HERE Maps API - How to display KML file data on the map

Introduction

HERE Maps API version 2.2.4 offers support for KML data formatted files and displaying the containing data on the HERE Maps. This article will offer an overview to KML, structure and how it can be used within HERE Maps.

Summary

HERE Maps API supported web browser (basically any modern web browser)

Important note about maps credentials

Nokia provides several services options within the Maps API offering. The service is free to use, but you must obtain and use authentication and authorization credentials to use the services. Please read the Terms and Conditions and check the Pricing Plans page to decide which business model best fits your needs. Authentication requires unique Maps API credentials, namely an AppId and a token. You can get these credentials free for free following the instructions here

Implementation

HERE Maps API has built in support for KML. It is an abbreviation for Keyhole Markup Language, representing an XML notation for expressing geographic annotation and visualization within Internet-based, two-dimensional maps and three-dimensional Earth browsers. KML was developed as a Google Maps feature by company called Keyhole, Inc, later acquired by Google.

Today, KML is an international standard of the Open Geospatial Consortium (http://en.wikipedia.org/wiki/Open_Geospatial_Consortium) and is supported in the HERE Maps API.

Restrictions in HERE Maps

HERE Maps API does not currently support 3D so, these values if provided would be discarded by the API.- HERE Maps API does not currently support reading packaged KMZ files.

Structure

KML is structured as tag-based, with nested-elements and attributes and based on the XML standard. The tags are case-sensitive and they must appear exactly as they are listed in the official KML Reference, found within the OGC pages. The KML Reference indicates which tags are optional. Within a given element, tags must appear in the order shown in the Reference. The KML file can contain information for Markers, Places, images, Polygons, 3D models, textual descriptions, and so on. Common for all items placed is that they always have a longitude and a latitude. Additional data e.g. tilt, heading and altitude can be provided as "camera view". The KML file specifies a set of features (placemarks, images, polygons, 3D models, textual descriptions, etc.)

Example KML document

In order to load a KML file successfully, the KML file should be hosted on the same domain as the JavaScript or the results may be unpredictable. Most browsers will automatically prohibit cross-domain access. The usual method for doing this is to use relative addressing as shown.

```
kml.parseKML("./path/" + "kml_data_file.kml")
```

Tip: The KML reader of the HERE Maps API for JavaScript will ignore elements it is unable to interpret. KML readers vary and are more or less forgiving in the strictness of interpreting the KML specifications. It is recommended that you validate your KML syntax yourself through an online validator such as: http://feedvalidator.org

Shown below is a simple KML document containing data for one Placemark, for Berlin, Alexanderplatz.

Please note that by the KML specification the coordinate is given in format: longitude, latitude.

File: kml_data_file.kml

(C) Copyright Nokia 2014. All rights reserved.

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
    <Document>
        <Placemark>
            <name>Berlin</name>
            <description>Alexanderplatz, Berlin, Germany</description>
            <Point>
                  <coordinates>13.41156,52.526666</coordinates>
            </Point>
        </Placemark>
    </Document>
</kml>
```

Example code

This code only works if the the KML file to be parsed must be hosted on the same domain.

```
function loadKMLFile(map){
 var kml = new nokia.maps.kml.Manager();
 // We define a callback function for parsing kml file,
 // and then push the parsing result to map display
 // Add an observer to kml manager
 kml.addObserver("state", onKMLParsed);
 kml.parseKML("./kml/battles.kml");
}
```

Where the callback function `onKMLParsed()` is as defined below:

```
function onKMLParsed(kmlManager){
 var resultSet;
 // KML file was successfully loaded
 if (kmlManager.state == "finished") {
  // KML file was successfully parsed
  resultSet = new nokia.maps.kml.component.KMLResultSet(kmlManager.kmlDocument, map);
  resultSet.addObserver("state", function (resultSet) {
    if (resultSet.state == "finished") {
      // Retrieve map objects container from KML resultSet
      container = resultSet.container;

      // Add the container to the map's object collection so they will be rendered onto
the map.
      map.objects.add(container);

      // Switch the viewport of the map do show all KML map objects within the container
      map.zoomTo(container.getBoundingBox());
    }
  });
  resultSet.create();
 }
}
```

The following fully working example can be found at:

http://heremaps.github.io/examples/examples.html#load-kml-file

Adding cross domain support

(C) Copyright Nokia 2014. All rights reserved.

Cross domain support can be added using the jQuery library. The KML is loaded using the following syntax:

```
function loadCrossDomainKMLFile(map){
 jQuery.support.cors = true;
 $.ajax({
  type: "GET",
  url:
"http://api.maps.nokia.com/en/playground/examples/maps/res/kml/berlin_airport/schoenefeld.kml"
,
  dataType: "xml",
  success: onKMLFileLoaded,
  error : onKMLFileLoadError});
}
```

Where the callback functions `onKMLFileLoaded()` and `onKMLFileLoadError()` are as defined below:

```
function onKMLFileLoaded(xml)
{
 var kml = new nokia.maps.kml.Manager();
 kml.addObserver("state", onKMLParsed);
 doc =  xml.getElementsByTagName('Document')[0];
 kml.parse( doc);
}
```

```
function onKMLFileLoadError(err){
 alert("An Error has occurred.");
}
```

The `onKMLParsed()` is a callback function for the `kml.parse()` function, which displaye the parsed KML as shown:

```
function onKMLParsed(kmlManager){
 var resultSet;
 // KML file was successfully loaded
 if (kmlManager.state == "finished") {
  // KML file was successfully parsed
  resultSet = new nokia.maps.kml.component.KMLResultSet(kmlManager.kmlDocument, map);
  resultSet.addObserver("state", function (resultSet) {
    if (resultSet.state == "finished") {
      // Retrieve map objects container from KML resultSet
      container = resultSet.container;

      // Add the container to the map's object collection so they will be rendered onto
the map.
      map.objects.add(container);

      // Switch the viewport of the map do show all KML map objects within the container
      map.zoomTo(container.getBoundingBox());
    }
  });
  resultSet.create();
 }
}
```

The following fully working cross-domain example can be found at:

http://heremaps.github.io/examples/examples.html#load-kml-file-cross-domain

(C) Copyright Nokia 2014. All rights reserved.

Geodetic reference systems in KML

For its reference system, KML uses 3D geographic coordinates: longitude, latitude and altitude, in that order. The longitude, latitude components are defined by the World Geodetic System of 1984 (WGS84). The vertical component (altitude) is measured from the WGS84 EGM96 Geoid vertical datum. If altitude is omitted from a coordinate string, e.g. (-122.917, 49.2623) then the default value of 0 (approximately sea level) is assumed for the altitude component, i.e. (-122.917, 49.2623, 0) is assumed. A formal definition of the coordinate reference system (encoded as GML) used by KML is contained in the OGC KML 2.2 Specification. This definition references well-known EPSG CRS components.

OGC standard process

The KML 2.2 specification was submitted to the Open Geospatial Consortium to assure its status as an open standard for all geobrowsers. In November 2007 a new KML 2.2 Standards Working Group was established within OGC to formalize KML 2.2 as an OGC standard. Comments were sought on the proposed standard until January 4, 2008,[4] and it became an official OGC standard on April 14, 2008.[5]

For more on the HERE Maps API

Please check out the HERE Maps API full documentation and API reference here:

- HERE Maps API

You may also access the interactive API explorer

- API explorer

And additional examples on GitHub

- HERE Maps on Github

Tested with

Google Chrome: 20x Mozilla Firefox 12.0

(C) Copyright Nokia 2014. All rights reserved.