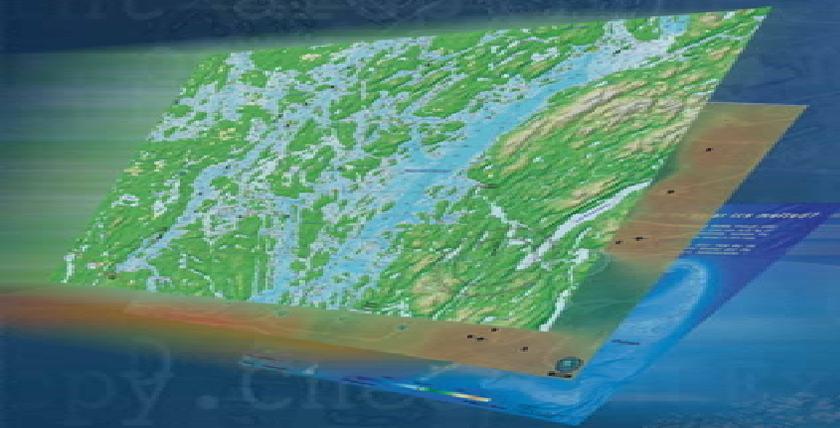


ESRI Developer Summit

March 22–25, 2010
Palm Springs, CA

Developer to Developer: ArcGIS 10 Dev Topics Display Performance Enhancements

Jeremy Wright
Craig Williams



What do we mean by “Display Performance”?

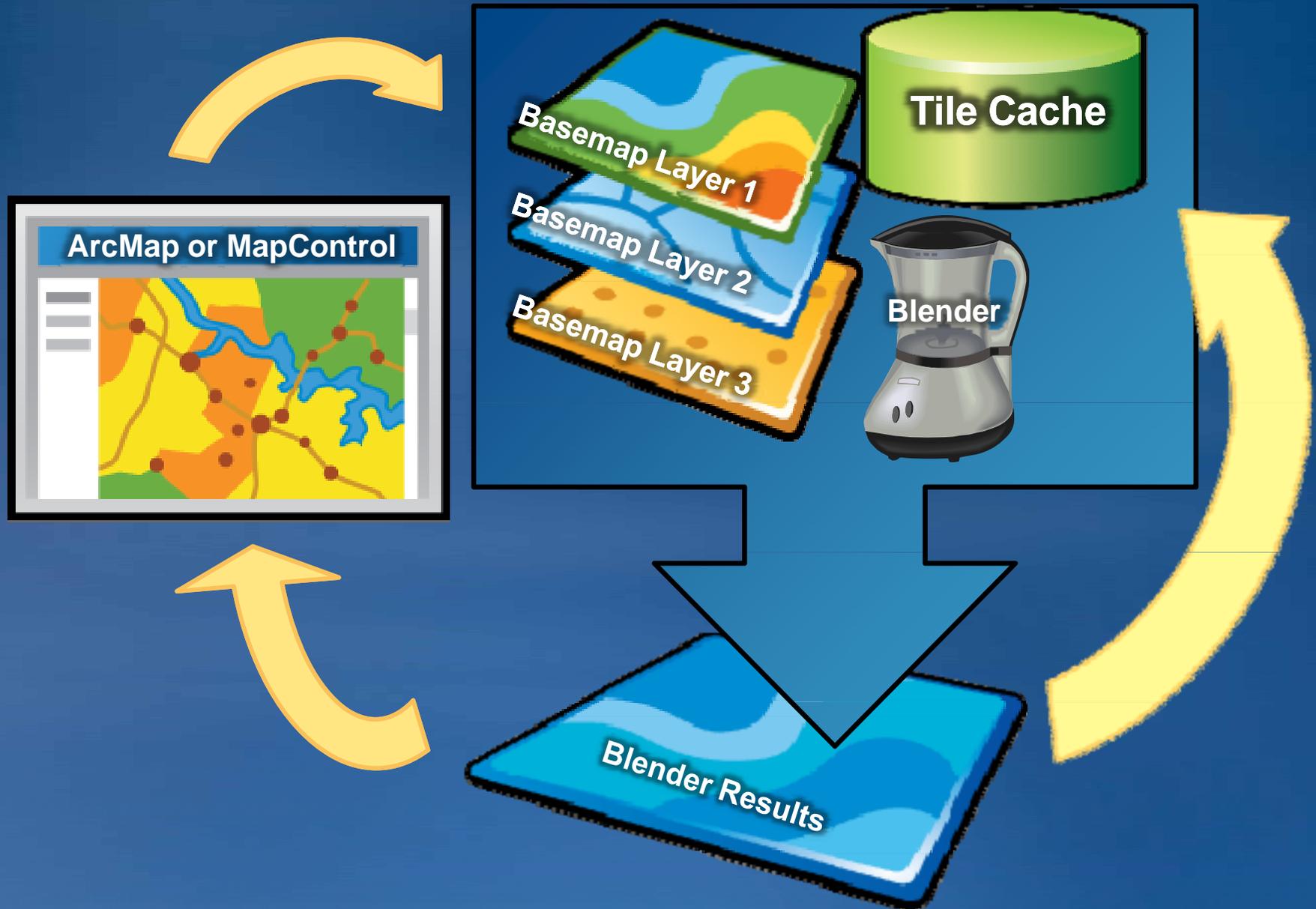
- “Speed” of drawing
 - Tough to measure in a GIS application
 - Can be dependent on data access
- Responsiveness to the user
 - UI locking
 - Application reaction time
- Frame Rate
 - Higher frame rates = smoother drawing
 - Higher frame rates can result in artifacts: “tearing”, “strobing”

Display Enhancements at ArcGIS 10

- **Basemap Layers:** a user- (or developer-) specified group of layers which displays continuously as you pan and zoom.
- **“QuickPan” navigation:** Used with Basemap Layers, QuickPan allows users to continuously navigate the map by moving their mouse in the direction they’d like to go.
- **GraphicTracker:** an easy-to-use, engine-friendly way to draw dynamic and animated items on a map or globe.
- **Dynamic Display:** existing developer-only experience for accelerated display.

Demo

Basemap Layer Drawing Engine



Panning or zooming a map with Basemap Layers

- As the user pans or zooms, areas are invalidated.
- If any portion of this area exists in the tile cache, it is drawn straight from the cache.
- If no cache tiles exist for the invalidated area, draw requests are submitted to each Basemap Layer's rendering threads.
- Asynchronous rendering threads process the draw requests and write them to that layer's screen cache.
- The blender progressively draws the results of the blended screen caches to the screen until the draw is complete, then copies the tile to the tile cache for reuse.

Basemap Layer Features: Shaders

- **IShader: A mechanism to modify RGB values based on a function.**
 - Implementation of shader defines processing for each band.
 - BasemapLayer creates a LUT to apply it for each pixel
- **IAdvancedShader: A mechanism to modify RGB values based on an image.**
 - Implementation of shader defines processing for each band, plus a method that receives an input image.
 - Processing of input image can modify the functions for RGB.
 - Does not update the passed-in image directly, creates LUT
- **These shaders do not invalidate the tile cache, so they have little effect on performance.**
- **Can be chained and reordered with no hit to performance.**

Basemap Layer Features: Dimming

- **Dimming allows your operational data to take prominence**
- **Dimming reduces color saturation, adds transparency**
- **Access via ILayerEffectDIM interface**

“How do I decide which ArcGIS display technology to use for my application?”

- **Basemap Layers:** as the name implies, use for Basemaps
 - Can be anywhere in the layer stack
 - Works with most core layer types*
 - See “[About creating and working with basemap layers](#)”

```
//Create a new Basemap Layer
IBasemapLayer bmLayer = new BasemapLayerClass();
//Cast to a Group Layer
IGroupLayer bmGroupLayer = bmLayer as
    IGroupLayer;
//Add a layer to the Basemap Layer
bmGroupLayer.Add(layer);
```

*use `BasemapSublayer::GetConversionStatus` to verify success.

“How do I decide which ArcGIS display technology to use for my app (cont’d)?”

- **GraphicTracker:** use when you need high-performance animation on a map or Globe
 - Works in conjunction with other Display technologies (Basemap Layers, Dynamic Display, etc)
 - Can have basic text
 - Each tracker can be updated independently
 - Refresh is handled automatically as you change each tracker
 - Also useful for adding easy to manage static graphics
 - Has some special considerations for projecting on the fly
 - See SDK topic “[Using a GraphicTracker](#)” to get started

“How do I decide which ArcGIS display technology to use for my app (cont’d)?”

- **Dynamic Display:** the fastest performer in many situations, but the most difficult to develop
 - Content must be rasterized to display efficiently
 - Not a plug and play solution
 - Don’t use over remote desktop session or Citrix
 - Some considerations to optimize performance
 - Reduce number of layers in the map
 - Simplify symbology
 - See [code gallery](#) slides and sample from “Leveraging Dynamic Display in ArcGIS Engine Applications”

“How do I decide which ArcGIS display technology to use for my app (cont’d)?”

- **RasterBasemapLayers: A *special type of Basemap Layer* for image analysis use cases:**
 - Optimized for seamless drawing of a single raster layer
 - Resource-intensive, so limit usage to 3-4 in a map
 - RasterBasemapLayer is not co-creatable, use a factory:

[C#]

```
private void CreateRasterBasemapLayer(IRasterLayer rasterLayer, IMap map)
{
    IRasterBasemapLayerFactory rasterBMLFactory = new RasterBasemapLayerFactoryClass
        ();
    IRasterBasemapLayer rasterBML = rasterBMLFactory.Create(rasterLayer);
    ILayer layer = (ILayer)rasterBML;
    layer.Name = layer.Name + " Accelerated";
    IAddLayersOperation addLayersOp = new AddLayersOperationClass();
    addLayersOp.SetDestinationInfo(0, map, null);
    addLayersOp.AddLayer(layer);
    addLayersOp.Name = "Add Accelerated Raster Layer";
    IOperation op = addLayersOp as IOperation;
    op.Do(); // Add the layer.
}
```

Best Practices

- **Don't use a solution if it's not required for your application.**
 - Basemap Layers + GraphicTracker is a good place to start
 - Add other technologies as needed, or for special purposes
 - No silver bullet!
- **Be aware of your target environment and platform.**
 - Dynamic Display won't work over remote desktop or Citrix
 - Dynamic Display does work well for resource-constrained systems that do have an OpenGL graphics card
- **Be aware of technology limitations.**
 - Dynamic Display doesn't work well for large numbers of layers
 - BasemapLayers not compatible with all layer types/properties.
 - GraphicTracker ::MoveTo method doesn't take into account projection on the fly.

Questions?