May 2012
Oracle Spatial User Conference
May 23, 2012
Ronald Reagan Building and International Trade Center
Washington, DC  USA
Liujian Qian
Director, Product Development
Building Applications with Oracle MapViewer
Program Agenda

• The Result
• MapViewer Overview
• The Steps
  – Prepare spatial content
  – Author maps with MapBuilder
  – Configure MapViewer server
  – Create sample application with MapBuilder
• Next
  – Review Oracle Maps Tutorials
The Goal
Oracle MapViewer

• Standards-based J2EE and Java Server Faces component
  – XML/HTTP, Java/AJAX
• Publish spatial data to the web
• Map and feature cache provides smooth scroll (pan, zoom)
• Rich Java, XML, JavaScript APIs provide client side interactivity
• Centrally managed map definitions, symbology, and styling rules

A standard feature of all Oracle Fusion Middleware editions
Oracle Mapviewer: architecture

Oracle Database

Mapbuilder
Define map metadata styles, themes, maps

WebLogic or J2EE Container

MapViewer server components

Oracle Maps server-side

Dynamic SQL query
map cache server
rendering engine

HTTP
(get map tiles)

Mapviewer client side

external map providers
eLocation, Nokia, Bing

HTTP: (Map interaction client lib)
(map tiles and feature data)
MapViewer Architecture

Application

JavaScript Map API

HTTP

Map Cache Server

FOI Server

Map Rendering Engine

JDBC

Spatial Tables

Map Definitions

Render spatial data stored in the database

Store and use rendered map tiles
The Steps
But we’ll go through them backwards (sort of)

- Import and/or prepare spatial data
- Create map metadata (styles, themes, maps, tile cache) with MapBuilder
- Set up MapViewer server (in WLS or some app server)
- Build starter html app with MapBuilder app wizard
Use the map builder app wizard
Map Builder has a simple app wizard

- Creates a simple Oracle Maps web app
- Uses the JavaScript slippy mapping API
Application Builder - Step 1 of 5 - Web Page Basic Parameters

Web Page Basic Parameters

- **Tile Layer**: WORLD_MAP
- **Page Title**: Sample Oracle Maps Application
- **Page Header**: Oracle Maps Sample Application
- **Map Height (pixels)**: 600

[Buttons: Help, <Back, Next>, Finish, Cancel]
Additional Application Parameters

MapViewer Server: http://maps.us.oracle.com/mapviewer
MapViewer Data Source: nedata
Center X: 10
Center Y: 20
SRID: 8307
Zoom Level: 1
HTML File: sharma.ORADE\Documents\oramap_sample.html
Snippets of generated code

```javascript
<TITLE> Sample Oracle Maps Application </TITLE>
<!– get the javascript lib -->
<script language="Javascript" src="http://maps.us.oracle.com/mapviewer/fsmc/jslib/oraclemaps.js"></script>

function showMap()
{
    var baseURL  = "http://maps.us.oracle.com/mapviewer";
    var mapCenterLon = 10.0;
    var mapCenterLat = 40.0;
    var mapZoom = 1;
    var mpoint = MVSdoGeometry.createPoint(mapCenterLon,mapCenterLat,8307);
    mapView = new MVMapView(document.getElementById("map"), baseURL);
    mapView.addMapTileLayer(new MVMapTileLayer("nedata.WORLD_MAP"));
    mapView.setCenter(mpoint);
    mapView.setZoomLevel(mapZoom);
    mapView.addNavigationPanel('EAST');
    addThemeBasedFOI();
    md1 = new MVMapDecoration("Made with Natural Earth",0.8, 0.95,null,null);
    mapView.addMapDecoration(md1);
    addToolBar();

    mapView.display();
}
Viewed in a browser
BUT HOW DO WE GET HERE FROM THERE …
The elements of a MapViewer app

- The javascript library
  - src="http://maps.us.oracle.com/mapviewer/fsmc/jslib/oraclemaps.js"

- The background map (tile layer)
  - MVMapTileLayer("nedata.WORLD_MAP")

- The mapviewer server and its data sources
  - baseURL = "http://maps.us.oracle.com/mapviewer" and "nedata"

- Interactive or dynamic elements (features-of-interest)
  - function addThemeBasedFOI() {
    var themebasedfoi1 = new MVThemeBasedFOI('themebasedfoi1','nedata.CHILD_MORTALITY');
  }

- Other stuff (the html div, events, etc.)
or …
Or maybe, since he’s already written the book on this … let’s use his workshop material instead.

And ignore minor details like the fact that the examples differ from previous screenshots ….
Oracle Maps is composed of client and server side components.

Client side:
- Comprised of a JavaScript viewing library (called FSMC - Free Scrolling Map Client) and a public API
- Resides inside the browser
- Assembles various contents into a map displayed to the end user
- Talks to server side components through AJAX calls

Server side:
- Map Cache server which manages pre-generated base map tiles
- FOI server which creates interactive FOI data for the client
- Both are relying upon the main MapViewer rendering engine
Oracle Maps: map cache server

Map Cache Server

- Map Cache Instance 1
  - Map source 1
- Map Cache Instance 2
  - Map source 2
- Map Cache Instance 3
  - Map source 3

map tiles
Map cache server: tiling

- Each map tile in a map cache instance is identified by a mesh code.

- Defined as a pair of integers \((Mx, My)\)
  - \(Mx\) specifies the X dimension index of the tile
  - \(My\) specifies the Y dimension index of the tile.

- Transparent to the application developer
  - Oracle Maps client viewing lib automatically calculates mesh codes.
Oracle Maps FOI server

• Any existing geometry theme can be added to the map as a FOI layer
• The FOI server requires zero-configuration; it relies on MapViewer to process a theme, then repackages the theme data into a format that can be interpreted by the client viewing library
• When using MapBuilder to create a predefined theme, you can specify “info columns” in the theme to be displayed in its FOI info-tip windows
Create map metadata in Map Builder
Map metadata

- Styles: color, line, marker, area, text, advanced
- Themes: associating styles with base tables
- Base map: a set of themes grouped together
- Tile layer: tiled maps
Updating Map Definitions

- Definitions are stored in the database
  - USER_SDO_STYLES
  - USER_SDO_THEMES
  - USER_SDO_MAPS
  - USER_SDO_CACHED_MAPS
- Updated using MapBuilder
Steps for Defining Maps

• First, define the styles you will need
  – Use meaningful names: “C.MAP_COUNTRIES” for the color of countries

• Then define the themes using the styles you defined

• Finally, define the map from the themes

• Can modify styles, themes and maps at any time
• Style information stored in the database
  – Text, colors, line styles, area and fill information, markers
  – Advanced styles for thematic mapping based on a column value
• Includes XML definition of each style
• Can add styles easily using Mapbuilder

```
SQL> describe user_sdo_styles
Name                      Type ----------------------  ------------------
-------------------------- --------------------------
NAME                      VARCHAR2(32)              
TYPE                      VARCHAR2(32)              
DESCRIPTION               VARCHAR2(4000)             
DEFINITION                CLOB                      
IMAGE                     BLOB                      
GEOMETRY                  MDSYS.SDO_GEOMETRY         
```
Pre-defined styles

• Set of styles provided with Mapviewer

• Load sample styles:
  – Script “defaultstyles.sql” provided in $MAPVIEWER_HOME/WEB-INF/admin
  – Run as your application user
  – Will populate USER_SDO_STYLES

```
SQL> connect scott/******
SQL> @defaultstyles.sql
```
USER_SDO_THEMES

- Stores user-defined themes
- User specifies:
  - Theme name and description
  - Table name
  - Geometry column name
  - Styling rules in XML based on values in USER_SDO_STYLES

```sql
SQL> describe user_sdo_themes
Name          Type
------------- ---------------
NAME          VARCHAR2(32)
DESCRIPTION   VARCHAR2(4000)
BASE_TABLE    VARCHAR2(32)
GEOMETRY_COLUMN VARCHAR2(2048)
STYLING_RULES CLOB
```
A map is a collection of themes
User specifies a map name and (optionally) a description
User specifies an XML definition:
  - Theme names to display
  - Zoom scale information to specify when themes display

```
SQL> describe user_sdo_maps
Name               Type
------------------- -------------------
NAME                VARCHAR2(32)
DESCRIPTION         VARCHAR2(4000)
DEFINITION          CLOB
```
Using MapBuilder

- Stand-alone java tool
- Provided with MapViewer
- Completely autonomous
  - No need to specify any CLASSPATH
- Single self-contained jar file
  - mapbuilder.jar
- Needs a JRE 1.5 or later
- Used to design maps (colors, scales, etc)
- Also used to view data
Start and Connect

Start MapBuilder

- Run `mapbuilder.jar`
  
  ```java
  java -jar mapbuilder.jar
  ```

- On Windows, double-click “mapbuilder.jar” (if your java environment is set up to allow this)

Setup a database connection

Startup Parameters

• Specify on the command line

```java
java -Xms128m -Xmx256m \
    -Duser.language=EN -Duser.region=US \
    -Dcom.sun.media.jai.disableMediaLib=true \
    -jar mapbuilder.jar
```

• Some useful parameters
  – `-Xms128m` and `-Xmx256m` to set initial and maximal heap space.
  – `-Dcom.sun.media.jai.disableMediaLib=true`. Use it to get rid of the following error messages

```
Error: Could not find mediaLib accelerator wrapper classes. Continuing in pure Java mode.
Occurs in: com.sun.media.jai.mlib.MediaLibAccessor
java.lang.NoClassDefFoundError: com/sun/media/jai/mlib/Image
    at com.sun.media.jai.mlib.MediaLibAccessor$1.run(MediaLibAccessor.java:248)
```
Styles
<?xml version="1.0" standalone="yes"?>
<svg height="1in" width="1in">
  <rect fill="#FFFFCC" stroke="#BB99BB" width="50" height="50"/>
</svg>
Hello World!
Thematics: Advanced Styles

- **Bucket**
  - Specify variable or fixed (equal-range) buckets
  - Separate color for each bucket

- **Color Scheme**
  - Variable or fixed buckets
  - Buckets assigned gradually darker colors from a base color

- **Dot Density**
  - Area filled with dots proportionally to the value of the variable

- **Variable Marker**
  - Size of chosen symbol increases proportionally to the value of the variable
Thematics: Advanced Styles

- Bar Chart
  - Combines multiple variables

- Pie Chart
  - Same

- Variable Pie Chart
  - Same, but size of pie chart increases proportionally to the sum of the variables

- Heat Map
  - Color ranges from “cool” (blue) to “hot” (red) according to density of geometries
Define a Variable Bucket style

- First define the base colors to use
  - Fill the table of min and max values for each bucket
  - Choose the base color for each bucket
  - Add a descriptive text (used in the legend)
Themes
What Are Themes?

- A theme applies one or more styles to a geometry column
  - Feature drawing information (color, fill, marker, and so on)
  - Text and label font, color, size
- Thematic mapping is supported by associating an advanced style to a theme.
- Themes can be based on any kind of spatial data:
  - Vector (SDO_GEOMETRY)
  - Raster (SDO_GEORASTER)
  - Network, topology,…
  - WMS or WFS servers
Base Maps
What Are Maps?

• A map groups one or more themes
• Can specify “min scale” and/or “max scale” values to control the visibility of each theme depending on map scale.
• Can also control label visibility
• Use ratio-scales (not “mapviewer native”)
• Can control the order in which themes are rendered
• A theme can be used in many different maps
Create a Map

- A map is a collection of themes
- A map will be used as a background showing BI results
- Select the themes to appear on the map
- Ordering of the themes is important
  - Rendered in the order they are listed
- Set theme visibility
  - Scales at which each theme is visible
  - Details only appear when zoomed in
Updating Definitions

- Changes are only applied to the database tables
  - USER_SDO_STYLES, _THEMES and _MAPS
- Existing applications will not see the changes
  - Definitions are cached in memory
  - Clear the cache:

Map Tile Caches are also invalid now!
  - Rebuild the cache!
Exporting/Importing the definitions

- Not automatically exported!
- Cannot export from views!
- First, save the definitions into regular tables

```
CREATE TABLE SAVED_MAPS     AS SELECT * FROM USER_SDO_MAPS;
CREATE TABLE SAVED_THEMES    AS SELECT * FROM USER_SDO_THEMES;
CREATE TABLE SAVED_STYLES    AS SELECT * FROM USER_SDO_STYLES;
CREATE TABLE SAVED_CACHED_MAPS  AS SELECT * FROM USER_SDO_CACHED_MAPS;
```

- Then export those tables and import them in the target database
- Now restore the definitions in the target database

```
INSERT INTO USER_SDO_MAPS     SELECT * FROM SAVED_MAPS;
INSERT INTO USER_SDO_THEMES    SELECT * FROM SAVED_THEMES;
INSERT INTO USER_SDO_STYLES    SELECT * FROM SAVED_STYLES;
INSERT INTO USER_SDO_CACHED_MAPS  SELECT * FROM SAVED_CACHED_MAPS;
COMMIT;
```

- Finally: restart Mapviewer
Exporting/Importing the definitions

- Can also export using MapBuilder
- “Tools” menu, then “Export Metadata”
- Creates a text file

To import: “Tools” menu then “Import Metadata”

If element already exists then the import fails!
Map Tile Layers (Map Caches)
How to Define a Map Tile Layer

Several methods are available
- MapViewer administration console
- MapBuilder
- SQL
- MapViewer XML administration interface
1. Using MapViewer Administration Console
Define a Map Tile Layer
Define a Map Tile Layer

- An “internal” cache is based on a locally defined base map
- An “external” cache is based on some external map provider via “adapters”
  - Web Map Server
  - Mapviewer service
  - Any other: write your own adapter
Define a Map Tile Layer

- **Name of the cache**
- **Base map for the cache**
- **Zoom levels and scales**
- **Area covered by the cache**
- **Tile size and format**
2. Using MapBuilder

- Use the Tile Layer definition wizard
- Select base map
- Select area from the map
- Select min and max scales from the map
- Generate scales for zoom levels
- Choose tile size
- Try it out
Manage Map Tile Layers

Existing map tile layers

Select a map tile layer and

<table>
<thead>
<tr>
<th>Select Name</th>
<th>Data Source</th>
<th>Base map</th>
<th>Zoom levels</th>
<th>Internal</th>
<th>Tile width</th>
<th>Tile height Online</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTOMER_MAP</td>
<td>MVDEMO</td>
<td>CUSTOMER_MAP</td>
<td>10</td>
<td>true</td>
<td>268</td>
<td>268 true</td>
</tr>
<tr>
<td>DEMO_MAP</td>
<td>MVDEMO</td>
<td>DEMO_MAP</td>
<td>10</td>
<td>true</td>
<td>268</td>
<td>268 true</td>
</tr>
<tr>
<td>US_BASE_MAP</td>
<td>SCOTT</td>
<td>US_BASE_MAP</td>
<td>10</td>
<td>true</td>
<td>268</td>
<td>268 true</td>
</tr>
</tbody>
</table>
Manage Map Tile Layers

![MapViewer](image)

**Editing Map Tile Layers**

- **Name:** US_BASE_MAP
- **Data source:** SCOTT
- **Max browser tile cache age (hours):** 180

**Basic settings**

- **Base map:** US_BASE_MAP
- **Background:** #ABCAF0, transparent
- **Anti-aliased:** 
- **Tile storage:** D:tilecache\SCOTT\US_BASE_MAP

- **Tile width (pixels):** 256
- **Tile height (pixels):** 256
- **Tile format:** PNG

**Coordinate System Definition**

**Zoom Level Definition**
Manage Map Tiles

Managing Map Tile Layers

Existing map tile layers

Select a map tile layer and

<table>
<thead>
<tr>
<th>Select Name</th>
<th>Data Source</th>
<th>Data Map</th>
<th>Zoom Levels</th>
<th>Internal</th>
<th>Tile Width</th>
<th>Tile Height Online</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTOMER_MAP</td>
<td>MV демо</td>
<td>CUSTOMER_MAP</td>
<td>10</td>
<td>true</td>
<td>266</td>
<td>266 true</td>
</tr>
<tr>
<td>DEMO_MAP</td>
<td>MV демо</td>
<td>DEMO_MAP</td>
<td>10</td>
<td>true</td>
<td>266</td>
<td>266 true</td>
</tr>
<tr>
<td>US_BASE_MAP</td>
<td>SCOTT</td>
<td>US_BASE_MAP</td>
<td>10</td>
<td>true</td>
<td>266</td>
<td>266 true</td>
</tr>
</tbody>
</table>
Manage Map Tiles
Set up MapViewer server
Set up MapViewer

- Can be deployed to any Java EE server
- Create data sources
- Quick tests with jview
- Explore the config file: mapViewerConfig.xml
- Use the admin web pages
The MapViewer Installation Kit

- Download from:

- A MapViewer installation includes the following components:
  - An enterprise archive (mapviewer.ear) file
  - The standalone map definition tool (mapbuilder.jar)
Use the quickstart kit with a standalone OC4J or ...
Using WebLogic Server

The following screenshots are for WLS 11g (10.3)
Installing the WebLogic Server

- Provided as a single ZIP file (wls1035_dev.zip)
- Platform-independent, use for development only
- Requires a full JDK 1.6
- Can also use a platform-specific kit with installer and JDK
  - For example wls1035_oepe111150_win32.exe for windows
- Can also install JDeveloper Studio Edition and use its embedded WLS server.
1. Unzip `wls1035_dev.zip` into a directory. This is now the MW_HOME directory

2. Configure (set ACLs, generate registry.xml, etc)

```bash
set MW_HOME= D:\Oracle\wls1035_dev
set JAVA_HOME=C:\Program Files\Java\jdk1.6.0_22
cd %MW_HOME%
call configure.cmd
```

3. Create a new domain (or use the wizard …)

```bash
mkdir wls_domains\mapviewer
```
Using the Domain Configuration Wizard

- Start the wizard

```plaintext
set JAVA_HOME=C:\Program Files\Java\jdk1.6.0_22
set MW_HOME=D:\Oracle\wls1035_dev
call "%MW_HOME%\wlserver\server\bin\setWLSEnv.cmd"
call %MW_HOME%\wlserver\common\bin\config.cmd
```

- Answer the questions
- Specify the name of the domain
- Specify the name of the admin user and password
- Will create the directory for the domain
Using the Domain Configuration Wizard
WebLogic Administration

http://localhost:7001/console
WebLogic Administration
Mapviewer Installation Steps

1. Launch WebLogic Server administration console (http://localhost:7001/console)

3. Manage deployments
Mapviewer Installation Steps

3. Click on “Install”
Mapviewer Installation Steps

4. Browse to the “mapviewer.ear” directory and select it.
Mapviewer Installation Steps

5. Install as an application

Install Application Assistant

Back  Next  Finish  Cancel

Choose targeting style
Targets are the servers, clusters, and virtual hosts on which this deployment will run. There are several ways you can target an application.

- Install this deployment as an application

The application and its components will be targeted to the same locations. This is the most common usage.

- Install this deployment as a library

Application libraries are deployments that are available for other deployments to share. Libraries should be available on all of the targets running their referencing applications.

Back  Next  Finish  Cancel
6. **Set “Source Accessibility”**

- Use the “mapviewer.ear” directory as deployment location
- This causes the unpacked MapViewer location to become the "working" directory of MapViewer.
- It makes it easier if you want to upgrade MapViewer in the future, in which case you simply unpack the new mapviewer.ear file to this directory and restart WebLogic Server.
Mapviewer Installation Steps

7. Installation completed
Mapviewer Installation Steps

• All files used by MapViewer are now in directory ....../mapviewer.ear/
• Web files are in directory ....../mapviewer.ear/web.war/WEB-INF/
• This directory contains the following sub-directories
  – conf = configuration file (mapViewerConfig.xml)
  – log = log files (mapviewerN.log)
  – admin = SQL scripts
Configuration and Administration
MapViewer Installation “Home”

• For OC4J
  – ..../j2ee/home/applications/mapviewer/web/
  – This is inside your OC4J server

• For WebLogic Server
  – ..../mapviewer.ear/web.war/
  – This can be anywhere.

• For Glassfish
  • ..../glassfish3/glassfish/domains/domain1/applications/mapviewer/web_war
  • This is inside your Glassfish server. “domain1” is the default domain.
## Default MapViewer URLs

<table>
<thead>
<tr>
<th>Application Server</th>
<th>URL Type</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MapViewer access</td>
<td><a href="http://localhost:8888/mapviewer">http://localhost:8888/mapviewer</a></td>
</tr>
<tr>
<td></td>
<td>Server administration</td>
<td><a href="http://localhost:7001/console">http://localhost:7001/console</a></td>
</tr>
<tr>
<td></td>
<td>MapViewer access</td>
<td><a href="http://localhost:7001/mapviewer">http://localhost:7001/mapviewer</a></td>
</tr>
<tr>
<td></td>
<td>Server administration</td>
<td><a href="http://localhost:4848">http://localhost:4848</a></td>
</tr>
<tr>
<td></td>
<td>MapViewer access</td>
<td><a href="http://localhost:8080/mapviewer">http://localhost:8080/mapviewer</a></td>
</tr>
</tbody>
</table>
Database Configuration

Create views and tables used for managing map caches

- Views and tables used for holding map cache definitions
- USER_SDO_CACHED_MAPS and USER_SDO_TILE_ADMIN_TASKS
- Automatically provided with 11.1.0.7 and later versions
- Must be manually added in 11.1.0.6 or earlier versions
- Script in $MAPVIEWER_HOME/WEB-INF/admin/mcsdefinition.sql
- Run as “SYSTEM” or “SYS”

```
SQL> connect system/******
SQL> @mcsdefinition.sql
```
MapViewer Configuration File

- File “mapViewerConfig.xml”
  - In $MAPVIEWER_HOME/WEB-INF/conf
- Set tracing and logging options
- Set permanent data sources
- Many other settings
  - Map cache location, data cache size, …
  - WMS parameters, proxy server, …
- Edit manually or via the Mapviewer console
- Restart Mapviewer for the changes to take effect
MapViewer Console
http://<server>/mapviewer

- Demonstrations
- API Documentation
- Administration
- Use the “Admin” button
- Must log in as administrator.
Mapviewer Administration

- Manage configuration
- Manage data sources
- Manage map caches
Manage Configuration

Edit the content of the configuration file

Then click on “Save & Restart”
Set Logging and Tracing

- Logging level
- Log destinations

```xml
<logging log_level="finest" log_thread_name="true"
         log_time="true">
    <log_output name="System.err" />
    <log_output name="../log/mapviewer.log" />
</logging>
```

- Log level “finest” good for debugging
- Will show each and every request and response
- Will show each SQL statement
Define a Data Source

- Specify JDBC connection parameters
- Host, port, sid
- Username and password

```xml
<map_data_source name="scott"
    jdbc_host="127.0.0.1"
    jdbc_port="1521"
    jdbc_sid="orcl111"
    jdbc_user="scott"
    jdbc_password="!tiger"
    jdbc_mode="thin"
    max_connections="5"
    number_of_mappers="3"
    allow_jdbc_theme_based_foi="true"
/>
```

**Database connection**

**Username and password**
Password preceded with “!”. Will be encrypted in the file.

**NOTE for OC4J**
OC4J uses a JDBC driver for database 10g
This driver converts your passwords to lowercase when connecting.

=> Use only lowercase passwords!
Define a Data Source

• Can also use a container-managed data source

```xml
<map_data_source name="scott"
    container_ds="jdbc/scottDS"
    max_connections="5"
    number_of_mappers="3"
    allow_jdbc_theme_based_foi="true"
/>
```
Dynamic Data Sources

Data source definition is temporary.
Will disappear on next restart
Using the “Jview” demonstrator
http://<server>/mapviewer/demo/jview.jsp
Using the “Jview” demonstrator
Prepare your spatial data
Prepare your spatial data

• Mostly database work
• Import data into a schema
• Ensure proper metadata and indexing
• Think about schema separation
  – Styles can be shared across schema
  – Themes, basemaps and tile layers cannot be shared
  – Themes can use base table from another schema
Next Steps

• Set up and review the Oracle Maps Tutorials
  – host:port/mapviewer, click on Oracle Maps Tutorial link
  – Install mvdemo dataset as described on the demo setup page
    host:port/mapviewer/fsmc/tutorial/setup.html

• Requires the mvdemo dataset from OTN
Learning Oracle Maps

• Install the Oracle Maps tutorial
  – Check the instructions for the setup
  – Requires importing a sample dataset

• Run the tutorial
  – Step-by-step introduction
  – Examples with associated source code
  – Progresses from simple to more complex examples
Installing the Oracle Maps Tutorial

1. Download from Mapviewer Sample code:
   - http://download.oracle.com/otn/other/mapviewer/mvdemo.zip

2. Create user MVDEMO
   - Grant basic privileges (resource, connect, create view)

3. Import MVDEMO.DMP into user MVDEMO
   - Creates tables CITIES, STATES, COUNTIES, INTERSTATES and some more

4. Run script MVDEMO.SQL
   - Populates CUSTOMERS,
   - Creates spatial indexes
   - Creates styles, themes, maps and map caches.

5. Define MapViewer data-source MVDEMO
Fully Functional Tutorials
# Oracle Maps demos

<table>
<thead>
<tr>
<th>Id</th>
<th>Demo</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Display Map</td>
<td>This example shows how to display a map.</td>
</tr>
<tr>
<td>2</td>
<td>Zoom In/Out</td>
<td>This example shows how to add zoom control outside the map area.</td>
</tr>
<tr>
<td>3</td>
<td>Navigation Panel</td>
<td>This example shows how to add the in-map navigation panel.</td>
</tr>
<tr>
<td>4</td>
<td>Theme Based FOI layer</td>
<td>This example shows how to add/remove a Theme Based FOI layer to the map. Note that this particular FOI layer is associated with the theme 'customers' in the datasource 'indus'.</td>
</tr>
<tr>
<td>5</td>
<td>Theme Based FOI layer visibility</td>
<td>This example shows how to show/hide an existing Theme Based FOI layer.</td>
</tr>
<tr>
<td>6</td>
<td>Custom Marker for a Theme Based FOI layer</td>
<td>This example shows how to set a custom marker for a Theme Based FOI layer.</td>
</tr>
<tr>
<td>7</td>
<td>Event Listeners for a Theme Based FOI layer</td>
<td>This example shows how to setup custom event listeners for a Theme Based FOI layer.</td>
</tr>
<tr>
<td>8</td>
<td>Get FOI attributes from a Theme Based FOI layer</td>
<td>This example shows how to get the attributes that are returned from query results for specific FOI objects in a Theme Based FOI layer.</td>
</tr>
<tr>
<td>9</td>
<td>Refresh a Theme Based FOI layer</td>
<td>This example shows how to refresh a Theme Based FOI layer.</td>
</tr>
<tr>
<td>10</td>
<td>Templated Theme Based FOI layer</td>
<td>This example shows how to use a templated Theme Based FOI layer. A templated Theme Based FOI layer is associated with a MapViewer theme whose query condition contains binding variables. You can supply actual values for these binding variables at run-time in the client API, so that MapViewer only fetches features that satisfy these conditions based on the supplied values.</td>
</tr>
<tr>
<td>11</td>
<td>Add/Remove, Show/Hide FOI</td>
<td>This example shows how to add/remove, show/hide individual FOI. Note that MapViewer will automatically transform the FOI geometry coordinates if the kind of FOI geometry is different from that of the base map.</td>
</tr>
<tr>
<td>12</td>
<td>FOI Utilities</td>
<td>This example shows some other utility methods to customize various aspects of FOI interaction.</td>
</tr>
<tr>
<td>13</td>
<td>Marker FOI</td>
<td>This example shows how to create and display custom marker features on the map.</td>
</tr>
</tbody>
</table>
Resources

http://www.oracle.com/technetwork/database/options/spatial

Examples, white papers, downloads, discussion forum, sample data ....
Q&A