AutoCAD® Map 3D 2013 Platform API Training

Resource Service

Contents

- Resource Service Overview
- Feature Source
- Layer Definition
- Creating Resources
- Managing Resources

Resource Service

Resources

- Files and configuration information required for map and layer display Stored in XML format in the resource repository
- Has a unique path in the repository. Root is “Library://”

  e.g. *Library://Data/Raster/Redding.LayerDefinition,*

  Library://Data/SDF/Zoning.FeatureSource
- MgResourceType specifies resource types
  - Only LayerDefinition and FeatureSource are currently defined in AutoCAD Map
- Getting the Resource Service object (AcMapResourceService)

```
// Get an MgService object from AcMapServiceFactory and cast it to
AcMapResourceService

AcMapResourceService  resServ = (AcMapResourceService )
AcMapServiceFactory.GetService(MgServiceType.ResourceService);
```

Resource Service

Resource Identifier

- A class, MgResourceIdentifier, that encapsulates the string defining the location of a resource
- String ends in “.LayerDefinition” or “.FeatureSource”
- Resource pointed to does not need to exist yet

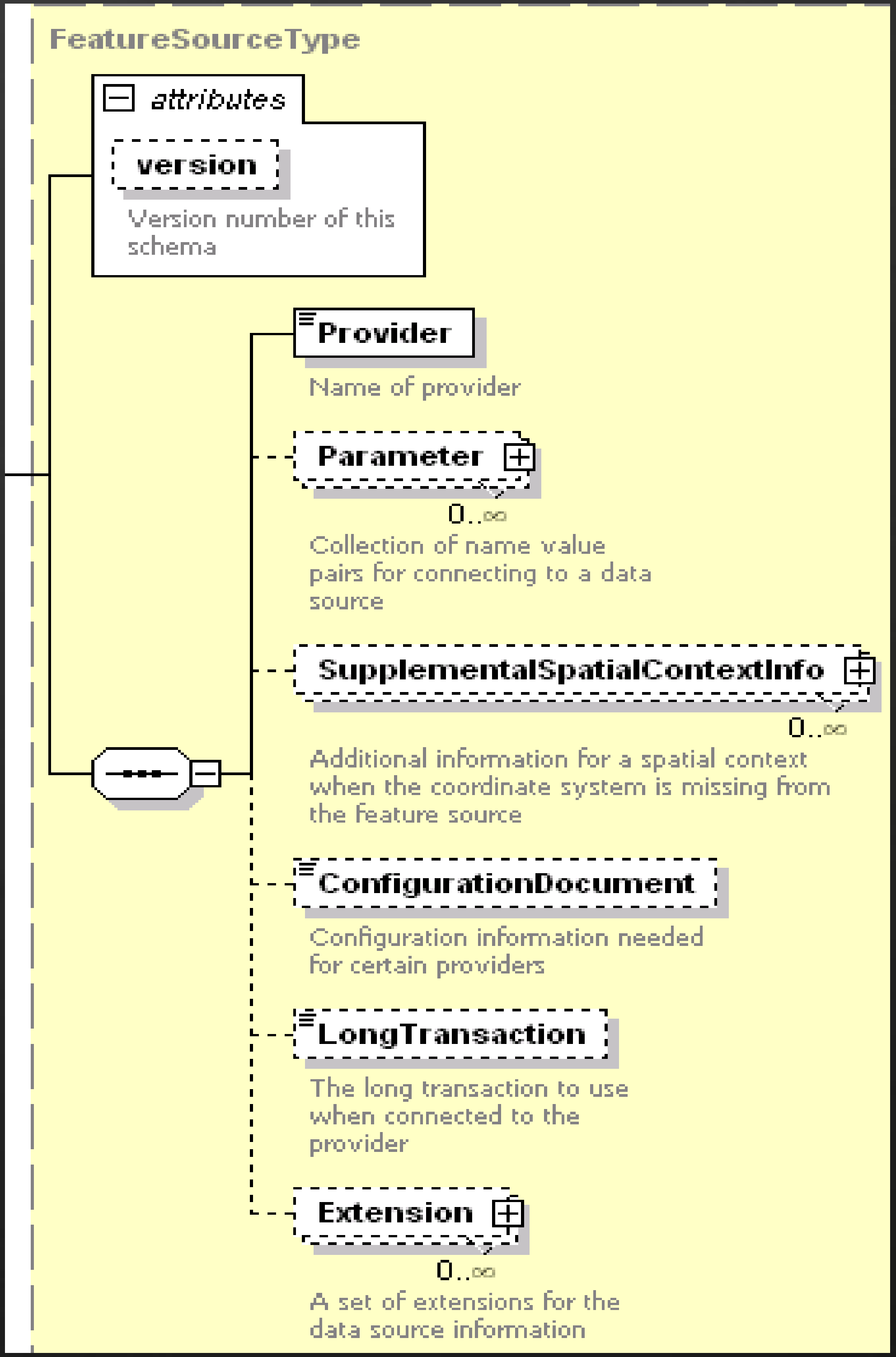
```
// Create a resource Id for a layer definition resource
MgResourceIdentifier LayerDefResId = new
MgResourceIdentifier("Library://Data/Raster/Redding.LayerDefinition


// Create a resource Id for a feature source resource
MgResourceIdentifier FtrSrcResId = new
MgResourceIdentifier("Library://Data/Raster/Redding.FeatureSource
```

Resource Service

FeatureSource

- Describes the information required to connect to a feature source
- See FeatureSource-1.0.0.xsd in “<Map 3D 2013 SDK Folder>\Schema”
- Feature source xml file must be created and added to the repository in order to connect to an FDO provider

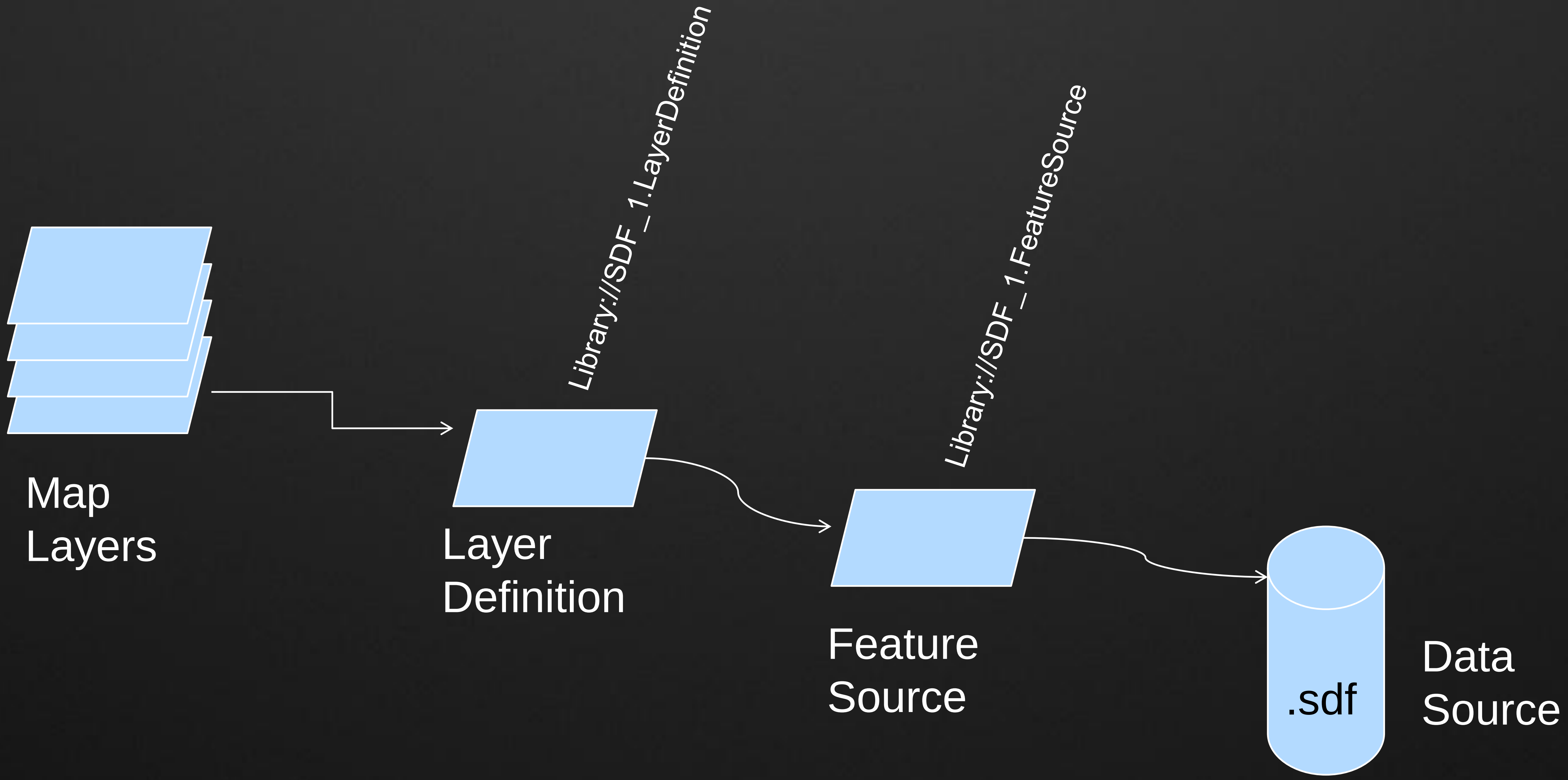


FeatureSourceType
attributes
version
Version number of this schema
Provider
Name of provider
Parameter
0..∞
Collection of name value pairs for connecting to a data source
SupplementalSpatialContextInfo
0..∞
Additional information for a spatial context when the coordinate system is missing from the feature source
ConfigurationDocument
Configuration information needed for certain providers
LongTransaction
The long transaction to use when connected to the provider
Extension
0..∞
A set of extensions for the data source information

Resource Service

FeatureSource

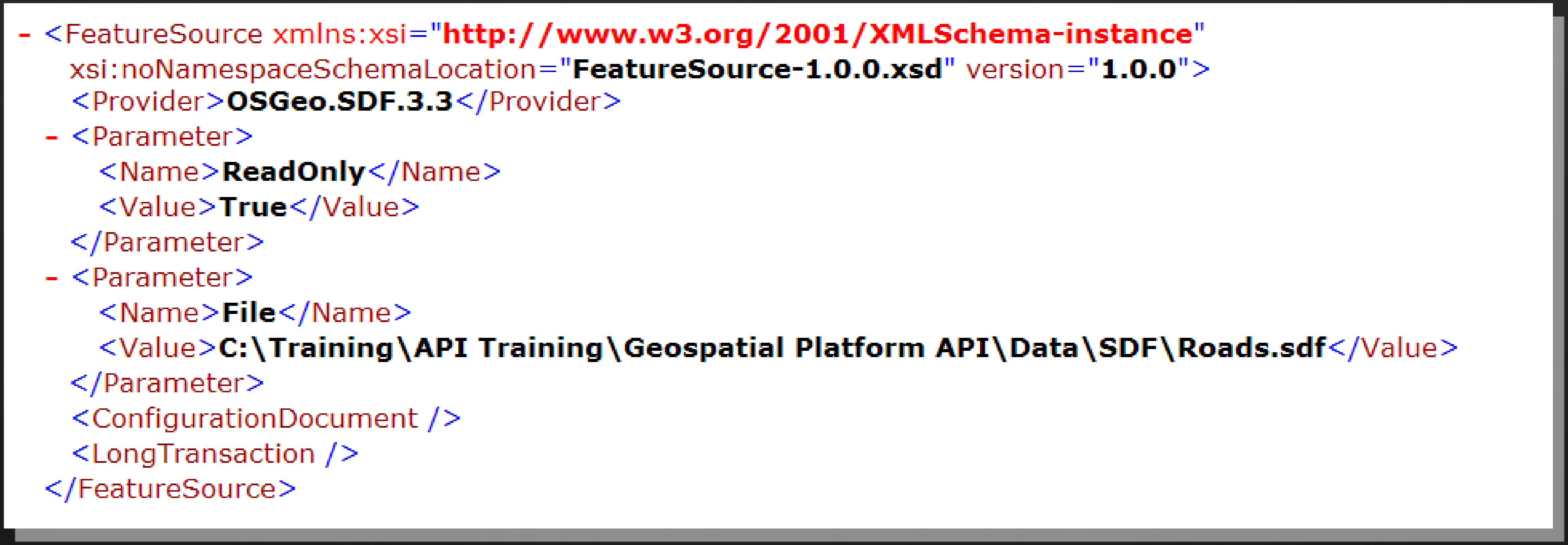
Resource Identifier of the FeatureSource is used in LayerDefinition


Library://SDF_1.LayerDefinition
Library://SDF_1.FeatureSource
Map
Layers
Layer
Definition
Feature
Source
.sdf
Data
Source

Resource Service

FeatureSource Elements

- “Provider” specifies the name of the FDO provider used
- “Parameter” specifies a collection of name/value pairs for connecting to the data source

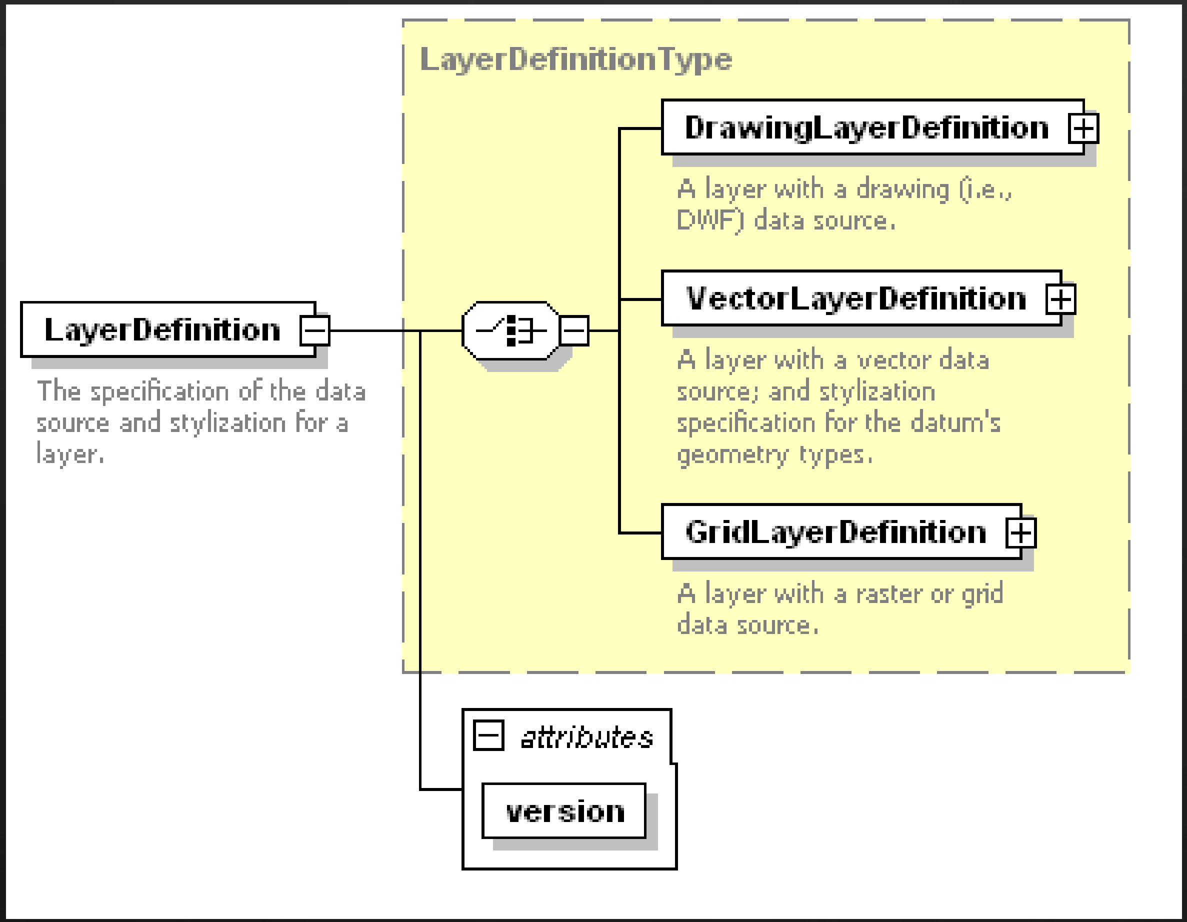


```
- <FeatureSource xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="FeatureSource-1.0.0.xsd" version="1.0.0">
    <Provider>OSGeo.SDF.3.3</Provider>
  - <Parameter>
      <Name>ReadOnly</Name>
      <Value>True</Value>
    </Parameter>
  - <Parameter>
      <Name>File</Name>
      <Value>C:\Training\API Training\Geospatial Platform API\Data\SDF\Roads.sdf</Value>
    </Parameter>
    <ConfigurationDocument />
    <LongTransaction />
  </FeatureSource>
```

Resource Service

LayerDefinition

- Specification of the data source and stylization for a layer
  - See LayerDefinition-1.0.0.xsd in “<Map 3D 2010 SDK Folder>\Schema”
- Required for creating a map layer


LayerDefinitionType
DrawingLayerDefinition
A layer with a drawing (i.e., DWF) data source.
VectorLayerDefinition
A layer with a vector data source; and stylization specification for the datum's geometry types.
GridLayerDefinition
A layer with a raster or grid data source.
LayerDefinition
The specification of the data source and stylization for a layer.
attributes
version

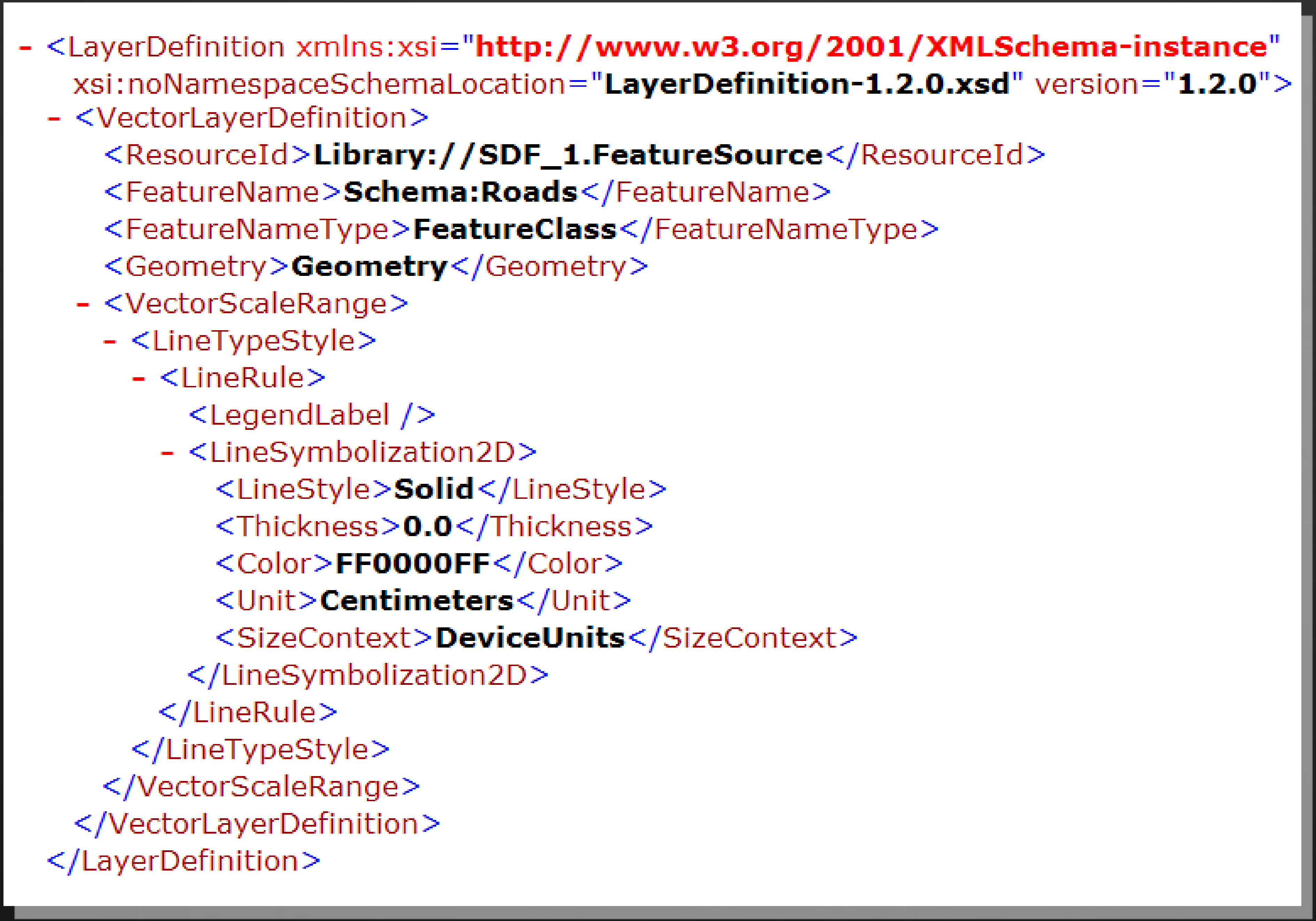
Resource Service

LayerDefinition Elements

- “ResourceId” defines the link to the data source
- “VectorScaleRange”/”GridScaleRange” defines the stylization to be applied to the features for a given scale range
- “Geometry” specifies the geometry property that should be used to get the feature geometries
- “FeatureName” specifies a feature class or named extension

Resource Service

LayerDefinition



```
- <LayerDefinition xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="LayerDefinition-1.2.0.xsd" version="1.2.0">
  - <VectorLayerDefinition>
      <ResourceId>Library://SDF_1.FeatureSource</ResourceId>
      <FeatureName>Schema:Roads</FeatureName>
      <FeatureNameType>FeatureClass</FeatureNameType>
      <Geometry>Geometry</Geometry>
    - <VectorScaleRange>
      - <LineTypeStyle>
        - <LineRule>
            <LegendLabel />
          - <LineSymbolization2D>
              <LineStyle>Solid</LineStyle>
              <Thickness>0.0</Thickness>
              <Color>FF0000FF</Color>
              <Unit>Centimeters</Unit>
              <SizeContext>DeviceUnits</SizeContext>
            </LineSymbolization2D>
          </LineRule>
        </LineTypeStyle>
      </VectorScaleRange>
    </VectorLayerDefinition>
  </LayerDefinition>
```

Resource Service

Creating a feature source in the repository

- Step 1: Generate feature source XML data from existing file

```
// Load XML into memory and modify
XmlDocument doc = new XmlDocument();
doc.Load(C:\XML\SampleFeatureSource.xml")
// Make modifications
XmlNode providerNode = doc.GetElementsByTagName(“Provider");
providerNode.InnerText = “OSGeo.SHP.3.7”;

//Save changes into a memory stream
MemoryStream xmlStream = new MemoryStream();
doc.Save(xmlStream);

//Get Feature source as a byte array
byte [] ftrSrc = xmlStream.ToArray();

Encoding utf8 = Encoding.UTF8;
String StrFtrSrc = new String(utf8.GetChars(ftrSrc));
ftrSrc = new byte[StrFtrSrc .Length-1];
```

Resource Service

Creating a feature source in the repository

- Step 2: Construct MgByteSource object from XML data
- Step 3: Extract data from MgByteSource object and add as (feature source) resource to the repository

```
// Construct byteSource object from data
MgByteSource byteSource = new MgByteSource(ftrSrc, ftrSrc.Length);
byteSource.SetMimeType(MgMimeType.Xml);

// Add to repository
MgResourceIdentifier resId = new
MgResourceIdentifier(“Library://Data/Raster/Redding.FeatureSource“);
AcMapResourceService rs =
AcMapServiceFactory.GetService(MgServiceType.ResourceService)
                    as AcMapResourceService;
rs.SetResource(resId, byteSource .GetReader(), null);
```

Resource Service

Enumerating Resources

- Use MgResourceService::EnumerateResources() to enumerate resources
  - EnumerateResources (MgResourceIdentifier resource, INT32 depth, String type)
  - Method returns MgByteReader, containing byte data. Convert to string for readability
- Type of resource must be specified or empty string used for all types

```
//resources are located in the Library repository
MgResourceIdentifier resourceID = new MgResourceIdentifier("Library://");
//Get LayerDefinitions in the repository
MgByteReader byteRdr = resService.EnumerateResources(resourceID ,0,
MgResourceType.LayerDefinition);
String byteRdrStr = byteRdr.ToString(); //convert to string
```

Resource Service

Getting Resource Content

- Get the XML content of a LayerDefinition or FeatureSource using AcMapResourceService ::GetResourceContent

```
//Get the content of a FeatureSource
MgResourceIdentifier id = new
MgResourceIdentifier("Library://Data/Raster/Redding.FeatureSource");
AcMapResourceService resServ =
(AcMapResourceService)AcMapServiceFactory.GetService(MgServiceType.Res
ourceService);
bool resourceExists = resServ ResourceExists(id);
MgByteReader byteRdr = null;
 if (resourceExists )
 {
        byteRdr = resServ.GetResourceContent(id);
         MgByteSink byteSink = new MgByteSink(byteRdr);
         byteSink.ToFile(@"c:\FeatureResource.Xml");
  }
```

Resource Service

Copying, Moving, Deleting Resources

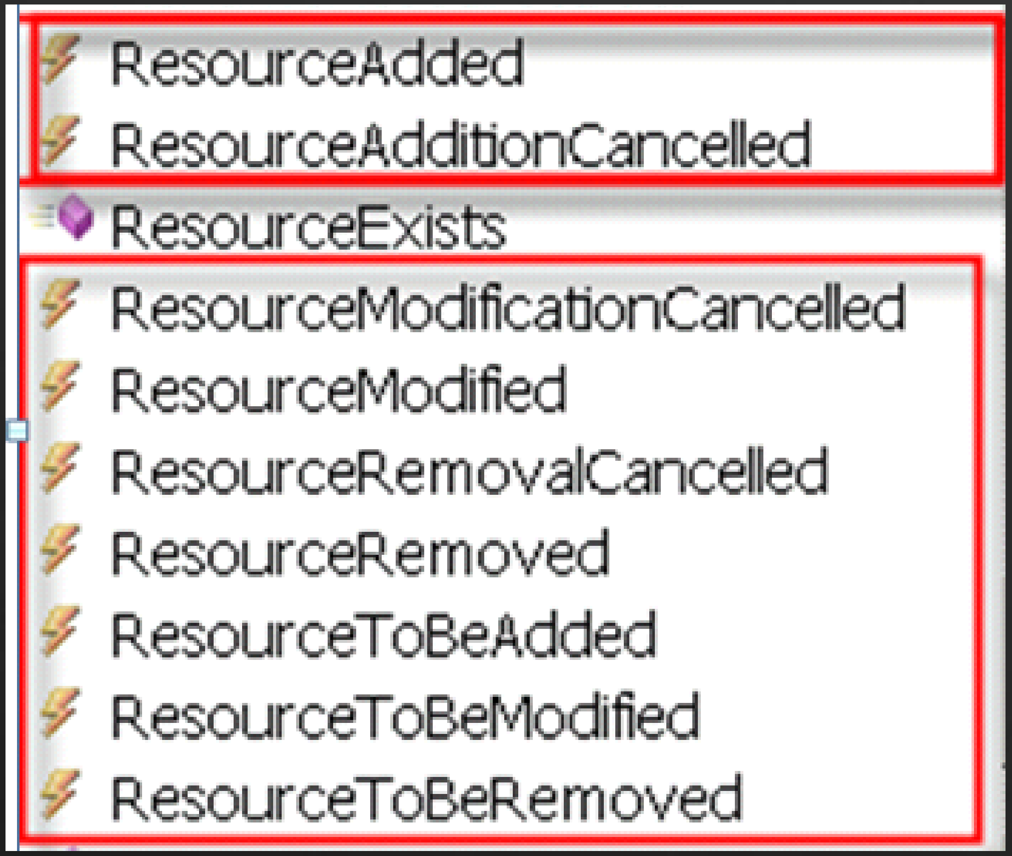
- Copy an existing resource to another location using AcMapResourceService:: CopyResource()
- Move an existing resource to another location with AcMapResourceService ::MoveResource()
- Delete a resource with AcMapResourceService ::DeleteResource()

```
//Copying a LayerDefinition
AcMapResourceService resServ =
(AcMapResourceService)AcMapServiceFactory.GetService(MgServiceType.Res
ourceService);
MgResourceIdentifier id1 = new
MgResourceIdentifier("Library://Data/Raster/Redding.LayerDefinition");
 MgResourceIdentifier id2 = new
MgResourceIdentifier("Library://Data/Raster/Redding2.LayerDefinition");
 if(resServ.ResourceExists(id2)) resServ  CopyResource(id1, id2, true);
```

Autodesk

Resource Service Events

- Occur when resource-related actions, such as resource addition, modification or removal occur



ResourceAdded
ResourceAdditionCancelled
ResourceExists
ResourceModificationCancelled
ResourceModified
ResourceRemovalCancelled
ResourceRemoved
ResourceToBeAdded
ResourceToBeModified
ResourceToBeRemoved

- Event functions are members of AcMapResourceService

ResourceService Events

```
//Handling the ResourceAdded event , declare delegate object
private static ResourceAddedHandler g_eventHandler = null;
public bool RegisterEvent()
{
   AcMapResourceService rs = AcMapServiceFactory.GetService(MgServiceType.ResourceService)
   as AcMapResourceService;

  //Create an instance of the delegate
   g_eventHandler = new ResourceAddedHandler(this.MyEventHandlerFunction

  //Associate delegate with the event
   rs.ResourceAdded += g_eventHandler
}

//Implement handler function
private void MyEventHandlerFunction (object sender, AcMapResourceEventArgs args
{
               MgResourceIdentifier resId = args.GetResourceIdentifier();
               MessageBox.Show(resId.ToString() + “ added”);
}
```

Autodesk®