TECHNOLOGY: Oracle Mobile Cloud Service

Maps Within Reach

By Chris Muir

Use your business data and Oracle Mobile Cloud Service to provide useful maps to your mobile staff.

Maps play a pivotal role in many successful mobile applications. Apps such as Uber; Airbnb; and, of course, Google Maps and Apple Maps are used daily by millions, if not billions, of people. For mobile app users, knowing where the things they are interested in are located is a great benefit.

I've discussed maps before in this article space in a rather trivial use case of a single location on a map based on a human-readable address. Given the importance of maps in mobile apps, however, this is an area I want to revisit. With a combination of Oracle Mobile Application Accelerator and Oracle Mobile Cloud Service, this article will explore how to build an application that enables users to run spatial queries that return multiple locations within a radius of their current location. This might sound a bit dull and technical, but in a business context, this could be an app that enables you to query all the customers within 5,000 meters of your current location and a very handy thing for salespeople to know when they're on the road and have time to visit other customers to make more sales.

As described in my previous map article, Oracle Mobile Application Accelerator supports the ability to plot maps. Oracle Mobile Cloud Service location-based services go beyond map plotting to enable mobile developers to store and retrieve information about *places*, such as physical addresses that might represent places of interest to a business; *devices*, such as iBeacons, Eddystone, and altBeacons, that an app can interact with as the user approaches a location not easily identified by GPS; and moving *assets*, which users may want to track to know the assets' position relative to their current locations.

In this article, I'll use Oracle Mobile Cloud Service's location-based services capabilities and Oracle Mobile Application Accelerator's map capabilities to support a solution to the traveling sales force's customer location problem.

Prerequisites

To follow the steps in this article, you will need access to Oracle Mobile Cloud Service, which you can obtain by clicking the **Free Trial** button on the Oracle Mobile Cloud Service home page. (You have the option to sign up for a trial account or free cloud credits.) After signing up for the trial and receiving approval, watch and follow the instructions in this video on how to set up and provision your Oracle Mobile Cloud Service instance. You will also need to set up two users with appropriate roles to build and access the application. First,

you'll need Jeff the developer, who will build a small application with Oracle Mobile Cloud Service and Oracle Mobile Application Accelerator. Second, you'll need Mary the mobile user, who will actually use the Oracle Mobile Application Accelerator application built by Jeff. Review this small video on how to create both users. Finally, download the demo zip file that contains the files for this article, and unzip it on your desktop.

Location-Based Services

As mentioned earlier, location-based services within Oracle Mobile Cloud Service support the concepts of places, assets, and devices. At the most fundamental level, Oracle Mobile Cloud Service provides several REST APIs that enable you to create, manage, and query these data objects by ID or spatially relative to a latitude, longitude, and radius. Simply put, your business may have 1,000 customers located at different places identified by latitude and longitude, and Oracle Mobile Cloud Service will enable you to run the spatial query "Tell me all the customers within 5,000 meters."

For the purposes of this article, because I can't tell where you, the reader, reside, you're not going to be able to build an app returning customers in your locality. Rather, you're going to build an app with customers located around San Francisco, California, and assume you're also located close to these customers. As such, the first task is to load the location-based services' places data object with these San Francisco customers.

1. Log in to Oracle Cloud as Jeff the developer.
2. From the Oracle Cloud My Services Dashboard, select **Mobile Environment Service**, and on the resulting screen, click **Open Service Console**.
3. For the purposes of testing in this article, you need to create an Oracle Mobile Cloud Service mobile back end. Click the hamburger menu, expand **Applications**, and click **Mobile Backends**. Select **New Mobile Backend**. In the resulting dialog box, enter `OraMagTest` for the name, enter the same for the description, and then click **Create**.
4. In the Oracle Mobile Cloud Service UI, click the hamburger menu, expand **Applications**, and select **APIs**. Scroll to the bottom of the new page, and select **Location Management**.

The resulting screen exposes the REST endpoints for maintaining places, assets, and devices in Oracle Mobile Cloud Service. By default, these will be empty for a new Oracle Mobile Cloud Service account or trial. Next, you'll see the **POST Add Places** endpoint to create three customers located at different latitudes and longitudes for the Oracle Mobile Cloud Service application to use.

5. Select **POST Add Places**.
6. From the downloaded and unzipped file, copy the contents of the customers.json file into the body field of the **POST Add Places** request.
7. In the Authentication section below, for **Mobile Backend**, select the **OraMagTest** mobile back end you created earlier, enter the username and password for Jeff the developer, and click the **Test Endpoint** button. The resulting status code 200 response will include the places you just successfully inserted into the location- based services.

Building an API for Oracle Mobile Application Accelerator

For Oracle Mobile Application Accelerator to display the location-based services places you just inserted into Oracle Mobile Cloud Service, it must access this data through a REST API. Unfortunately, Oracle Mobile Application Accelerator can't talk directly to the location-based services REST API to extract places, so you must create your own custom API in Oracle Mobile Cloud Service to wrap a call to the places API. I've covered how to create REST APIs in previous articles in *Oracle Magazine*, so I'll avoid covering this again. Instead you'll import a prebuilt API that will do the work for you.

8. In the Oracle Mobile Cloud Service UI, click the hamburger menu, expand **Applications**, select **Packages**, and then select **New Import**.
9. In the first step of the Import wizard, click **Choose a package file**, and in the dialog box, select the **package-spatial.zip** file from the collection of files you downloaded and extracted.
10. In the remaining steps of the wizard, click **Next** and **Finish** to complete the wizard.
11. In the Oracle Mobile Cloud Service UI, click the hamburger menu, expand **Applications**, and select **APIs**. On the resulting screen, enter `spatial` in the **Filter APIs** field, and then press Return. Select the **spatial** API in the resulting list, and then click the **Test** button.
12. In the resulting test page for the newly created API, you will see an endpoint preselected to GET /customers. This requires three parameters: latitude, longitude, and radius. These three parameters are fed into the following Node.js code you just imported:

```
13.    module.exports = function(service) {
14.
15.      service.get('/mobile/custom/spatial/customers', function(req,res)
   {
16.        var sdk = req.oracleMobile;
17.
18.        var latitude = parseFloat(req.query.latitude);
19.        var longitude = parseFloat(req.query.longitude);
20.        var radius = parseInt(req.query.radius);
21.
22.        const spatialQuery = {"inGeoFence":{"gpsCircle":{
23.          "latitude":latitude,"longitude":longitude,"radius":radius}}};
24.
25.        sdk.location.places.query(spatialQuery).then(
26.          function (success) {
27.            res.send(success.statusCode, success.result);
28.          },
29.          function (error) {
30.            res.send(500, error.error);
31.          });
32.      });
33.    };
```

As shown in the Node.js code, for the specific GET /mobile/custom/spatial/customers endpoint router function, the three query parameters are extracted from the request and then wrapped in a JSON object and fed into the SDK location.places.query method, which runs a spatial query on places within the radius of the latitude and longitude.

13. Still on the test page, enter a latitude of 37.66, longitude of -121 (note the negative), and radius of 70000 meters (enter the number only, not the units). As you can see, two places are returned within that radius. Locate the latitude and longitude of each place within the JSON payload (provided in the customers .json file), because this information will be used in the Oracle Mobile Application Accelerator app map screen you will build in a moment.

Build Your Oracle Mobile Application Accelerator App

With the places API (named "spatial") in place, you can now build your Oracle Mobile Application Accelerator app with a map. The goal will be to display on a map all the location-based services places within a radius of your current location.

However, as explained earlier, there is one snag for demonstration purposes, and that is that I simply don't know where you are located. As such, when you created the places earlier in this article, they may be nowhere near you. For this demo app, you're going to hardcode your latitude and longitude, but while proceeding through the steps, I'll show you how to remove the hard coding and use your current location for the real application you may choose to build after reading this article.

14. In the Oracle Mobile Cloud Service UI, click the hamburger menu; and this time, rather than expanding the **Applications** menu, select it.
15. On the resulting screen, select the mobile apps blue icon/box.
16. Choose **New Application** when Oracle Mobile Application Accelerator prompts you to, and enter the application name `Customers` in the first step of the wizard.
17. In the next step of the wizard, "First Screen," ensure that the **Simple Screen** option is selected, and then click **Next**.
18. For **Screen Title**, enter `Customers`; click **Next**; and then complete the wizard by clicking **Next** and then **Create**.
19. On the resulting Oracle Mobile Application Accelerator designer screen, under **Components**, drag the map icon into the mobile app. By default, the map will display Oracle's headquarters, in Redwood Shores, California.
20. With the map selected, select the **Properties** tab on the right and make sure **Best Fit for Points** is selected.
21. Still on the right, select the **Data** tab. In the resulting options, make sure **Multiple Points** and **Geolocation Code** are selected.
22. Still on the **Data** tab, click the **Business Object** list and select **Browse Service Catalog**.

In Oracle Mobile Application Accelerator, APIs are called business objects. In the following steps, you will bind the map to your places API, mapping the data fields of the map to the data supplied by the API as well as supplying values for the parameters.

23. In the resulting dialog box, locate and select the **spatial v1.3** API you just imported and click **Next**. On the resulting screen, select **customer** as the business object and then click the **Select** button.
24. In the the wizard, make sure **I want to use location** is selected and then click **Next**.

The resulting wizard page is used to bind the API (aka business object) to the map. The first tab represents the mapping of the API's returned payload to the required fields of the map on the right. The second tab represents the query parameters required by the API, which you must satisfy in some form. As you'll recall, the API requires the user's current latitude, longitude, and radius to search for places.

With the **Data** tab selected, note that the map wants a location value on the right. A location value is made up of a latitude and a longitude. With the **Business Object** option selected, note the data fields available from the API. Latitude and longitude are available in the **address** and **gpsPoint** fields.

25. Select the **address** and **gpsPoint** fields until you see the **latitude** and **longitude** fields. Drag and drop the **latitude** field into the **Location value** field, and then drag and drop the **longitude** field into the **Location value** field. Ensure that the order of the fields is **latitude** and then **longitude**.
26. Click the **Live Preview** button, and note that the map shows one marker on a very boring background. This unimaginative location in the hills of San Francisco is derived from the "mock" data of the spatial API, which currently returns only a single latitude and longitude value. You'll see more-exciting data in a moment.
27. Click the **Next** button, which will take you to the API query parameter screen, showing latitude, longitude, and radius.

To satisfy the original
requirements for this app to
show places within a radius of the user's current location, you would select the **Device Service** option and then copy in the device's **Current Latitude** and **Current Longitude** fields. However, as explained earlier, for demo purposes, you'll hardcode these values.

28. To hardcode the values, select the **Fixed Value** box (on the left) and drag and drop the **Fixed Value** field into all three fields (latitude, longitude, and radius). Then enter `37.66` for **latitude**, `-121` (note the negative) for **longitude**, and `70000` for **radius**.
29. Click **Finish**.

On returning to the Oracle Mobile Application Accelerator designer, you may be a little disappointed to discover the single marker against a boring background again. By default, like the **Live Preview** option, the Oracle Mobile Application Accelerator designer shows only test data derived from the API. To see live data, you must run the application.

30. Click the **Test** button (the gray right arrow at the top right); when prompted, enter Mary the mobile user's username and password; and click **Sign In**.
31. The resulting app displays the map, and given the virtual location of your app user, you can see the two places within the 70-kilometer radius of your current latitude/longitude, 37.66/-121, as shown in **Figure 1**.

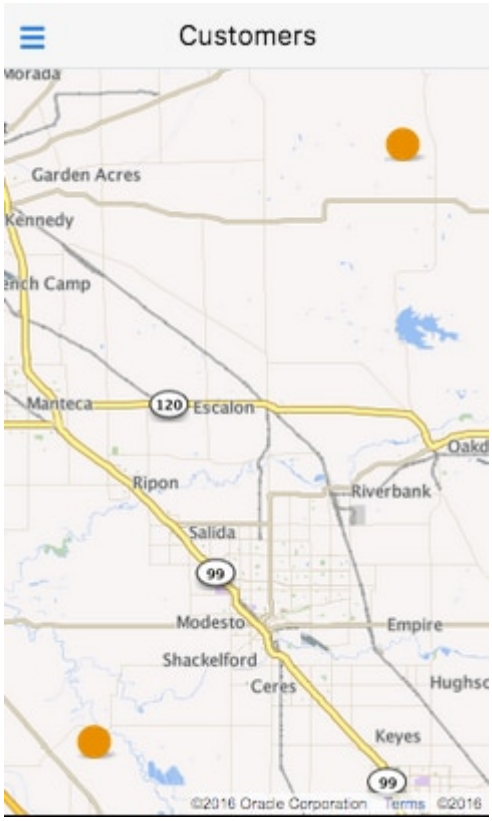

Figure 1: Two customers appear on your app map.

Conclusion

What excites me most about the mapping and spatial functionality provided through Oracle Mobile Application Accelerator and Oracle Mobile Cloud Service is that I've visited absolutely loads of customers who have address and location data in their enterprise databases but have never actually been able to display the data on a map. It's mostly raw data, and it comes alive only when it's on a map. As you saw in my previous article, where I plotted human-readable addresses on a map, and in this article, where I plotted numerous markers based on latitude and longitude, this simple-to-build functionality is now within reach, literally, of your mobile users.

Chris Muir is a senior principal product manager for mobility, cloud, and development tools at Oracle.