Archived:HERE Maps API - Passing Query Parameters

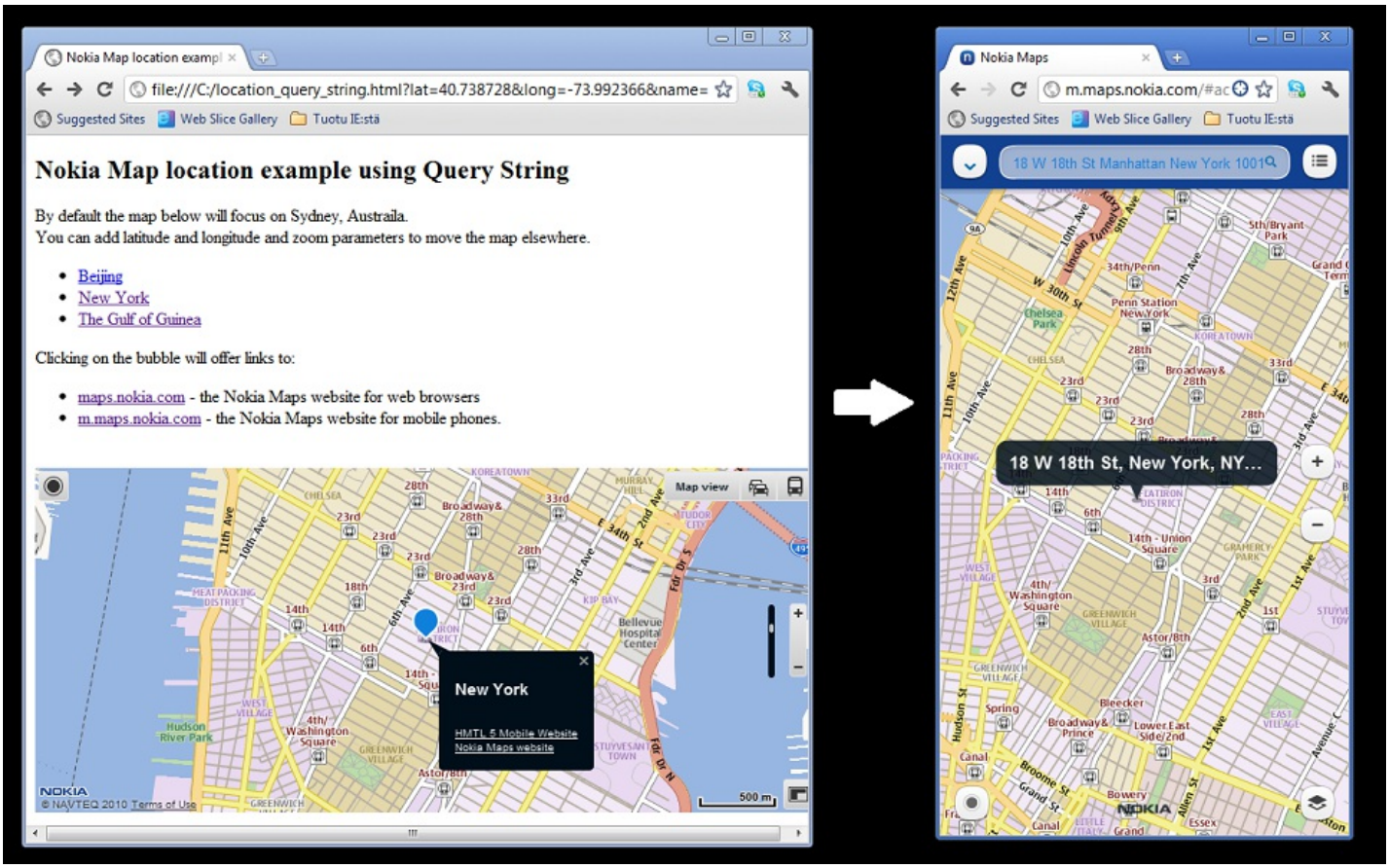
Archived: This article is **archived** because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

Equivalent deeplinks are now available directly within the Nokia Maps website itself, for example see : http://maps.nokia.com/whereis/New York. For further details read the blog entry here

This article explains how to read in query parameters to display a map at a known location, and define links to both the main http://maps.nokia.com website and the mobile HTML5 website http://m.maps.nokia.com . It is an example of the use of both Geocoding and Reverse Geocoding.

Introduction

It is sometimes useful to be able to supply a link to an **interactive** map of a known location using the Nokia Maps API in the same manner as the RESTful Map API does for static maps. This could be done to allow your user to make a search or a routing request on the ready-made http://maps.nokia.com website for example. Alternatively if your user is using an Android or iPhone, you wish to forward them to the mobile HTML5 website http://m.maps.nokia.com, so they can still access Nokia Maps. Currently neither of these websites websites offers an API directly, but since they both use the same services as the Nokia Maps API, it is relatively simple task to create a query-string-based web page which creates URLs of an appropriate format for these websites to consume.



Defining a Minimal set of Query String Parameters

The URL of an http "Get"' request consists of the path to a file location followed by a query string. The query string contains the parameters which are passed to the page. The RESTful Map API documentation contains a multitude of parameters for modifying a map. This example will replicate some of the major request parameters defining the location of a map.

- A pair of parameters for defining the longitude **long** and latitude **lat** at the center of the map. This can be used for reverse geocoding.
- A free text address **address** to define the location at the center of the map. This can be used for geocoding.
- A `zoomLevel` **zoom** for the map.

(C) Copyright Nokia 2014. All rights reserved.

- Some associated free text **name** to display on the map. Obviously additional parameters can be added in the same manner.

How to read query string parameters

The parsing of the query string is absolutely the **first** thing that must be done even before setting up the map. This can be done with a standard library function as shown below:

```
function splitQueryString() {
 var query = window.location.search.substring(1);
 var parms = query.split('&');
 for (var i=0; i<parms.length; i++) {
   var pos = parms[i].indexOf('=');
   if (pos > 0) {
     var key = parms[i].substring(0,pos);
     var val = parms[i].substring(pos+1);
     param[key] = decodeURI(val);
   }
 }
}
```

Sensible defaults can be added to the parameters - in this case we are defaulting to display a map of Sydney, Australia. The default values can then be overwritten by reading in the actual values from the query string.

```
var param = new Array();
param['long'] = "151.211111";
param['lat'] = "-33.859972";
param['name'] = "";
param['zoom'] = "14"; // Default Zoom
param['address'] = "";
splitQueryString();
```

Displaying a map with query string parameters

Having obtained a latitude. longitude and zoom level, the initial map can be set up in the usual manner:

```
  var center = new nokia.maps.geo.Coordinate(parseFloat(param['lat']),
parseFloat(param['long']));
   var map = new nokia.maps.map.Display(mapContainer, {
 // initial center and zoom level of the map
 center: center,
 zoomLevel: param['zoom'],
 components: [  etc ....
```

How to generate links to maps.nokia.com and m.maps.nokia.com

Depending upon the parameters passed we may have a latitude, longitude or alternatively we may have an address. We need **both** in order to create a link to m.maps.nokia.com. Therefore we need to create use the `nokia.places.search.manager` and either make a `geocode()` request or a `reverseGeoCode()` request as necessary.

```
var locName =  param['name'];
var addressQuery = param['address'];

if (addressQuery !=  ""){
    nokia.places.search.manager.geoCode({
```


(C) Copyright Nokia 2014. All rights reserved.

```
                searchTerm : addressQuery,
                onComplete: processResults
            });
} else {
    nokia.places.search.manager.reverseGeoCode({
                latitude : center.latitude,
                longitude : center.longitude,
                onComplete: processResults
            });
}
```

The callback function `processResults()` needs build up the address if necessary (this has been removed for brevity - check the coded example) , and then pass the location and address for further processing.

```
var processResults = function (data, requestStatus) {
        // If the search has finished we can process the results
        if (requestStatus == "OK") {
   ///... build up address if necessary
   if (addressQuery != ""){
                                // This is a geocode request, center on the location
    addMarker(locName, addressQuery, data.location.position);
    map.setCenter(data.location.position);
   } else {
                               // This is a reverse geocode request, center is already
correct.
    addMarker(locName, addressText, center);
   }

  }
  } else if (requestStatus == "ERROR") {
    addMarker(locName, "", center);
  }
  });
```

The `addMarker()` function creates and adds the necessary links to an info bubble. The links are defined in the `getNokiaMapsHTML5Link()` and `getNokiaMapsLink()` methods.

```
  var infoBubbles = new nokia.maps.map.component.InfoBubbles();
  map.addComponent( infoBubbles);

function addMarker(locName, address, coordinate){
 var marker = new nokia.maps.map.StandardMarker(coordinate, {
  $html: "<h2>" + unescape(locName) + "</h2><br/>"
     + getNokiaMapsHTML5Link(address, coordinate) + "<br/>"
     + getNokiaMapsLink(coordinate)
  });

 marker.addListener("click" ,  function(evt) {
  infoBubbles.addBubble(evt.target.$html, evt.target.coordinate);
  currentMarker = evt.target;
 }, false);

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


(C) Copyright Nokia 2014. All rights reserved.

```
function getNokiaMapsHTML5Link(address, coordinate ){
    return "<a
href='http://m.maps.nokia.com/#action=search&params={%22id%22:%22826gcpvj-
8a33c3c9e61545eda32879b880d3c96f%22,%22query%22:%22" +
 escape(address) + "%22,%22slatitude%22:" + coordinate.latitude + ",%22slongitude%22:" +
coordinate.longitude + "}&bmk=1'>HMTL 5 Mobile Website</a>";
}

function getNokiaMapsLink (coordinate){
     return "<a href='http://maps.nokia.com/services/#|" + coordinate.latitude + "|" +
coordinate.longitude +
        "|"  + map.get('zoomLevel') + "|0|0|normal.day'>Nokia Maps website</a>";
}
```

How to forward directly to m.maps.nokia.com

Note: Not all phones or web browsers are able to use the m.maps.nokia.com website.

Once a URL of a link has been calculated, it is a trivial matter to change the code from displaying that link to forwarding a user on to that URL. Just use `window.location` as shown:

```
window.location=url_of_the_new_location
```

The coded example

The coded example consists of two files:

- `location_query_string.html` - displays the map of an arbitrary location based upon the query string parameters.
- `forward_query_string.html` - forwards directly to the mobile HTML5 website. This could be used as a link in a phone app for example.

(C) Copyright Nokia 2014. All rights reserved.