

使用.Net或Java进行ArcGIS Engine开发

罗雄

ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值

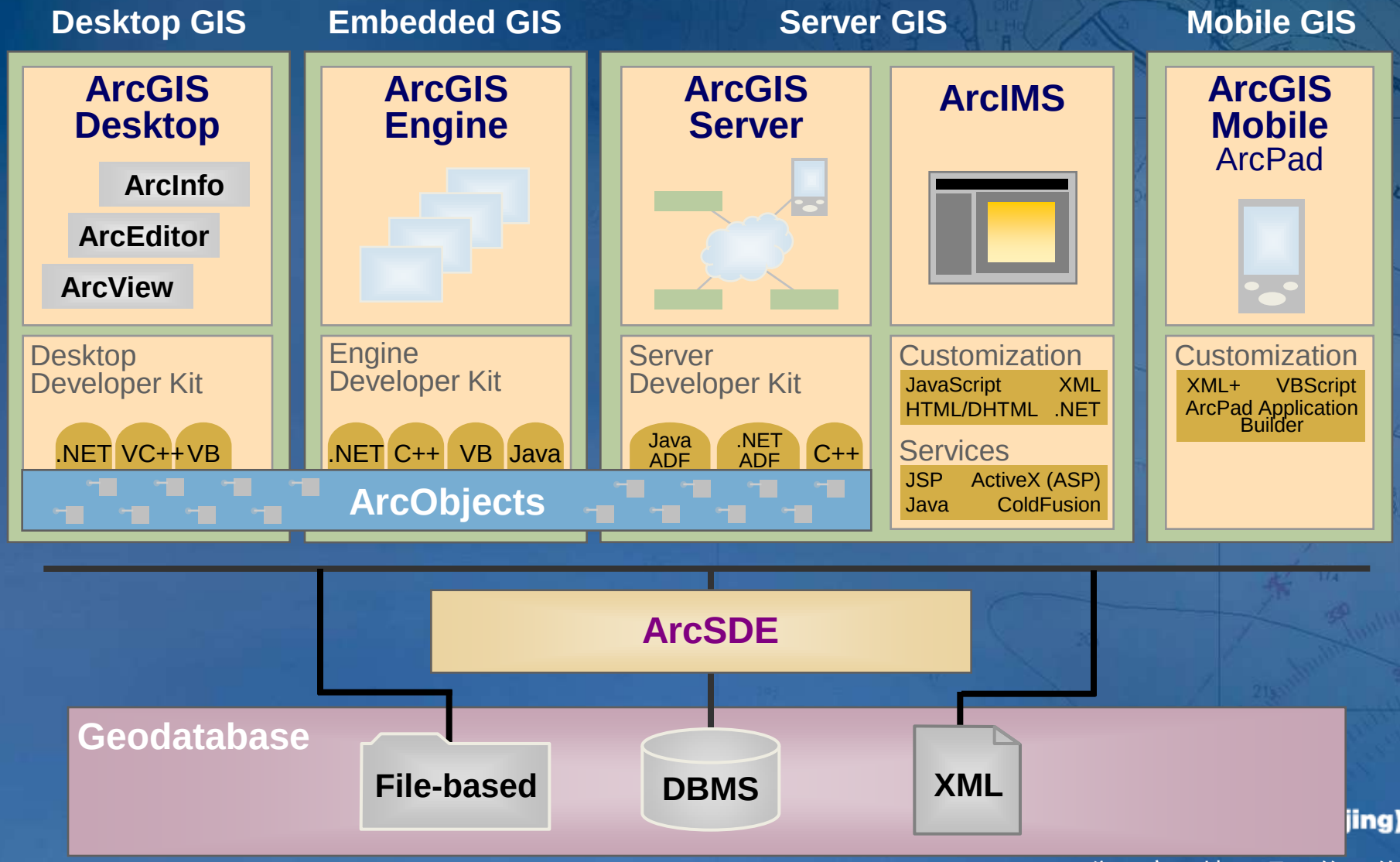
内容概要

- ArcGIS Engine 概览
- 基于.Net或Java构建Engine应用程序
- 版本迁移
- 高级功能开发

ArcGIS Engine 概览

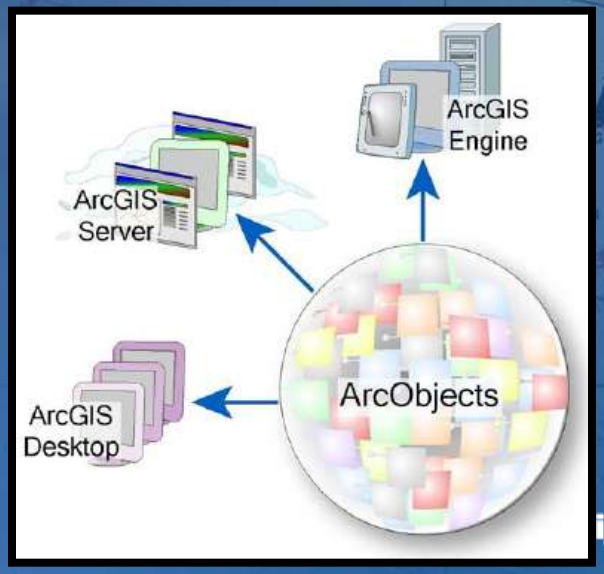
ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值

ArcGIS Overview

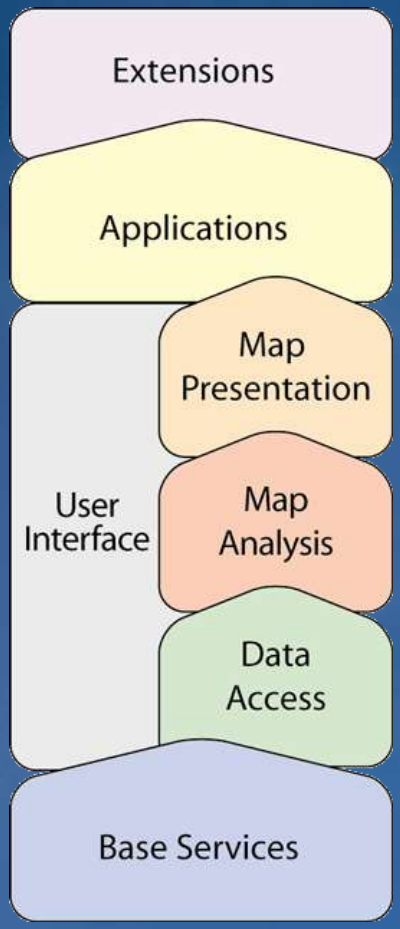


ArcObjects

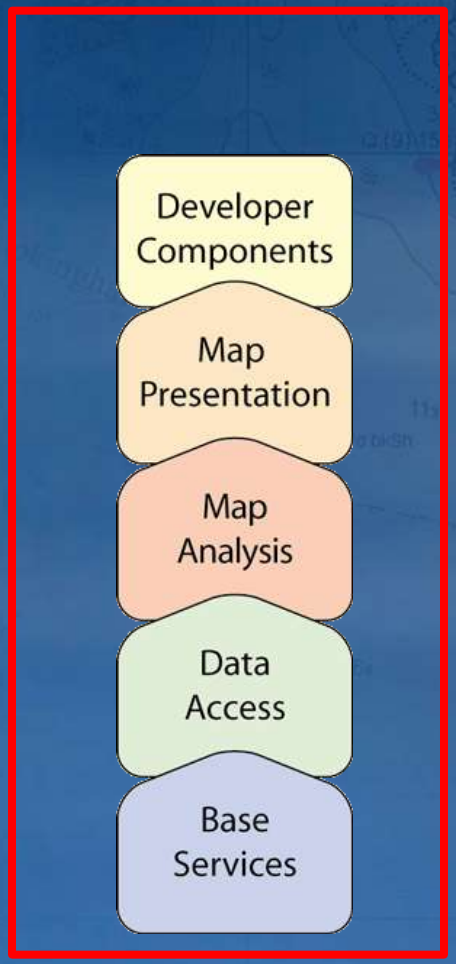
- ArcObjects是一套组件的统称
- ESRI用ArcObjects开发软件及应用
- 开发者可以使用ArcObjects来自定义软件或构建自己的应用系统
- ArcObjects是ArcGIS的核心



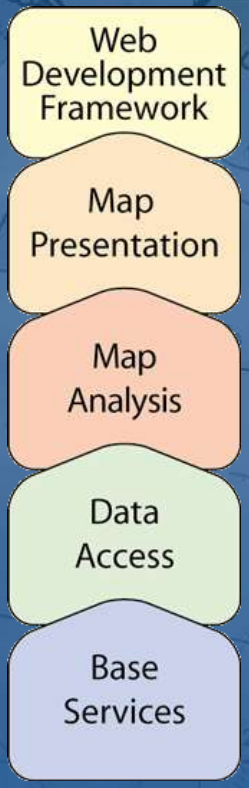
ArcGIS产品与ArcObjects



ArcGIS Desktop



ArcGIS Engine



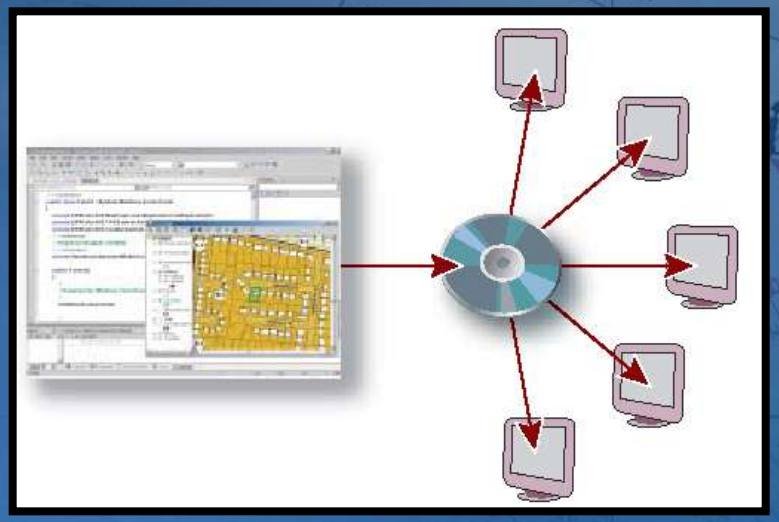
ArcGIS Server

ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值

ArcGIS Engine

- 用于构建定制应用的一个完整的嵌入式GIS组件库
- ArcObjects中的核心类库及组件是AO的一个子集

- 开发语言上支持
  - COM
  - .Net
  - Java
  - C++

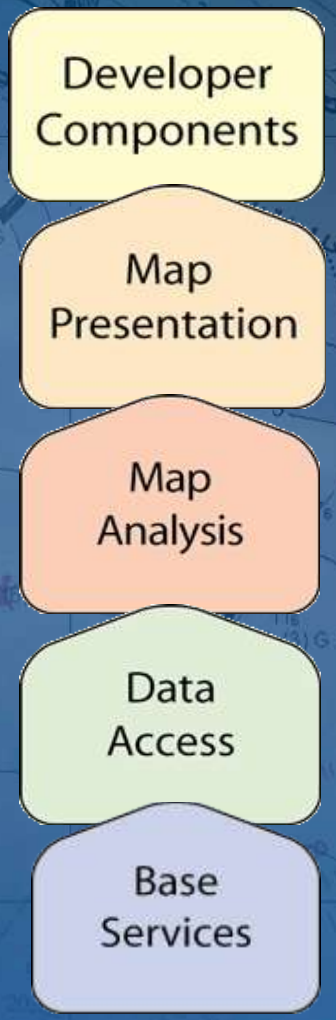


ArcGIS Engine 组成

- ArcGIS Engine Developer Kit
  - 开发者用于创建自定义的GIS和制图应用的开发包
- ArcGIS Engine Runtime
  - 运行自定义的Engine应用的可分发的ArcObejcts
  - ArcGIS Desktop 9.2后其中已经包含ArcGIS Engine Runtime
- 开发者与最终用户部署都需要安装

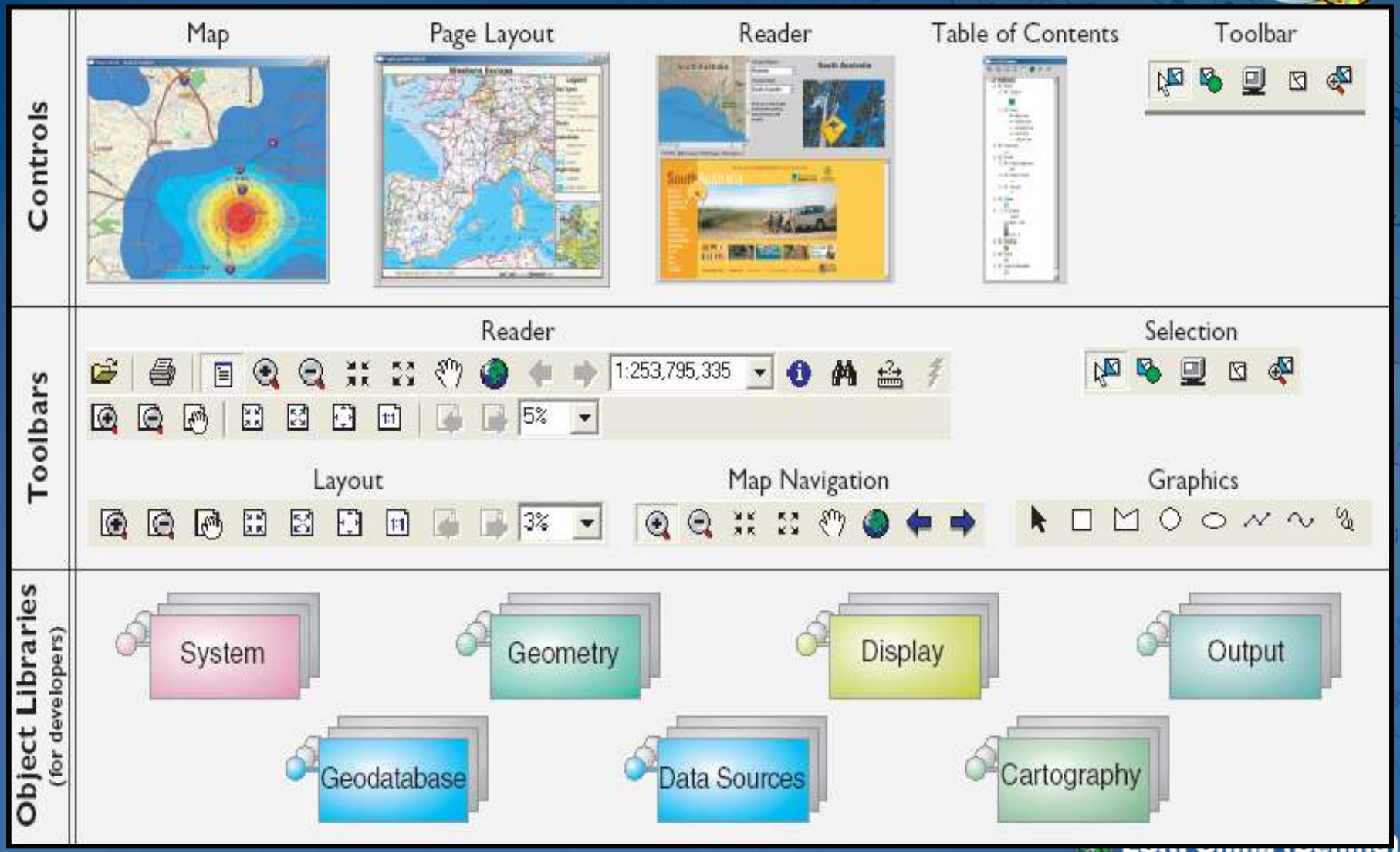
ArcGIS Engine的功能框架

- 基本服务
  - 应用层需要用到的ArcObjects核心对象
  - 几何图形的操作与显示
- 数据访问
  - 地理数据集，矢量，栅格数据
- 地图分析
  - 属性、空间查询，地理数据处理
- 地图显示
  - 渲染、标注、制图表达
- 开发组件
  - 各种应用层面上的Control及Commands&Tools



ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值

ArcGIS Engine总览



Geographic Thinking
分 享 地 理 价 值

ArcGIS Engine应用程序类型



独立运行的应用程序



后台非可视化运行程序



嵌入式应用程序

ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值

基于.Net或Java构建Engine应用程序

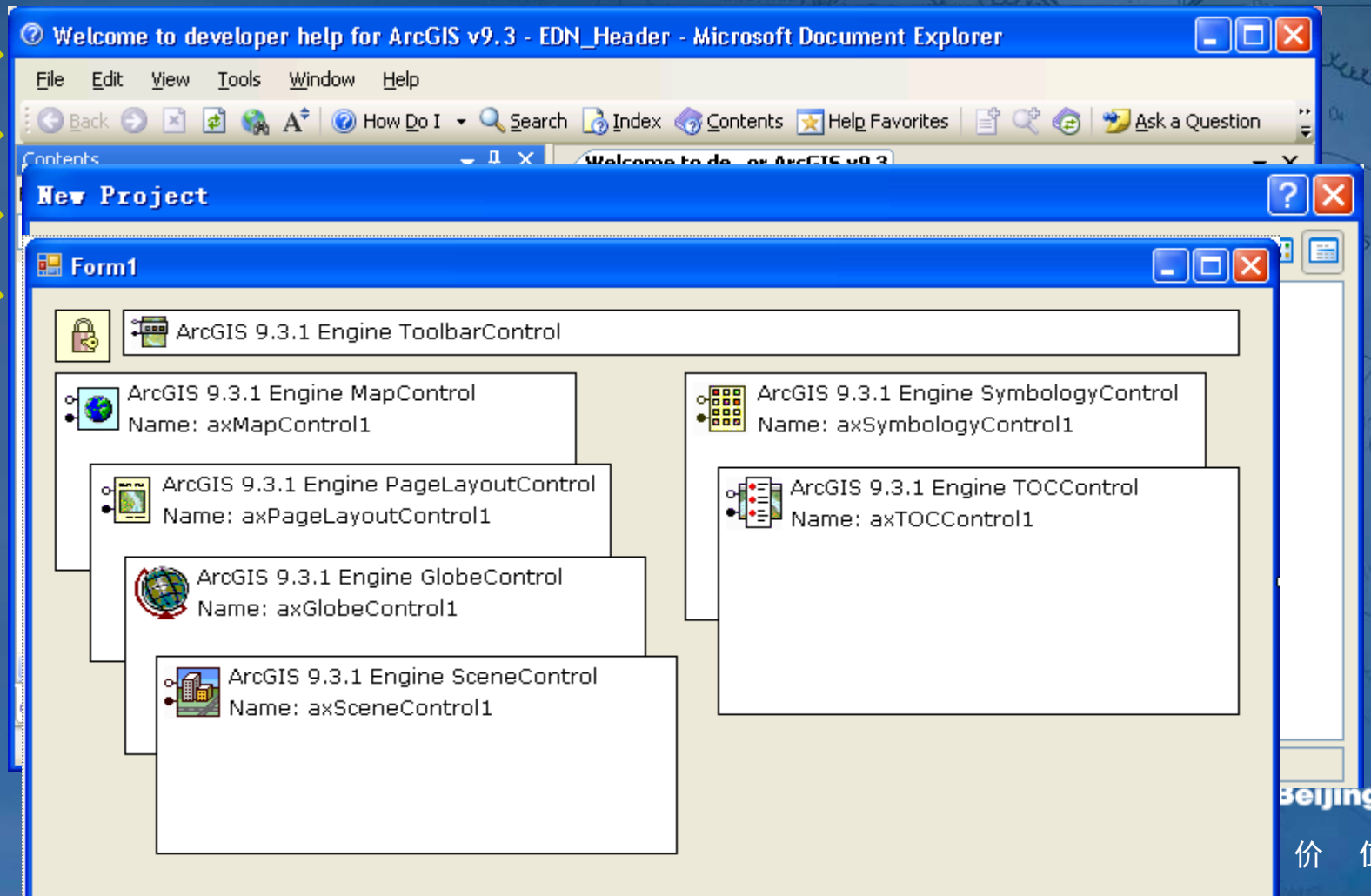
ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值

准备条件

- 安装Visual Studio
  - 2005或2008
  - .NET2.0 SDK和Runtime（3.0WPF）
- 安装ArcGIS Engine Runtime 和SDK for .NET
  - Runtime也需要独立注册许可
  - Runtime与SDK许可差别：是否有Designer
  - ArcGIS Desktop已经包含了Runtime

Engine for .NET SDK

- 帮助文档



构建WPF应用程序

- 什么是WPF？
  - 客户端展现技术
  - 极炫的可视化效果
  - 设计理念：用户界面和程序实现分离
- 在ArcGIS Engine开发中使用WPF
  - 将WPF用户控件放到AE应用程序中
  - 将AE控件放到WPF应用程序中



步骤

- 创建一个WPF应用程序
- 添加AE引用
- 将AE控件对象装载在WindowsFormsHost

```
globeControl = new AxGlobeContro
mapHost.Child = globeControl;
globeControl.BackColor = System.D
globeControl.BorderStyle = 0;
```



调用GP工具

- 引用 ESRI.ArcGIS.Geoprocessor
  - 所使用工具的引用，如ESRI.ArcGIS.AnalysisTools
- GP调用方式
  - 使用GP工具的名字

```
GP.Execute("kriging_3d", pArray, null);
```

  - 使用AE提供的GP类

```
SpatialAnalystTools.Idw idw = new
ESRI.ArcGIS.SpatialAnalystTools.Idw();
GP.Execute(idw, null);
```

  - GpDispatch反射方式调用

调用GP工具

- 调用GP的优越性
  - 高效率、方便易用
- Tips
  - 返回GP执行结果信息用于排查错误

```
private void ReturnMessages(Geoprocessor gp)
{
    if (gp.MessageCount > 0)
    {
        for                                          - 1; Count++)
        {
            MessageBox.Show(gp.GetMessage(Count));
        }
    }
}
string name = "SdeConnection";    //生成的sdeworkspace文件名
IWorkspaceName pWSName = pWF.Create(parentDir, name, pPropSet, 0);
```

```
int Geoprocessor.MessageCount
Return the number of messages of the last tool executed.
```

常见问题——内存释放

- Engine .NET API都是直接通过RCW(runtime callable wrapper)方式调用AO底层的组件，这些托管组件由CLR(Common Language Runtime)来管理，即通过GC(Garbage ollection)机制自动回收;

- AE for .NET释放方法
  - ESRI.ArcGIS.ADF.ComReleaser.ReleaseCOMObject(comObject);
  - ESRI.ArcGIS.ADF.COMSupport.AOUninitialize.Shutdown() ;
- CLR中的释放方法
  - System.Runtime.InteropServices.Marshal.ReleaseComObject(obj);

常见问题——组件注册

- 开发环境
  - 工程属性设置Register for COM interop

```
private static void ArcGISCategoryRegistration(Type registerType)
{
    string regKey = string.Format("HKEY_CLASSES_ROOT\\CLSID\\{{{0}}}", registerType.GUID);
    ControlsCommands.Register(regKey);

}
```

- 部署环境
  - C:\Windows\Microsoft.NET\Framework\v2.0.50727
  - regasm my.dll /tlb /codebase
  - regasm my.dll /unregister

常见问题——多线程

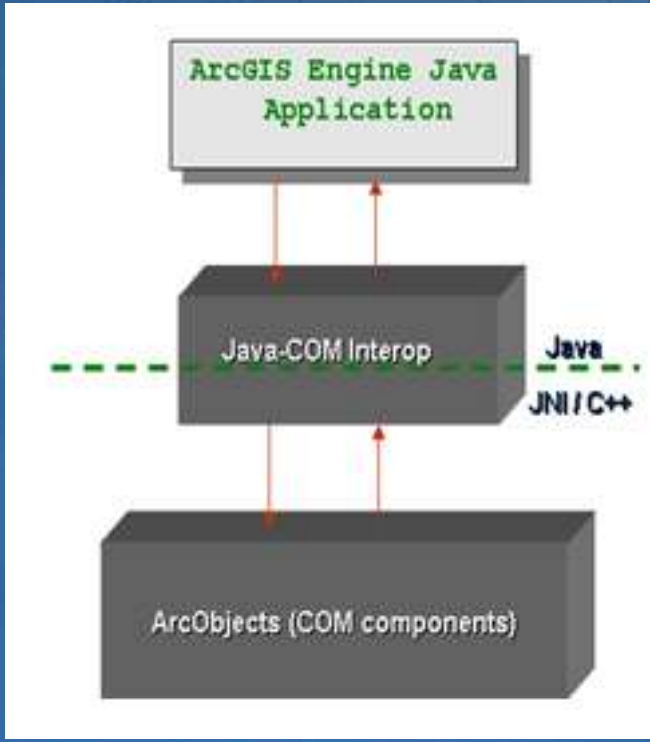
- AE中的COM组件都是STA线程模式的(强线程相关性），跨线程使用AE的Com对象会有问题；
- 强烈不推荐在AE中采用多线程操作AE对象，因此一般将一些独立的耗时较长的数据处理操作放在子线程（后台线程）中处理；
- DrawShape在子线程中绘制
  - 将AE对象作为UI控件来处理；
  - 在主线程中建立事件的委托代理，在子线程中采用Invoke方法触发；

Engine for Java开发

- 环境配置
  - Engine SDK、Runtime/JDK 1.5以上、开发工具（如Eclipse）
  - Java环境变量
    - Linux
      - setenv JAVA_HOME "/mymachine/java/jdk1.6.0_03"
      - setenv PATH $JAVA_HOME/jre/bin:$JAVA_HOME/bin:$PATH
      - 设置ARCGISHOME等环境变量：source init_java.sh
    - Windows
      - JAVA_HOME、JRE_HOME、PATH

理解AO的Java Interop

- ESRI的Java COM Interop提供了ArcObjects的Java API以及用Java语言创建和使用ArcObjects的运行基础

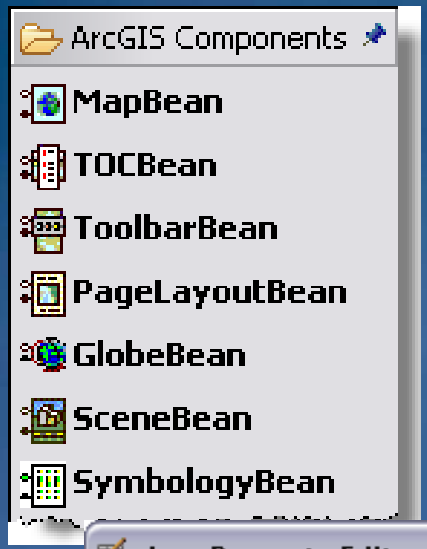


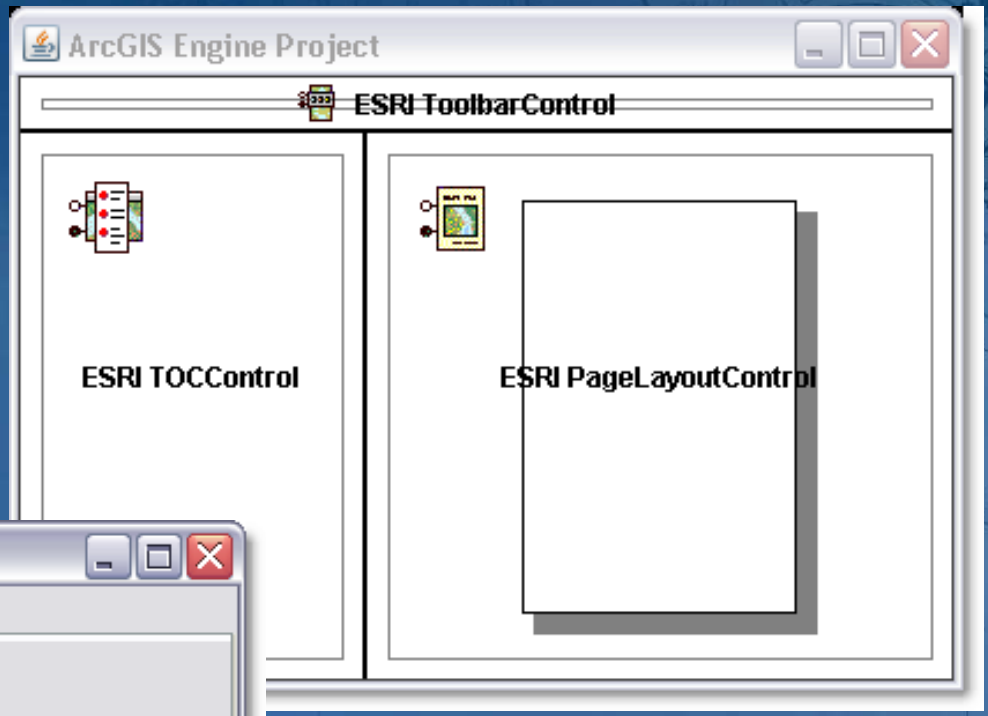
构建Java应用程序

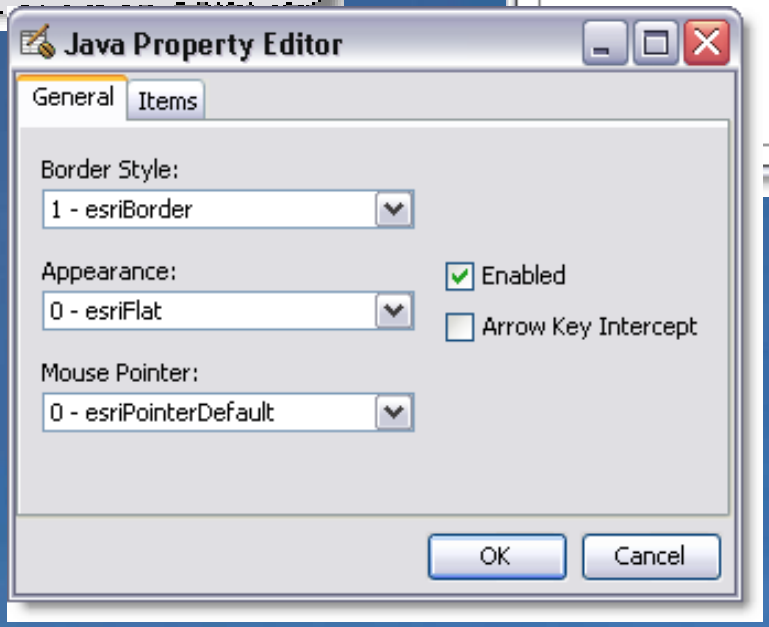
- 控制台程序
  - **EngineInitializer. initializeEngine();**
- 有用户界面的程序
  - **ArcGIS Visual Java Beans： Java-COM interop以AWT（Abstract Window Toolkit）控件的方式呈现给Java开发人员；**
  - **初始化Java Interop环境：**

    EngineInitializer.initializeVisualBeans();

使用Visual Java Beans







ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值

Java中常见问题

- 接口查询
  - 不是所有的AO类都支持接口转换；
  - 有单参数的构造函数或代理类接口IxxxProxy才行

```
IFeatureClass fc = fw.openFeatureClass("name");
FeatureClass featureClass = new FeatureClass(fc);

IFeatureClass fc = fw.openFeatureClass("name");
IGeoDataset gds = new IGeoDatasetProxy(fc);
```

Java中常见问题

- Engine控件（MapBean、ToolbarBean等）始终位于界面的最上面，它遮挡了菜单和下拉框
  - 原因：Engine Java的控件都是基于AWT构建的heavyweight 组件，所以和swing的lightweight叠加时会产生覆盖；
  - 解决办法：

    在程序初始化的时候，取消lightweight的支持。

    JPopupMenu.setDefaultLightWeightPopupEnabled(false);

Java中常见问题

- 实现ICommand或ITool接口， Bitmap和Cursor该如何获取？
  - AE for Java SDK提供了基础类BaseCommand和BaseTool，我们可以直接继承自它们
  - 直接获取picture或cursor

```
BasicRasterPicture brp = new BasicRasterPicture();
IPicture pic = brp.loadPicture(“path", “xx.bmp") ;
handle = pic.getHandle();
ISystemMouseCursor cursnew SystemMouseCursor();
curs.loadFromFile( “path", “xx.cur");
```

版本迁移

ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值

版本更新

- AE升级到高版本
  - 移除AE引用后重新添加新的引用；
  - 编译工程，修改提示错误的地方；
- 开发平台升级
  - 直接用平台提供的update向导将应用程序转换到高版本即可；
  - 如Engine 93 for VS2005升级到VS2008；

VB6或VBA迁移到.NET

- 为什么要迁移？
  - ArcGIS9.4不再使用VBA，推荐用Python；
  - 9.4版本不再有VB6 SDK；
  - 微软的.NET开发平台是大势所趋；
- 两类开发语言的差异
  - 对象类型、事件处理、接口查询、Win32 API的使用方式等均有差别；
  - Commands或Tools的Bitmap属性获取方式

    VB6——返回esriSystem.OLE_Handle

    .NET——返回Int型，常用Bitmap.GetHbitmap（）

如何迁移？

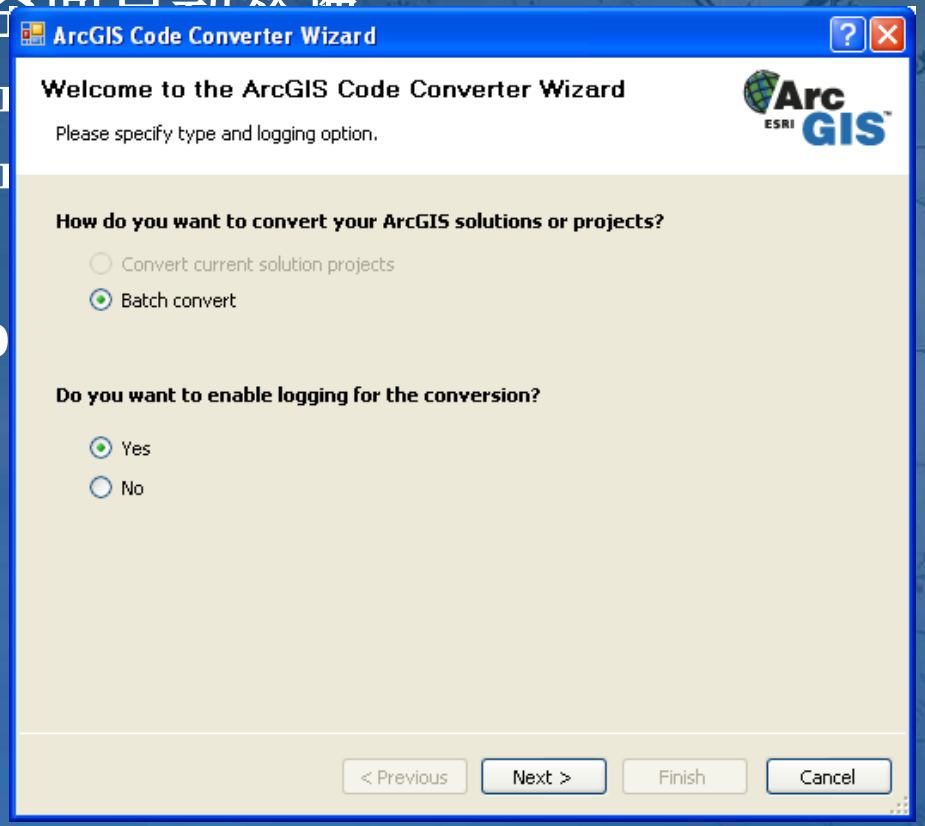
- Microsoft
  - Visual Basic Upgrade Wizard



ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值

如何迁移？

- ESRI-ArcGIS Code Converter；
  - ArcGIS8.x到9.x命名空间自动替换
    - Dim pPoly As ESRI.Ar
    - Dim pPoly As ESRI.Ar
  - 更新dll引用
    - 如用ESRI.ArcGIS.AD

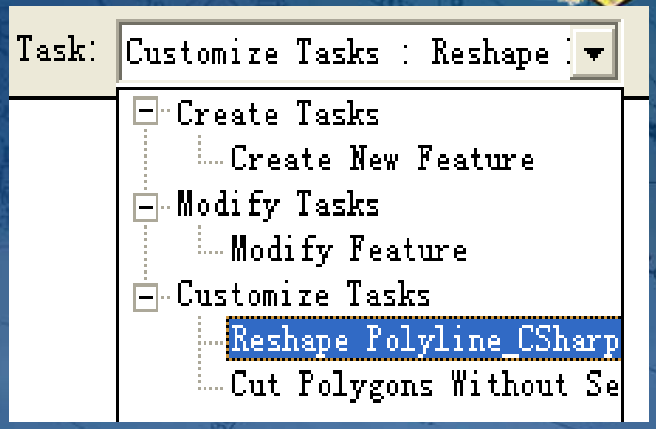


高级功能开发

ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值

新的Editor框架使用

- 可扩展的Editing框架
  - Engine Editor核心对象的开放
    - Start / Stop Edit Sessions
    - Editable Layers
    - Control of Edit Operations
    - Editing Events
  - 自定义Editor Task
  - 自定义Snapping环境
  - 以Vertex Commands为代表的更多的编辑命令




```
private void OnSketchFinishedMethod()
{
    IEngineEditLayers pEditLayer = m_EngineEditor as IEngineEditLayers;
    IFeatureLayer pFeatureLayer = pEditLayer.TargetLayer;
    IEngineEditSketch pEditSketch = m_EngineEditor as IEngineEditSketch;
    bool abortEdit = false;                    //是否要放弃编辑
```

ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值

新的Editor框架改进

要 点	Engine 9.2	Engine 9.3
查看编辑状态	并不直接支持，需查看TargetLayer是否为空	IEngineEditor:EditState
得到当前正在编辑的Workspace对象	并不直接支持，需通过TargetLayer的IDataset的Workspace得到	IEngineEditor:EditWorkspace
设置被编辑图层	不支持	IEngineEditLayers:SetCurrentLayer
设置当前的Edit Task	不支持	IEngineEditor:CurrentTask
创建自定义Edit Tasks并添加到Task Control	不支持	Implement IEngineEditTask
监听编辑事件OnFinishSketch	不支持	IEngineEditEvents
得到Edit Sketch	不支持	IEngineEditSketch:Geometry
向OperationStack 中添加edit operations (ToolbarControl)	不支持	支持

Editing 框架演示

ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值

Dynamic Display

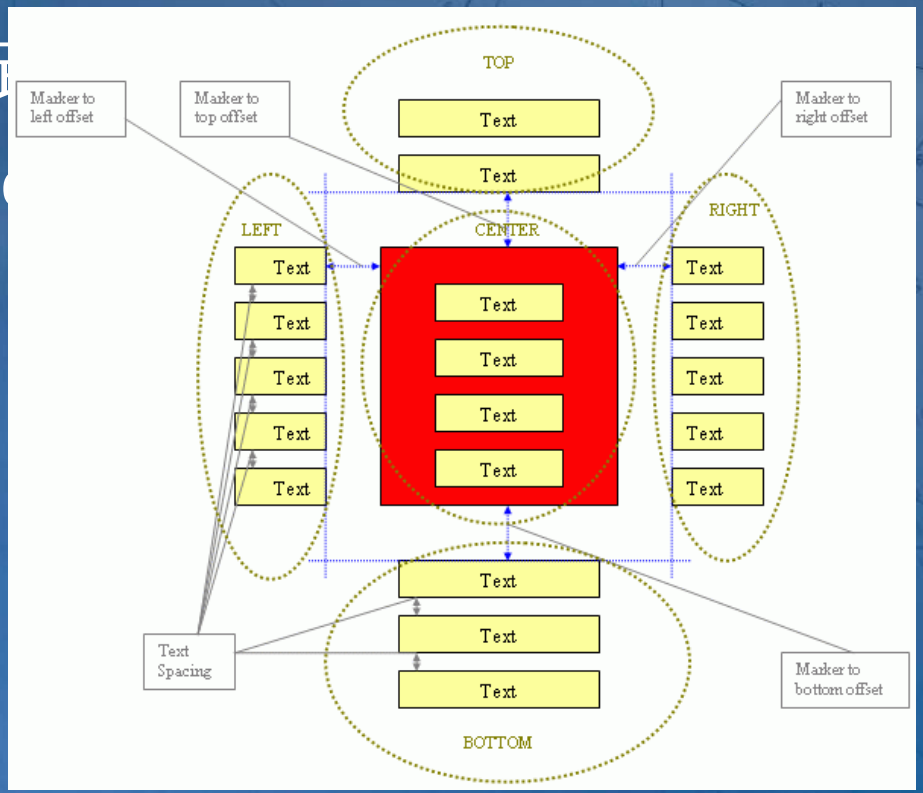
- ArcGIS一种新的动态显示数据或对象移动的技术
- 几乎实时的显示监测目标的地理信息的需求催生了Dynamic Display技术
- Dynamic Display——Engine 9.3
  - 频繁的刷新速率下显示地理信息
  - 快速几乎无间隙的更新速度
  - 平滑的、连续的缩放、旋转Dynamic图形
  - 几乎实时的显示性能

Cache方面的改进

- 在Dynamic显示状态下Cache的使用及重用
- 可用程序事先生成
- 并可具体指定创建Cache的区域
- 与ArcGIS Server的Cache结构相同并可互用
- 加强了对Cache的管理

Text 方面的改进

- 支持Text Box
- 对从右到左书写顺序的语言良好支持
- 普遍支持了在Marker周围绘制文本的功能
- 加强了对文本之间的间距
- 在屏幕上绘制Compound
- 线型的连续性



其他方面的改进

- 在Dynamic Display状态下选择要素
- 连续的缩放
- SDK中增加了更多的示例及技术文档
- Bugs的解决

Dynamic Display不适用情况

- 并不是所有显示需求的灵丹妙药
- 在默认显示状态下不推荐使用
- 打印
- 对显示效果要求较高的情况下慎用
- 在编辑状态下不推荐使用

Dynamic Display 演示

Tracking Analyst

- 需要有Tracking Analyst扩展许可
- 通常是对已经存在的轨迹数据或历史数据变迁的一种回放
- 除了点的轨迹外，也支持线、面要素的动态变化
- 时间字段和事件ID字段是里面比较重要的字段

使用Tracking Analyst开发步骤

一、添加Tracking扩展

```
object mapObj = m_mapControl.Object;

//添加Tracking扩展
IExtensionManager extentionManager = new ExtensionManagerClass();
UID uid = new UIDClass();
uid.Value = "esriTrackingAnalyst.TrackingEngineUtil";
((IExtensionManagerAdmin)extentionManager).AddExtension(uid, ref mapObj);
```

使用Tracking Analyst开发步骤

二、设置Tracking环境

```
//定义Tracking环境
m_trackingEnv = new TrackingEnvironmentClass();
m_trackingEnv.Initialize(ref mapObj);

m_trackingEnv.EnableTemporalDisplayManagement = true;
m_trackingEnv.DefaultTemporalReference.TemporalMode
    = enumTemporalMode.enumHistoric;

//默认时间段单位days
m_trackingEnv.DefaultTemporalReference.TemporalPeriodUnits
    = enumTemporalUnits.enumDays;
m_trackingEnv.DisplayManager.ManualUpdate = true;
```

ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值

使用Tracking Analyst开发步骤

三、添加FeatureClass，创建Tracking图层，设置Renderer和其他ITemporalLayer属性

```
ITemporalRenderer temporalRenderer =
    SetRenderer(pFeatCls, "TA_DATE", "EVENTID");
m_trackingLyr = new TemporalFeatureLayerClass();
((IFeatureLayer)m_trackingLyr).FeatureClass = pFeatCls;
m_trackingLyr.Renderer = temporalRenderer as IFeatureRenderer;
```

- 注：ITemporalLayer必须在Tracking Environment设置好之后创建

使用Tracking Analyst开发步骤

四、随时间变化更新轨迹

- 修改TemporalPerspective属性（时间值）来控制当前播放的位置

```
void m_timer_Elapsed(object sender, System.Timers.ElapsedEventArgs e)
{
    //图层动态变化
    m_trackInvoke.IncrementTimeStamp();

    //播放控制条变化
    this.Invoke(new OnPlayingDelegagte(OnPlaying));
}
```

使用Tracking Analyst开发步骤

- Tracking事件与UI的交互需要用到多线程，在主线程里通过Invoke方法激发自定义事件代理：
- Invoke(new Increment(OnIncrement));

```
//时间增加
m_temporalEnv.DefaultTemporalReference.TemporalPerspective = (object)date;

//刷新
m_activeView.PartialRefresh(esriViewDrawPhase.esriViewGeography,
    m_temporalLayer, null);
```

Tracking Analyst演示

ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值

动态目标追踪的实现方法（一）

- 使用Element绘制，也就是将轨迹动态地以Element的形式添加到地图上
- 优点
  - 可以很方便地定义符号和文字标识；
  - 很多细节都可以控制，如选择、修改等
- 缺点
  - 支持不了太多的数量，一般大几百个Element后速度会慢；
  - 受底图影响大，特别是底图包含大量label；
  - 绘制的Element位于最上层，会压盖底图的label（不过这点可以通过控制label行为来解决）

动态目标追踪的实现方法（二）

- 使用MapControl的DrawShape方法（或IScreenDisplay.Draw**），也是根据轨迹线的shape直接显示
- 优点
  - 屏幕绘制，效率比Element高，刷新快
  - 不会压盖底图标注
- 缺点：
  - 细节控制不方便
  - 刷新后消失，因此需要在OnAfterDraw事件里去重绘才能保持

动态目标追踪的实现方法（三）

- IRealTimeFeedManager等接口实现gps追踪的模拟
- 优点
  - 在内存中构建FeatureLayer，动态更新方便，由于是图层加载，显示较快
- 缺点
  - 细节控制不方便，需要动态构建FeatureClass模拟GPS
- 具体可参考以下例子：\DeveloperKit\SamplesCOM\Applications\GPSNavigatorVisual_Basic

动态目标追踪的实现方法（四）

- 动态图层技术（IDynamicLayer)
- 优点
  - 效率高，能支持数万的的轨迹要素
  - 底图Cache，可重用，不受底图复杂度影响
  - 无缝平滑缩放
- 缺点
  - 细节控制不太方便，选择、查询、移动等都得自己写

结束

- 欢迎大家接着参与下一个讲座

  数据互操作与空间ETL

  ——主讲人：张政



ESRI China (Beijing)
Geographic Thinking
分 享 地 理 价 值