Building HTML5 Maps with Oracle Maps API

LJ Qian, Director of Software Development
Oracle
The Spatial & Graph SIG User Group

• The SIG promotes interaction and communication that can drive the market for spatial technology and data

• Members connect and exchange knowledge via online communities and at annual conferences and events

• Meet us here at the Summit
  
  Morning Reception
  Tuesday and Wednesday
  7:45 to 8:30 a.m.
  Registration Area

  Social Hours
  Tuesday and Wednesday
  6 to 7 p.m.
  Registration Area

• Join us online
  – LinkedIn (search for “LinkedIn Oracle Spatial”)
  – Google+ (search for “Google+ Oracle Spatial”)
  – IOUG SIG (sign up for free membership through www.ioug.org)
  – OTN Spatial – Communities (search for “Oracle Spatial and Graph Community”)

• Contact the Board at oraclespatialsig@gmail.com
Agenda

- Oracle Maps HTML5 API: an overview
- Deep Dive
- Extensibility
- Demos
- Roadmap
- Resources
- Q & A
Oracle Maps HTML5 API: an overview

• What’s with all the different names?
• What is it?
• Who can use it?
• Why use it?
• How to use it?
Oracle Maps HTML5 API: an overview

What’s with all the different names?

- Oracle Maps V2 API
- Oracle Maps HTML5 API
- Oracle MapViewer V2 JavaScript API

All of the above are essentially the same thing!

There is an older MapViewer JavaScript mapping lib (aka the V1 API), that relies mostly on server side rendering. While it will continue to be supported and maintained, the V2 API is preferable.
Oracle Maps HTML5 API: an overview

What is it?

• A comprehensive JavaScript API for Mapping, Visualization and Editing
• Makes heavy use of modern browser’s HTML5 capabilities
• Works seamlessly with Oracle Spatial & Graph and MapViewer
• Works equally well standalone!
• Suitable for both desktop and mobile Web applications
Oracle Maps HTML5 API: an overview

Who can use it?

• Spatial & Graph app developers
• BI app developers
• APEX developers
• Oracle FMW web app developers
• Mobile app developers
• Basically any enterprise Web app developers who want to:
  • mash up business data on maps
  • interactively manipulate/edit geo spatial data
  • integrate multiple geo spatial data sources and layers into one map
Oracle Maps HTML5 API: an overview

Why use it?

• Rich set of functions out of the box
• Wide range of mash up capabilities
• Seamless integration with your backend Oracle Spatial & Graph database
• High performance with built-in client side spatial index
• Flexible thematic mapping capabilities
• Powerful geometry and feature data editing capabilities
• Interfaces that allow developers to extend the base functions
Oracle Maps HTML5 API: an overview

How to use it?

• Import or embed the JavaScript library into your Web app
• Create an HTML container element (DIV) for the map
• Create an instance of the OM.Map class using the DIV
• Hook it up with a MapViewer server, or, use it standalone
• Add a background map tile layer or two (or ten!)
• Add one or more interactive vector (geometry) layers
• Off you go!
Agenda

• Oracle Maps HTML5 API: an overview
• Deep Dive
• Extensibility
• Demos
• Roadmap
• Resources
• Q & A
Deep Dive
Main Components

<Map (OM.Map)>

1 Universe

* Layers

* Tile layers

* Vector Layers

* Controls, Decorations, Tools
var div = document.getElementById('map');
var map = new OM.Map(div, {mapviewerURL: ''});
var tileLayer = new OM.layer.ElocationTileLayer();
map.addLayer(tileLayer);
addMarker(...); //actual code skipped for brevity
map.setZoomLevel(3);
map.init();
** Universe *(OM.universe.)*

- Defines overall map bounds and zoom levels
- Explicitly specified (on creating a new OM.Map instance)
- Or implicitly specified when adding a Tile Layer.
- Affects zoom slider
- May change when switching Tile Layers of different SRS

```javascript
var myUniverse = new OM.universe.Universe(
  { srid : 8307,
    bounds : new OM.geometry.Rectangle
       (-180, -90, 180, 90, 8307),
    numberOfZoomLevels: 19
  });
var map = new OM.Map(mapDiv,
  {universe:myUniverse});
```
Deep Dive

Everything is a layer!

- Map contents are always represented as layers
- Even a single marker is (added as part of) a Layer!
- Tile Layer: displays tile images from tile services
- Vector Layer: displays interactive geometries
Tile Layers

Tile Layer
(OM.layer)

MapViewer Native Tile Layers

Oracle Spatial & Graph

Google Maps
OSM Maps
Here Maps
Bing Maps
LocationBox Tile Layers
TomTom Maps
Custom Tile Layers

Written by you using 3rd party tile service APIs
Vector Layers

- Displays geometries in specific **Styles**
- Supports programmatic or user interactions: select, hover, edit, filter
- Provides event handling
- Supports many geometry exchange formats
  - geoJson, geoRss, KML, WKT …
- A single class to learn: OM.layer.VectorLayer
Vector Layer

Supported Data Sources

- geoJson feeds
- KML feeds
- GeoRSS feeds
- WKT feeds
- Programmaticaly created geoms

Create geometries using the OM.geometry.* constructors
Vector Layer

Main components

Vector Layer

Feature
Feature
Feature
Feature
Feature
(OM.Feature)

Geometry

Properties
{state:"Hawaii"}
Vector Layer

Events emitted by vector layers

Layer Wide Events
- LAYER_INIT
- FEATURES_LOADED
- FEATURES_DISPLAYED
- LAYER_Z_INDEX_CHANGED
- LAYER_REMOVED
- LAYER_VISIBILITY_CHANGED
- BEFORE_REFRESH
- AFTER_REFRESH
- RENDERING_STYLE_CHANGED
- ...

Feature Level Events
- FEATURE_SELECTED
- FEATURE_DESELECTED
- FEATURE_MOVED
- FEATURE_EDITED
- FEATURE_MOVING

```javascript
layer.addEventListener(FEATURE_SELECTED, function(evt){
    alert("You selected:"+evt.selectedFeature.id);
});
```
Vector Layer

Tooltip and Info-window

- Info-window is a dialog displayed on mouse click
- Typically shows all the properties of the feature
- You can customize the format and style of a layer’s infowindow;
- InfoWindow can have multiple Tabs
- You can even replace the info window with any 3rd party UI dialogs
- Tooltip is a small popup displayed on mouse hover
- Typically shows the label text of the feature
- Customize with setToolTipCustomizer()
Vector Layer
Rich set of built-in functionalities

Vector Layer

- Auto clustering
- Indicator features
- On Demand Loading
- Inject external attribute data
- ...
- ...

Oracle Spatial Summit 2015
Vector Layer

Styles: control how features are rendered

```javascript
var style1 = new OM.style.Color({ stroke:"#ac9898",fill:"#2bec6"});
layer.setRenderingStyle(style1);
```

- **Vector Layer**
  - Rendering style
  - Selection style
  - Hover style
  - Label style
Style

• An instance of one of the OM.style.* classes
• Tells Rendering Engine how to draw a feature
• Many different types:
  • COLOR, LINE, MARKER, PIE/BAR CHART, HEAT MAP, TEXT, AREA, GRADIENT, ADVANCED (BUCKET) …
• Supports animation and effects such as drop shadow
• Applied to Features of a Vector Layer at run time
• Can be based on server side (Map Builder created) styles

To change how a Vector Layer displays its features, change its styles
Styles
# Style

## Basic Styles’ Key Properties

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Fill</td>
<td>Width</td>
<td>Stroke</td>
<td>FontFamily</td>
<td>Stroke</td>
</tr>
<tr>
<td>Stroke</td>
<td>Height</td>
<td>StrokeThickness</td>
<td>FontStyle</td>
<td>StrokeThickness</td>
</tr>
<tr>
<td>StrokeThickness</td>
<td>Text</td>
<td>StrokeOpacity</td>
<td>FontWeight</td>
<td>FillOpacity</td>
</tr>
<tr>
<td>FillOpacity</td>
<td>OffsetX/Y</td>
<td>StrokeDash</td>
<td>FontSize</td>
<td>StrokeOpacity</td>
</tr>
<tr>
<td>StrokeOpacity</td>
<td>ImageUrl</td>
<td>StartMarker</td>
<td>SizeUnit</td>
<td>StrokeOpacity</td>
</tr>
<tr>
<td>OutlineColor</td>
<td>VectorDefinition</td>
<td>EndMarker</td>
<td>Fill</td>
<td>StrokeDash</td>
</tr>
<tr>
<td>StrokeDash</td>
<td>SizeUnit</td>
<td>OrientedMarker</td>
<td>IsSticky</td>
<td>PatternImage</td>
</tr>
<tr>
<td>Gradient</td>
<td>Gradient</td>
<td>CenterLine</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Style
### Advanced Styles

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ColorStops</td>
<td>BarArray</td>
<td>ColorStyleArray</td>
<td>DefaultStyle</td>
</tr>
<tr>
<td>SpotlightRadius</td>
<td>Height</td>
<td>Stroke</td>
<td>NumClasses</td>
</tr>
<tr>
<td>lengthUnit</td>
<td>Width</td>
<td>StrokeThickness</td>
<td>DefaultColor</td>
</tr>
<tr>
<td>Opacity</td>
<td>LengthUnit</td>
<td>StartingAngle</td>
<td>BaseColor</td>
</tr>
<tr>
<td>ContainerLayer</td>
<td>MinLength</td>
<td>Radius</td>
<td>FromColor</td>
</tr>
<tr>
<td>MinMaxValues</td>
<td>MaxLength</td>
<td>Direction</td>
<td>ToColor</td>
</tr>
<tr>
<td>SampleFactor</td>
<td>ShowXAxis</td>
<td>Effects</td>
<td>BucketArray</td>
</tr>
<tr>
<td></td>
<td>ShowYAxis</td>
<td>OrientedMarker</td>
<td>Classification</td>
</tr>
<tr>
<td></td>
<td>EnableHighlight</td>
<td>Algorithm</td>
<td>Algorithm</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>O.s.BucketStyle</th>
<th>O.s.LinearGradient</th>
<th>O.s.RadialGradient</th>
</tr>
</thead>
<tbody>
<tr>
<td>StyleArray</td>
<td>ColorStops</td>
<td>ColorStops</td>
</tr>
<tr>
<td>BucketArray</td>
<td>Opacity</td>
<td>Opacity</td>
</tr>
<tr>
<td>Classification</td>
<td>Stroke*</td>
<td>Stroke*</td>
</tr>
<tr>
<td>Algorithm</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td></td>
<td>Gradient</td>
<td>Gradient</td>
</tr>
<tr>
<td></td>
<td>Effects</td>
<td>Effects</td>
</tr>
</tbody>
</table>

```
OM.style.HeatMap
- ColorStops
- SpotlightRadius
- lengthUnit
- Opacity
- ContainerLayer
- MinMaxValues
- SampleFactor

OM.style.BarChart
- BarArray
- Height
- Width
- LengthUnit
- MinLength
- MaxLength
- ShowXAxis
- ShowYAxis
- EnableHighlight

O.s.ColorScheme
- ColorStyleArray
- NumClasses
- DefaultColor
- BaseColor
- FromColor
- ToColor
- BucketArray
- Classification
- Algorithm

O.s.VariableMarker
- DefaultStyle
- NumClasses
- MarkerStyle
- StartSize
- Increment
- Buckets
- Classification
- Algorithm
```

```
O.s.BucketStyle
- StyleArray
- BucketArray
- Classification
- Algorithm
- Low
- High
- Gradient
- Effects

O.s.LinearGradient
- ColorStops
- Opacity
- Stroke*
- FillOpacity

O.s.RadialGradient
- ColorStops
- Opacity
- Stroke*
- FillOpacity
```
Thematic Mapping

• Certain aspects (color, size, etc) of features’ display are based on their property values (sales, population etc.)

• Two ways of achieving thematic mapping
  • Use advanced styles such as BucketStyle
  • Use Style Binding
Thematic Mapping

Using Advanced Styles

Create a Vector Layer

Create an Advanced Style

Join style with feature properties

Layer feature properties

var layer = new OM.layer.VectorLayer(…);

var style1 = new OM.style.BucketStyle(…);

layer.setRenderingStyle(style1, ['sales']);
Thematic Mapping
Similarly for Heat maps

Create a Vector Layer of **Points**

Create a Heat Map Style

Set style on the layer

```javascript
// creates a layer of customer locations
var layer = new OM.layer.VectorLayer(…);

var style1 = new OM.style.HeatMap({
    spotlightRadius:90,
    lengthUnit:"kilometer",
    colorStops:colors,
    opacity:0.5,
    maxValue:200,
    minValue:10,
    styleAttributes:['Sales']
});

layer.setRenderingStyle(style1);
```

containerVectorLayer:

statesLayer
Thematic Mapping

Two types of Heatmaps are supported

Location/Density based

Metric based
Thematic Mapping

Using Style Bindings

• Binds sets of data values to one or more visual aspects of a Style

• Visual aspects bind-able by style:
  • Marker styles: Fill, Size, Width, Height
  • Color styles: Fill, Stroke, StrokeThickness
  • Line styles: Fill, FillWidth
  • BarChart styles: Size, Width, Height
  • PieChart styles: Radius, Size
Thematic Mapping

Style Binding: three players

- **Data Wrapper**: a lightweight wrapper of arbitrary application data set or data service. Instance of `OM.Column`.
- **Data Formatter**: a function that converts/maps measure or categorical values into visual values (e.g. sizes or fill colors)
Thematic Mapping

Style Binding: sample code

Arbitrary application data structure:
var myData = [
{“name”:”GM”, “sales”:”175”},
{“name”:”FORD”, “sales”:”215”}
…
]

var aColumn = new OM.Column(
{data: myData,
keyGetter: function(){return this.name;},
valueGetter: function(){return this.sales;}
});

var aFormatter = new OM.style.ColorFormatter(
{colors: OM.style.colorbrewer.OrRd["7"],
scale: ‘log’ //map the colors to the values using a logarithmic scale
});

circle.bindData(‘Fill’, aColumn, aFormatter);

//creates a vector Marker
var circle = new OM.style.Marker(…);
Thematic Mapping

One Style, multiple Bindings

- A single Style can have multiple bindings
- Each bound data set drives one visual aspect of the Style during rendering

‘Sales’ drive Marker Sizes

‘Account Manager’ names drive Marker Colors
Deep Dive

Map Controls and Decorations

- Dynamically created and added to the map
- Normally anchored to fixed positions on the map window
- Can be draggable as well
- Map Decoration is the generic control interface
- Many specialized controls available
- Positioning & display styles are customizable
Deep Dive

Map Controls and Decorations

OM.Map

Map Decoration

Custom map controls

Copyright

LayerControl

Magnifier

Navigation Controls

ScaleBar

Overview Control

Toolbar
Deep Dive

Built in map tools

• Tools support interactive drawings on the map
• Can be programmatically triggered, controlled and accessed
• Geometry editing is done using the Redline Tool and many others
• Listener functions provided by most Tools
Deep Dive

Built in map tools

OM.Map

Map Decoration

Custom map controls

Circle Tool

Rectangle Tool

Distance Tool

Marquee Zoom Tool

Void Polygon Tool

White Board Tool

Toolbar
Deep Dive

Client Side Filtering

• Supports filtering of a VectorLayer’s features
• Filter by simple expressions (a > b, a==d, a like b) on attributes
• Filter by geometry: insidePolygon, AnyInteract
• Composite filter: combining multiple filters
• Typically applied to a Vector Layer (‘s Features)
• Applying a filter on a layer produces either a new layer or reduces current layer’s feature set
Deep Dive
Client Side Filters

OM.filter.Filter

Greater, Less
Like, Between, Or, And
Not, Any, NotEquals
...
AnyInteract
InsidePolygon

```javascript
var polygon = polygonTool.getGeometry();
var filter1 = new OM.filter.InsidePolygon(polygon);
var vectorlayer2 = vectorlayer1.applyFilter(filter1, false);
vectorlayer2.setRenderingStyle(pinStyle);
```
Deep Dive

Events and Listeners

- On zoom & pan
- On addition or removal of map content
- When Map Tools and Controls are used
- When the user interacts with features and layers
- Touch events on mobile devices are translated into mouse events
- You can listen to and handle all of the above events
- Most classes are a type of EventSource, meaning you can attach/remove listeners to/from them
Deep Dive

Visual effects (aka filters)

- Two main types:
  - OM.visualfilter.Glow
  - OM.visualfilter.DropShadow
- Enhances rendering appeal of a vector layer’s features
- Can be applied in two ways:
  - As effect(s) of a Style object
  - Directly set on a VectorLayer via setVisualFilter()
- Use a Gradient style for a Gradient effect on a vector layer
## Deep Dive
### Main Public Packages and classes

<table>
<thead>
<tr>
<th>Feature Area</th>
<th>Packages &amp; Key Classes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main API classes</td>
<td>OM, OM.Map, OM.Feature, OM.gv</td>
<td>Key top level classes</td>
</tr>
<tr>
<td>Controls, decorations</td>
<td>OM.control.*</td>
<td>Map controls</td>
</tr>
<tr>
<td>Event types and listeners</td>
<td>OM.event.*, OM.event.EventSource</td>
<td>Defines event types and EventSource class</td>
</tr>
<tr>
<td>Filters</td>
<td>OM.filter.*</td>
<td>Client side filtering</td>
</tr>
<tr>
<td>Geometry</td>
<td>OM.geometry.*</td>
<td>Geometry data structures</td>
</tr>
<tr>
<td>InfoWindow classes</td>
<td>OM.infowindow.*</td>
<td>InfoWindow and Tabbed InfoWindow classes</td>
</tr>
</tbody>
</table>
# Deep Dive

## Main Public Packages and classes

<table>
<thead>
<tr>
<th>Feature Area</th>
<th>Packages &amp; Key Classes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Map layers</td>
<td>OM.layer.*, TileLayer, VectorLayer</td>
<td>All types of map layers</td>
</tr>
<tr>
<td>Built-in LBS functions</td>
<td>OM.lbs.*</td>
<td>Classes for accessing maps.oracle services (GeoCoding etc)</td>
</tr>
<tr>
<td>Map projections</td>
<td>OM.proj.*</td>
<td>Client side map projections</td>
</tr>
<tr>
<td>Layer Styles</td>
<td>OM.style.*</td>
<td>Color, marker, text, and advanced styles</td>
</tr>
<tr>
<td>Templates</td>
<td>OM.template.*</td>
<td>Map templates</td>
</tr>
</tbody>
</table>
### Deep Dive

**Main Public Packages and classes**

<table>
<thead>
<tr>
<th>Feature Area</th>
<th>Packages &amp; Key Classes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Built in tools</td>
<td>OM.tool.*</td>
<td>Map Tools</td>
</tr>
<tr>
<td>Map Universe</td>
<td>OM.universe.*</td>
<td>Map universe definitions</td>
</tr>
<tr>
<td>Visual effects</td>
<td>OM.visualfilter.*</td>
<td>Visual effects (DropShadow)</td>
</tr>
<tr>
<td>Utilities</td>
<td>OM.util.*</td>
<td>Utility classes</td>
</tr>
</tbody>
</table>
Deep Dive

Mash up possibilities

HTML5 Apps
- Oracle Maps v2 API

Server components
- Tile Server
- Mapping Engine
- Oracle Database
  - OGR adapters
  - WMS
  - WFS
  - GeoRSS
  - WMTS

Map Data Server
- geoJson*

Data packs / Templates
- Oracle
- Here
- TomTom
- Bing
- OSM
- et al

* With some extensions

MapViewer Editor

Tile layers

Oracle Spatial Summit 2015
Deep Dive

Many different use cases

• Without backend databases
• Without MapViewer server
• Import the v2 API from maps.oracle.com
• Embed the v2 API files directly in your app
• Run everything on a Mobile device
Deep Dive

Miscellaneous (global) configurations

- Configure API resource paths
- Change default AJAX HTTP method (GET or POST)
- Change API internal logging levels
- Switch between SVG or Canvas renderer
- Use https for loading external images
- Change default marker and label text styles

Most global config are set in the OM.gv class
Agenda

- Oracle Maps HTML5 API: an overview
- Deep Dive
- **Extensibility**
- Demos
- Roadmap
- Resources
- Q & A
Extensibility

Enhance, extend existing functions

- Register custom SRS and CS transformation functions
- Interface to use 3rd party tile services’ native APIs
- Info Window: fully customizable L&F; or replace with 3rd party dialogs
- Custom HTML content as styles (eg. D3.js charts on Features)
- Add custom HTML content as Map Decorations
- Extend the Redline tool for custom editing functions
- Customize Style Binding behaviors using a custom formatter
Agenda

- Oracle Maps HTML5 API: an overview
- Deep Dive
- Extensibility
- Demos
- Roadmap
- Resources
- Q & A
Agenda

- Oracle Maps HTML5 API: an overview
- Deep Dive
- Extensibility
- Demos
- **Roadmap**
- Resources
- Q & A
Roadmap

• Improve performance with large # of features
• Enhance geometry & feature editing capabilities
• Enhance built-in map legend and layer control
• Templates and wizards for rapid application development
• Make API more modular with selective loading
• Vector tile support for enhanced performance
• UTF grid support and interactive tile layers
• Utilize more HTML5 functions in modern browsers
• Support WMS type services as tile layers
Agenda

• Oracle Maps HTML5 API: an overview
• Deep Dive
• Extensibility
• Demos
• Road Maps
• **Resources**
• Q & A
Resources: Oracle Technology Network

- www.oracle.com/technetwork/database/options/spatialandgraph
- www.oracle.com/technetwork/middleware/mapviewer
- blogs.oracle.com ➔ oraclespatial ➔ oracle_maps_blog
- Public demos: http://slc02okf.oracle.com
More Resources
Certification

• **Individual Certification, Partner Specialization**
  • Credentials for individuals with Spatial implementation expertise
  • OPN Specialization – differentiates partner organizations delivering Spatial services
  - Talk to Oracle team this week