Chapter 3: Working with Site Service and Resource Service

Connecting to site and accessing resources

Autodesk

Chapter Overview

- Site service
  - Web configuration file
  - User authentication and site connection
  - User sessions
  - Resource identifier

- Resource service
  - Resource service creation
  - Resource enumeration
  - Understand a resource in a repository
  - Copy, move, rename, and delete a resource
  - Get and set a resource header, content and data

Web Configuration File – webconfig.ini

- A text file used to perform basic initialization when a session starts
- Read by MgInitializeWebTier()
- Contains parameters required to connect to the site server:
  - IP address
  - port numbers
  - Map agent requests customization, e.g. pause time between successive requests
  - Passwords for OGC (WMS & WFS) requests
- Values set in file affect server performance. In most cases, no need to change.

Initializing the Web Tier

- Must be done before using the API
- The configuration file (webconfig.ini) is used to initialize the web tier

```
MapGuideApi.MgInitializeWebTier("webconfig.ini")
```

User Authentication and Site Connection

- User authentication is required for connection to site
- Instantiate MgUserInformation object with the user ID and password
- User ID and password can be hard-coded to allow automatic connection

```
//Instantiate user information object
MgUserInformation userInfo = new MgUserInformation("UserID", "Password");

//Instantiate a site connection object and
//establish first connection
MgSiteConnection siteConn = new MgSiteConnection();
siteConn.Open(userInfo);
```

User Sessions

- User session is a course of time during which a certain user interacts with the application.
- Created with CreateSession() method.
- Session id is used to identify a user's authentication and temporary resources on MGE until the session expires.
- Session id string is maintained in the web session state object by map viewer automatically.

```
//Create a session repository, session identifier returned
MgSite site = siteConn.GetSite();
String sessionId = site.CreateSession();
```

```
//Pass the session id to the map viewer
<frame src="/mapserver201x/mapviewernet/ajaxviewer.aspx?SESSION=<%=
sessionId %>&WEBLAYOUT=<%= webLayout %>" name="ViewerFrame" />
```

User Sessions

- Session identifier can be propagated from one web page to another using standard web tier session mechanisms like client side cookies, hidden form fields, and additional URL parameters and used for reconnection to site

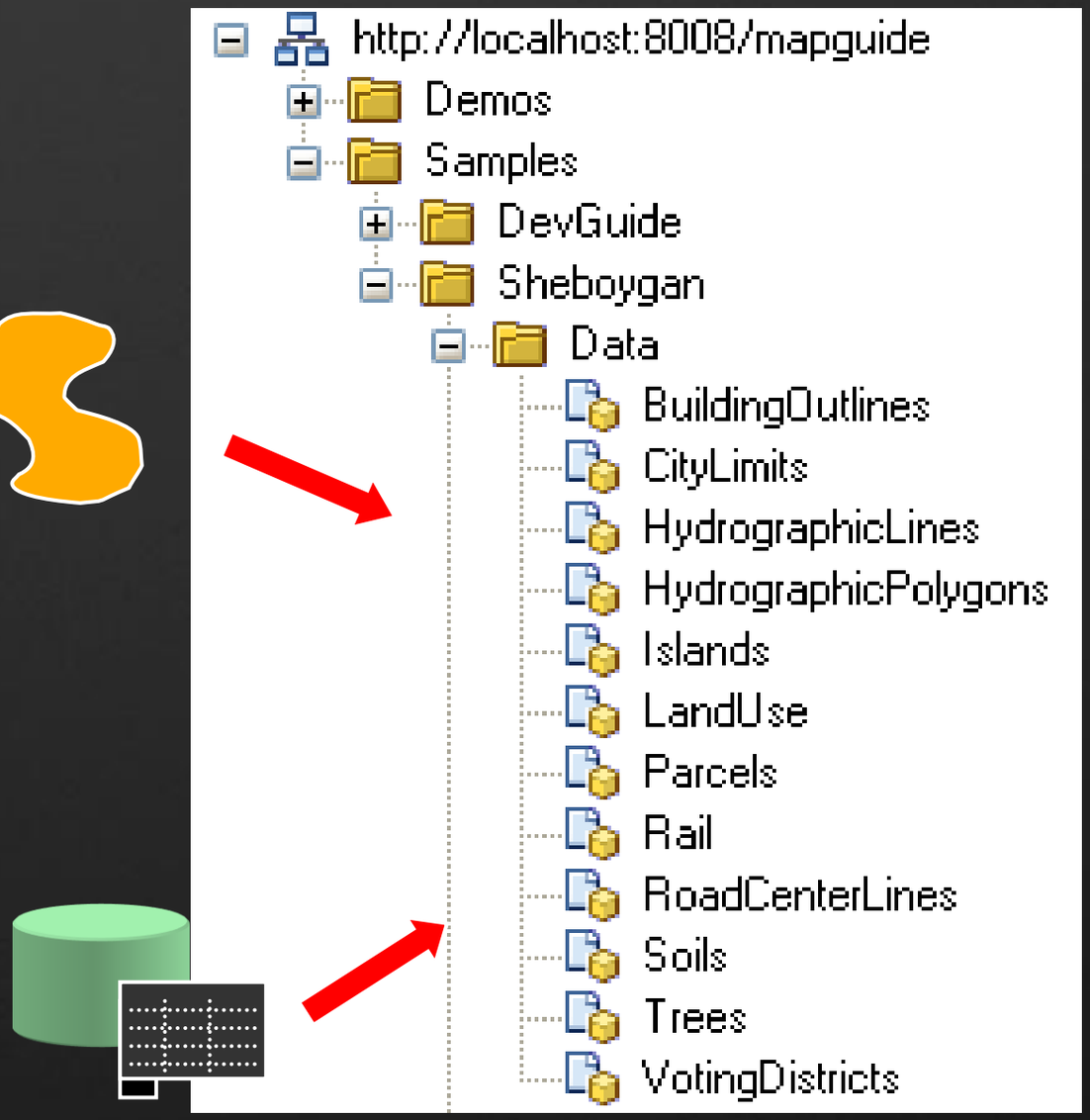
- Session identifier is a string type, e.g. ce6f7738-ffff-ffff-8001-005056c00008_en

```
//Pass the session id to the map viewer
String sessionId = Request.Form.Get("SESSION");
```

```
//Stored session identifier is used for subsequent reconnection
MgUserInformation userInfo = new MgUserInformation(sessionId);
MgSiteConnection siteConn = new MgSiteConnection();
siteConn.Open(userInfo);
```

Resources and Repositories

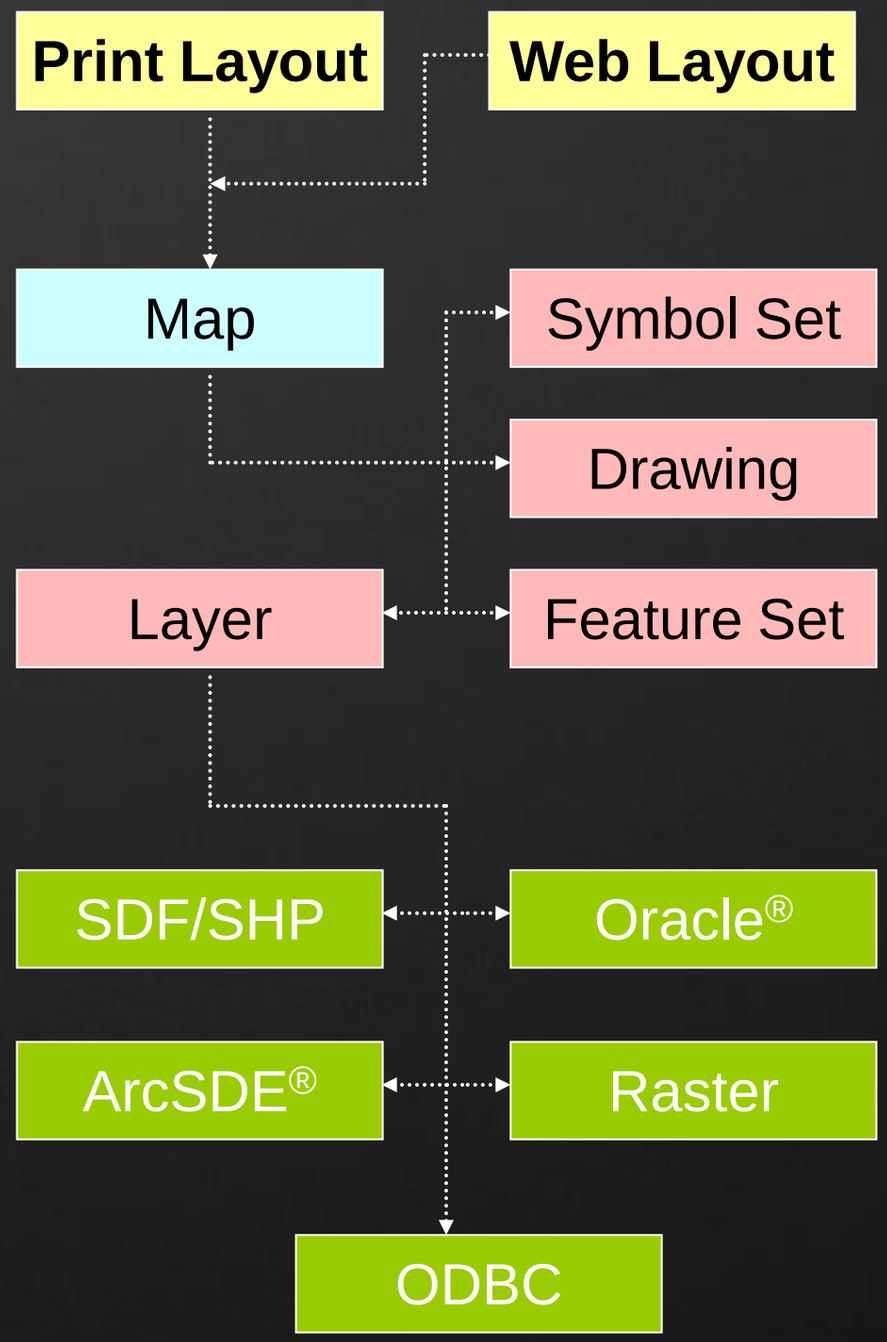
- A repository is an XML database that stores and manages site data.
- Repository types
  - Library repository
  - Session repository
- Repository resides on the site server
  - **File-based** resources can be copied and physically stored, such as SDF, SHP, DWF, and raster images, etc. or they can be referenced
  - **Database resources** are referenced with credentials, such as Oracle Spatial®, ArcSDE®, and ODBC, etc.



Autodesk

Understand Resources

- Resource in MapGuide is in a hierarchical relationship
- Feature resource is composed of 3 parts:
  - **Resource Contents**
    - Meta data in XML format
    - For layers, it is the layer symbol and resource ID of the feature data.
    - For maps, it is the spatial extent, resource ID of the layers, and their selectability and visibility.
  - **Resource Header**
    - Security restraints in XML format
    - Who can access what
  - **Resource Data**
    - Actual resource data. It can be a file, stream, or string.



Autodesk

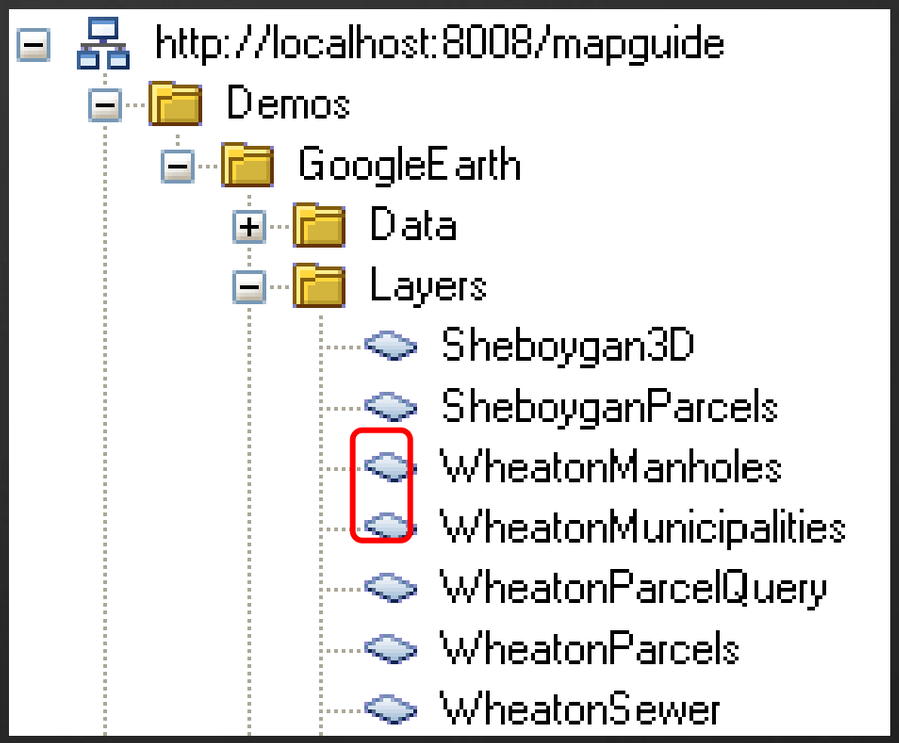
Resource Service

- Resource service enables you to manipulate the site repository and resources
- Resource identifier is used to reference a resource
- Two types of repository

- **Library**
  - Persistent resource storage
  - Resource identifier correspondent to Site Explorer in MapGuide Studio

- **Session Repository**
  - Temporary resource storage
  - Valid only for the duration of session
  - Destroyed after session expiration
  - Resource identifier correspondent to session ID

Library:



Library://Demos/GoogleEarth/Layers/SheboyganParcels.LayerDefinition

Session Repository:

Session:a421c694-ffff-ffff-8000-005056c00008_eng//tempLayer.LayerDefinition

Autodesk

Resource Identifiers

- Each resource has a string type ID, prefixed by its repository type.
- For example
  - Library repository resource identifier for a web layout:
    Library://Samples/Sheboygan/Layouts/SheboyganPhp.WebLayout

  - Session repository resource identifier for a layer definition:
    Session:70ea89fe-0000-1000-8000-005056c00008_en//layer.LayerDefinition

Creating Resource Services

- A resource service manages data in a repository
- To use a service, web page must open a site connection and create an instance of a resource service
- MgSiteConnection object is used in creating the resource service

```
MgUserInformation userInfo = new MgUserInformation(sessionID);
siteConnection = new MgSiteConnection();
siteConnection.Open(userInfo);
MgResourceService resourceService =
    (MgResourceService)siteConnection.CreateService(
    MgServiceType.ResourceService);
```

Enumerating Repository Resources

- Use MgResourceService::EnumerateResources() to enumerate resources
- A resource identifier (MgResourceIdentifier) specifying the location of the resources to be enumerated is required for enumeration
- Type of resource must be specified or use empty string for all types

```
MgResourceService resService =
    (MgResourceService)siteConn.CreateService(
    MgServiceType.ResourceService
//resources are located in a Library repository
MgResourceIdentifier resourceID = new
    MgResourceIdentifier("Library://");
//return info on MapDefs in folder and all its descendants
MgByteReader byteRdr = resService.EnumerateResources(
    resourceID, -1, "MapDefinition")
//convert to string
String byteRdrStr = byteRdr.ToString();
```

Enumerating Repository Resources

- Create an MgByteSink object from the returned MgByteReader and save it to a file.

```
MgResourceIdentifier resourceID = new
    MgResourceIdentifier("Library://Samples/Sheboygan/Data/");
//return info on “Data” folder only
MgByteReader byteRdr = resService.EnumerateResources(
    resourceID, 0, “Folder");

//Create a byte sink object with the byte reader
MgByteSink byteSink = new MgByteSink(byteRdr);

// write to file
byteSink.ToFile(“MapDef.xml”);
```

Resource Contents

- Resource contents can be obtained using GetResourceContent()
- MapDefinition, LayerDefinition, PrintLayout, SymbolLibrary, LoadProcedure have associated XML Schemas
  - See C:\Program Files\Autodesk\MapGuideEnterprise2010\Server\Schema

```
//A map definition resource
MgResourceIdentifier resourceID = new MgResourceIdentifier(
    "Library://Samples/Sheboygan/Maps/Sheboygan.MapDefinition");

//return resource content
MgByteReader byteRdr = resService.GetResourceContent(resourceID);

//Create a byte sink object with the byte reader
MgByteSink byteSink = new MgByteSink(byteRdr);

// write to file
byteSink.ToFile("MapDef.xml");
```

Resource Contents

```
<?xml version="1.0" encoding="UTF-8" ?>
- <MapDefinition xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://
    1.0.0.xsd">
    <Name>Sheboygan Map</Name>
    <CoordinateSystem>GEOGCS["WGS84 Lat/Long's, Degrees, -180 ==> +180",DATUM["D_WGS
      ["World_Geodetic_System_of_1984",6378137,298.257222932867]],PRIMEM["Greenwich"
  - <Extents>
      <MinX>-87.764986990962839</MinX>
      <MaxX>-87.695521510899724</MaxX>
      <MinY>43.6913981287877 82</MinY>
      <MaxY>43.797520000480347</MaxY>
    </Extents>
    <BackgroundColor>FFCDBD9C</BackgroundColor>
  - <MapLayer>
      <Name>Roads</Name>
      <ResourceId>Library://Samples/Sheboygan/Layers/Roads.LayerDefinition</ResourceId>
      <Selectable>false</Selectable>
      <ShowInLegend>true</ShowInLegend>
      <LegendLabel>Roads</LegendLabel>
      <ExpandInLegend>true</ExpandInLegend>
      <Visible>true</Visible>
      <Group>Transportation</Group>
    </MapLayer>
  - <MapLayer>
      <Name>Rail Lines</Name>
      <ResourceId>Library://Samples/Sheboygan/Layers/Tracks.LayerDefinition</ResourceId>
      <Selectable>false</Selectable>
      <ShowInLegend>true</ShowInLegend>
      <LegendLabel>Rail Lines</LegendLabel>
      <ExpandInLegend>true</ExpandInLegend>
      <Visible>false</Visible>
      <Group>Transportation</Group>
    </MapLayer>
```

Working with Resources

- Get the header associated with the specified resource using GetResourceHeader()
- GetResourceHeader() returns MgByteReader containing the header information

```
//return info on resource header
MgByteReader byteRdr = resService.GetResourceHeader(resourceID);

//Create a byte sink object with the byte reader
MgByteSink byteSink = new MgByteSink(byteRdr );

// write to file
byteSink.ToFile("resHdr.xml")
```

Resource headers

```
<?xml version="1.0" encoding="utf-8" ?>
- <ResourceDocumentHeader xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xsi:noNamespaceSchemaLocation="ResourceDocumentHeader-1.0.0.xsd">
  - <General>
    <IconName>LayerDefinitionVector</IconName>
  </General>
  - <Security xsi:noNamespaceSchemaLocation="ResourceSecurity-1.0.0.xsd">
    <Inherited>true</Inherited>
  </Security>
  - <Metadata>
    - <Simple>
      - <Property xsi:noNamespaceSchemaLocation="Property-1.0.0.xsd">
        <Name>_Queryable</Name>
        <Value>0</Value>
      </Property>
      - <Property xsi:noNamespaceSchemaLocation="Property-1.0.0.xsd">
        <Name>_Bounds</Name>
        <Value><Bounds SRS="EPSG:4326" west="-87.74" south="43.68"
          east="-87.69" north="43.815"/></Value>
      </Property>
      - <Property xsi:noNamespaceSchemaLocation="Property-1.0.0.xsd">
        <Name>_Title</Name>
        <Value>City Limits</Value>
      </Property>
      - <Property xsi:noNamespaceSchemaLocation="Property-1.0.0.xsd">
        <Name>_Opaque</Name>
        <Value>0</Value>
      </Property>
      - <Property xsi:noNamespaceSchemaLocation="Property-1.0.0.xsd">
        <Name>_IsPublished</Name>
        <Value>1</Value>
      </Property>
    </Simple>
  </Metadata>
</ResourceDocumentHeader>
```

Working with Resources

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

```
//Rename UKMapI.MapDefinition to UKMapII.MapDefinition
MgResourceIdentifier oldID = new MgResourceIdentifier(
    "Library://Samples/UK/Maps/UKMapI.MapDefinition");

MgResourceIdentifier newID = new MgResourceIdentifier(
    "Library://Samples/UK/Maps/UKMapII.MapDefinition");

resService.MoveResource(oldID, newID, true);
```

Resource Data

- Data used by a resource, but not stored in the resource itself, e.g. a binary SDF file, stored separately from the XML used to store the SDF resource itself
- Resource data can be stored as *files*, *streams*, or *strings*
  - Files are used when the data is large, e.g. SDF files
  - Streams are used for faster access for smaller pieces of binary data, e.g. symbols
  - Strings are used for small pieces of text data, e.g. database access credentials

```
//Create the resource ID
MgResourceIdentifier resourceID = new
MgResourceIdentifier("Library://Samples/UK/Data/UKBase.FeatureSource")
;
//Data name as specified in XXX.FeatureSource
String name = "UKBase.sdf";
// Get the layer SDF data and write it out to file
MgByteReader byteRdr = resService.GetResourceData(resourceID,name);
MgByteSink byteSink = new MgByteSink(byteRdr);
byteSink.ToFile(SDFPath);
```

Questions

Questions ?

Autodesk

Demo & Exercise

- http://localhost/mapserver/mapagent/index.html
- Maestro

- resource service API

- Check the XML presentation