## Contents

# Module 2.3 - Web-based Mapping Clients: OpenLayers Javascript Framework - Part II

### Outline

- More detailed Map Object Options
- More detailed Layer Object Options
- Additional Map Layer Types - With Examples

### Map Object Options

Map Object Options API Reference

Two methods for constructing a new `OpenLayers.Map` object

```
    // create a map with default options in an element with the id "map1"
    var map = new OpenLayers.Map("map1");

    // create a map with non-default options in an element with id "map2"
    var options = {
        maxExtent: new OpenLayers.Bounds(-200000, -200000, 200000, 200000),
        maxResolution: 156543,
        units: 'm',
        projection: "EPSG:41001"
    };
    var map = new OpenLayers.Map("map2", options);

    // map with non-default options - same as above but with a single argument
    var map = new OpenLayers.Map({
        div: "map_id",
        maxExtent: new OpenLayers.Bounds(-200000, -200000, 200000, 200000),
        maxResolution: 156543,
        units: 'm',
        projection: "EPSG:41001"
    });
```

### Map Object Options - continued

Excerpts from the API documentation

**`allOverlays`** {Boolean} Allow the map to function with "overlays" only. Defaults to false. If true, the lowest layer in the draw order will act as the base layer. In addition, if set to true, all layers will have isBaseLayer set to false when they are added to the map.

**`div`** {DOMElement|String} The element that contains the map (or an id for that element). If the OpenLayers.Map constructor is called with two arguments, this should be provided as the first argument. Alternatively, the map constructor can be called with the options object as the only argument. In this case (one argument), a div property may or may not be provided. If the div property is not provided, the map can be rendered to a container later using the render method.

### Map Object Options - continued

**`layers`** {Array(OpenLayers.Layer)} Ordered list of layers in the map

**`tileSize`** {OpenLayers.Size} Set in the map options to override the default tile size for this map.

**`projection`** {String} Set in the map options to override the default projection string this map - also set `maxExtent`, `maxResolution`, and `units` if appropriate. Default is "EPSG:4326".

**`units`** {String} The map units. Defaults to 'degrees'. Possible values are 'degrees' (or 'dd'), 'm', 'ft', 'km', 'mi', 'inches'.

### Map Object Options - continued

**`resolutions`** {Array(Float)} A list of map resolutions (map units per pixel) in descending order. If this is not set in the layer constructor, it will be set based on other resolution related properties (maxExtent, maxResolution, maxScale, etc.).

**`maxResolution`** {Float} Default max is 360 deg / 256 px, which corresponds to zoom level 0 on gmaps. Specify a different value in the map options if you are not using a geographic projection and displaying the whole world.

**`minResolution`** {Float}

**`maxScale`** {Float}

**`minScale`** {Float}

### Map Object Options - continued

**maxExtent** {OpenLayers.Bounds} The maximum extent for the map. Defaults to the whole world in decimal degrees (-180, -90, 180, 90). Specify a different extent in the map options if you are not using a geographic projection and displaying the whole world.

**minExtent** {OpenLayers.Bounds}

**restrictedExtent** {OpenLayers.Bounds} Limit map navigation to this extent where possible. If a non-null restrictedExtent is set, panning will be restricted to the given bounds. In addition, zooming to a resolution that displays more than the restricted extent will center the map on the restricted extent. If you wish to limit the zoom level or resolution, use maxResolution.

**numZoomLevels** {Integer} Number of zoom levels for the map. Defaults to 16. Set a different value in the map options if needed.

### Layer Object Options

Layer Object Options API Reference

Common Pattern of Layer Object Creation (varies some depending upon the specific layer type)

```
    new OpenLayers.Layer.*** (
        'layer name',
        'layer URL',
        {server-related options},
        {OpenLayers Layer Object options}
    )
```

### Layer Object Options - continued

**id** {String}

**name** {String}

**isBaseLayer** {Boolean} Whether or not the layer is a base layer. This should be set individually by all subclasses. Default is false

**displayInLayerSwitcher** {Boolean} Display the layer's name in the layer switcher. Default is true.

**visibility** {Boolean} The layer should be displayed in the map. Default is true.

### Layer Object Options - continued

**attribution** {String} Attribution string, displayed when an OpenLayers.Control.Attribution has been added to the map.

**projection** {OpenLayers.Projection} or {String} Set in the layer options to override the default projection string this layer - also set maxExtent, maxResolution, and units if appropriate. Can be either a string or an OpenLayers.Projection object when created – will be converted to an object when setMap is called if a string is passed.

**units** {String} The layer map units. Defaults to 'degrees'. Possible values are 'degrees" (or 'dd'), 'm', 'ft', 'km', 'mi', 'inches'.

**`scales`** {Array} An array of map scales in descending order. The values in the array correspond to the map scale denominator. Note that these values only make sense if the display (monitor) resolution of the client is correctly guessed by whomever is configuring the application. In addition, the units property must also be set. Use resolutions instead wherever possible.

### Layer Object Options - continued

**`resolutions`** {Array} A list of map resolutions (map units per pixel) in descending order. If this is not set in the layer constructor, it will be set based on other resolution related properties (maxExtent, maxResolution, maxScale, etc.).

**`maxExtent`** {OpenLayers.Bounds} The center of these bounds will not stray outside of the viewport extent during panning. In addition, if displayOutsideMaxExtent is set to false, data will not be requested that falls completely outside of these bounds.

**`minExtent`** {OpenLayers.Bounds}

**`maxResolution`** {Float} Default max is 360 deg / 256 px, which corresponds to zoom level 0 on gmaps. Specify a different value in the layer options if you are not using a geographic projection and displaying the whole world.

**`minResolution`** {Float}

### Layer Object Options - continued

**`numZoomLevels`** {Integer}

**`minScale`** {Float}

**`maxScale`** {Float}

**`displayOutsideMaxExtent`** {Boolean} Request map tiles that are completely outside of the max extent for this layer. Defaults to false.

**`transitionEffect`** {String} The transition effect to use when the map is panned or zoomed.

**There are currently two supported values** `null` No transition effect (the default).

`resize` Existing tiles are resized on zoom to provide a visual effect of the zoom having taken place immediately. As the new tiles become available, they are drawn over top of the resized tiles.

### Additional Map and Layer Object Functions & Events

Both Map and Layer Objects have a number of associated functions as well

- Retrieving object properties programmatically with `Get` functions.
- Modifying existing object properties with `Set` functions
- Map destruction, and reconfiguration
- Linkage of object events with Javascript functions

### WMS Layer Configuration

Some key issues to be aware of when using the WMS Layer Class:

- The *projection* of the map object must be supported by the included WMS service (review the WMS GetCapabilities response to see what projections are supported by the service)
- The *layers* parameter/property must be provided as part of the server-related property list (the layer names also come from the GetCapabilities response)
- Other WMS parameters may be provided as well to "adjust" the request automatically generated by OpenLayers

Sample WMS Layer Object Creation

```
countiesLayer = new OpenLayers.Layer.WMS(
    "US Counties",
    "http://webservices.nationalatlas.gov/wms?",
    {layers: "counties", version: '1.3.0', transparent: 'TRUE'},
    {isBaseLayer: false, visibility: false, opacity: .8}
);
map.addLayer(countiesLayer);
```

Example

### Vector Layer Configuration

Vector layers support

- External Data in a Variety of supported formats for both *reading* and *writing* (just a sample): ArcXML.Features, GeoJSON, GeoRSS, GPX, JSON, KML, WFS, WKT
- Directly encoded [geometries][OpenLayers.Geometry API Link]: Collection, Curve, LinearRing, LineString, MultiLineString, MultiPoint, MultiPolygon, Point, Polygon, Rectangle
- User created features, including support for interactive editing of features
- Styling of Vector features

### Vector Layer Configuration - Continued

Vector Layer Objects are Typically Defined using three OpenLayers classes

**`Protocol`** Connection protocol for requesting the data that would be provided from an external source

**`Format`** The OpenLayers supported format of the vector data object

**`Strategy`** A specification of how OpenLayers should request the data from the server, and also handle the data within the client (browser).

### Vector Layer Configuration - Continued

Sample Point Feature Object creation

```
    var Coord_classroom = new OpenLayers.Geometry.Point(-106.624073,35.084280);
    var Point_classroom = new OpenLayers.Feature.Vector(Coord_classroom);
    Layers["localFeatures"].addFeatures([Point_classroom])
```

Sample KML Layer Object creation

```
    Layers.counties = new OpenLayers.Layer.Vector("KML - Counties", {
        projection: map.displayProjection,
        strategies: [new OpenLayers.Strategy.Fixed()],
        protocol: new OpenLayers.Protocol.HTTP({
            url: "NMCounties.kml",
            format: new OpenLayers.Format.KML({
                extractAttributes: true
            })
        })
    });
    map.addLayer(Layers.counties)
```

Example