReader, ArcGlobe, ArcEngine
- **Allows organizations to centrally manage their geographic data**
- **Allows end users and analysts to work with data in a 3D mapping environment**

ArcGIS Server 9.2 : Globe Services

- **Globe Services are new functionality of ArcGIS Server at 9.2**
- **Globe Services are available both in ArcGIS Server for Microsoft .Net and ArcGIS Server for Java products.**
- **To be able to publish Globe Services you'll need to have ArcGIS Server with 3dserver license.**
- **You'll also need to have ArcGIS Desktop with 3D Analyst license to prepare the publishing Documents.**

What is a Globe Service ?

- **Ability to publish your 3D GIS data to the Web**
- **Efficient streaming of globe visualization pages, animation and ability to perform spatial queries on 3D data.**
- **Provides a means for generating, managing and serving optimized globe data.**
- **Supports both just in time (on-demand) and pre-cached data serving models.**

What is a Globe Service ? Cont'd

- **All ArcGlobe supported data types can be served.**
- **Supports Identify, Search, and Find features**
- **Supports Local area (LAN) and Web based access**
- **A Globe Server is equipped to build a data cache, populate the cache on demand, or partially update the cache on request.**

Layer Types supported as a Globe Service

- **Elevation**
- **Imagery**
- **Rasterized 2D Vectors**
- **Rasterized 2D Text**
- **3D Vectors**
- **3D Graphics and Text (bill boarded)**

Demo

Custom Applications
Consuming Web Services
Embeddable Applications

Embeddable Architecture

2007 ESRI Developer Summit: Developers Guide to 3D Visualization in ArcGIS 9.2 (Best Practices) Listen as Tamrat Belayneh and Nathan Shephard, both 3D visualization product engineers, talk about important points to consider when developing 3D applications and discuss the best practices they will provide during their session at the upcoming ESRI Developer Summit.



Best Practices Guide

- **Calls to Locate() will result in additional calls to your AfterDraw() routine. This means that your AfterDraw() could be called from an unexpected location. The easiest way to prevent against this is to make sure you only execute the rest of your AfterDraw() when OpenGL is in GL_RENDER mode:**

```
{  
    int rmode;  
    GL.glGetIntegerv(GL.GL_RENDER_MODE, out rmode);  
    if (rmode != GL.GL_RENDER)  
        return;  
}
```

Best Practices Guide

- **Locate()** could be slow as it does a redraw in **GL_SELECT** mode, which means its speed varies on how complex your dataset is.
- **GlobeDisplay::GetSurfacePoint()** coupled with **GlobeDisplay::GetSurfaceElevation()** provides a good alternative.

```
{  
    void IGlobeDisplay.Locate(ISceneViewer pViewer,  
        int xView, int yView, bool bGetTransformed,  
        bool bGetEmptyGlobe, out ppPoint, out ppOwner);  
}
```

Best Practices Guide

- **Avoid calling methods on ArcObjects from different threads, unless you can guarantee that the object was created on that thread.**
- **Using '==' comparisons in a different thread than a creation thread with ArcObjects may not work as expected. Some objects (like Layer objects) are cloned several times, so if there isn't an appropriate Equals() override on that object, '==' will always return false because the underlying pointer addresses are different (because of the clone).**

Best Practices Guide

- **When controlling the GlobeCamera object in local mode, offset the camera by a little (for example use about 0.01 radian), when looking straight down. This will prevent the camera from running into uncontrollable spinning**

Best Practices Guide

- **When changing the enabled state of any OpenGL attributes always use `glPushAttrib` with the `GL_ENABLE_BIT` to save and `glPopAttrib` to restore the previous state. If you don't leave OpenGL in the correct state it can adversely affect the next draw cycle.**
- **Drawing the target position can be a useful visual cue, particularly when navigating underground. Make sure to use a local origin when doing so, otherwise it's likely to jump around quite a lot as you move.**

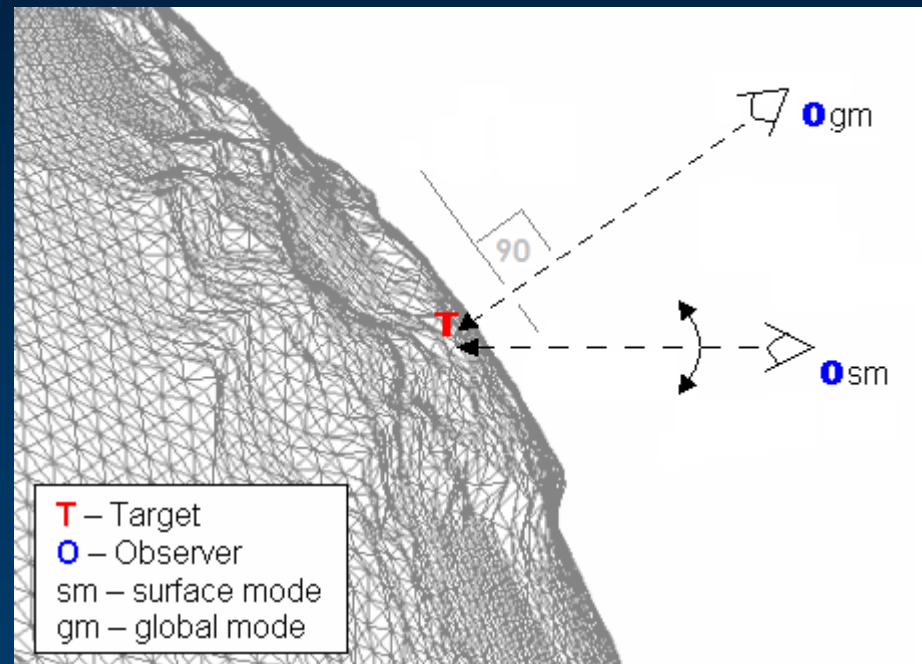
Best Practices Guide

- **Use custom Layers to extend the supported data types**
- **Some of the advantages a CustomGlobeLayer gives you are the control over properties such as DrawType, DrawMode and cache usage.**
- **The custom layers in globe can be classified according to the DrawType they implement, which defines the layers drawing mechanism.**
- **Use custom Layers for dynamically changing data.**

Best Practices Guide

Understand the relationship between Observer, Target and Surface

In **global** mode the target is located at the center of the globe at all times. In **surface** mode the target is typically positioned on the globe surface.



Appendices

Graphics card settings

- Enable **Anisotropic Filtering** for better texturing quality, avoiding 'snowy' artifacts in perspective view. Set it to the maximum supported by the card
- Limit **Anti-Aliasing** (use 2x or disable it). Although it reduces line and edge artifacts, it can reduce rendering performance
- Turn off **Vertical Sync** option for better performance

Appendices

Advanced registry settings

- **“Maximum internal viewport width” and “Maximum internal viewport height”**

ArcGlobe memory and computation demands increase with the size of the rendering window. This can make full screen views sluggish in high-definition displays. For this reason, there is an internal limit in to the application window (viewport) size used to calculate the rendering detail.

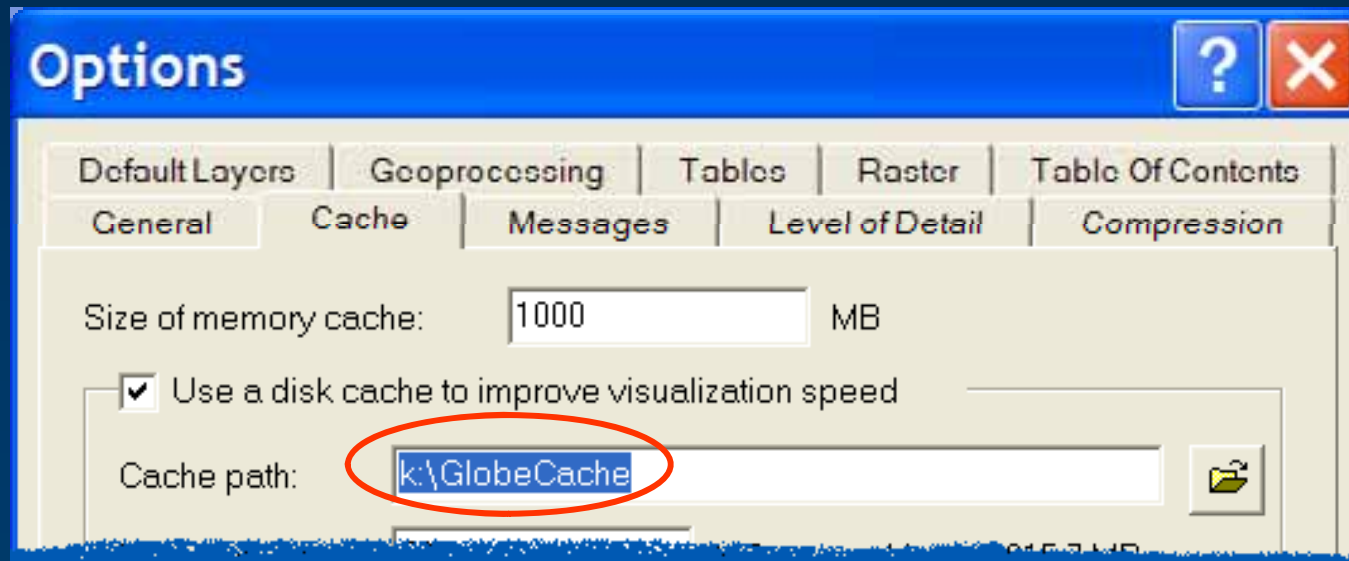
To make this limit more restrictive or more relaxed, edit the values in the registry.

Appendices

- **Data Caching Tips**
- **Cache data when smooth interaction / presentation is important**
- **When a visibility distance is set, only the visible levels need to be cached**
- **Save a layer file or document if you want to preserve the cache. If the layer is invalidated afterwards, it must be saved again**
- **Layer files are the best way to reuse and share caches safely**

Appendices

- **Cache Location Tip** : If you need to move or share caches, set a portable location:
 - **UNC paths** (e.g. `\\myServer\ArcGlobeCache`) are accessible from multiple machines
 - **Drive letters** (`k:\ArcGlobeCache`) allow moving caches easily by remapping the drive



Appendices

ArcGlobe/GlobeControl Dev Samples

<http://edn.esri.com>

- **ArcGlobe Visualization**
 - Samples : 3D Analyst : Visualization : ArcGlobe
- **GlobeControl: Getting Started**
 - Samples : Controls : Getting Started with the Globe Control
- **GlobeControl: Animation, Navigation, Effects**
 - Samples : Controls : Globe Control

Appendices

OpenGL Resources

- The OpenGL Homepage
 - <http://www.opengl.org>
- Microsoft OpenGL Reference
 - <http://msdn.microsoft.com/library/en-us/dnanchor/html/opengl.asp>
- NeHe (OpenGL Tutorial)
 - <http://nehe.gamedev.net/>
- Colin Fahey's C# OpenGL Wrapper
 - <http://www.colinfahey.com/opengl/csharp.htm>
- OpenGL for Visual Basic
 - <http://is6.pacific.net.hk/~edx/>