The demo App is for Titanium Studio. It first gets authorized by Mojio server, then obtains the information of a vehicle and shows the location of the vehicle on the map.

1. The UI Design

The Titanium Studio employs the model-view-controller (MVC) paradigm, which separates the application into three different components: model, view and controller. The demo app doesn’t have a complex business logic, so the model folder is empty(/app/model). The demo App has three buttons: login, obtain token, get data, which are defined in the index.xml file as

<Alloy>

<Window>

<ImageView id = "imageView" ></ImageView>

<Label id="l">Example App</Label>

<Button id = "LoginButton" onClick="LoginButton">Login </Button>

<Button id = "GetTokenButton" onClick="GetTokenButton"> GetToken </Button>

<Button id = "ObtainDataButton" onClick="ObtainDataButton"> Obtain Data </Button>

</Window>

</Alloy>

where the item Imageview shows a Mojio Logo on and the item label shows the text. There are three items define the three buttons. Take the LoginButton as an example,

<Button id = "LoginButton" onClick="LoginButton">Login </Button>

the ID of the button is “LoginButton”, the callback function for the click event is LoginButton, which is defined in the index.js file as we will discuss later.

The style of these buttons and the Mojio logo image is defined in the style file index.tss, which can be found under the folder /app/styles. Still take the LoginButton as example,

"#LoginButton":{

width: "35%", //Titanium.UI.SIZE,

height:40,

top:180

}

It defines the width, height and the gap between the button and the screen top.

1. The Control Logic

The actual behavior of the App is implemented in the index.js file, which can be found at /app/controllers/. The logic flow of this App has five steps: initialize the SDK, authorization, obtain the authorization token, fetch data, show data on map.

* 1. INITIALIZING THE SDK

First you will want to download the SDK for Titanium Studio. Once you have downloaded and included the appropriate SDK, connecting to our API is as simple as:

var MojioClient = require('mojiojs/lib/titanium/MojioClient');

var App, MojioClient, buildMojioMap, config, mojio\_client;

config = {

application: '21de13d2-a016-4fed-8e5e-43f0432ea717', //Your application ID

redirect\_uri: 'myfirstappeceleraterapp://', //Redirect to your App

hostname: 'api.moj.io',

version: 'v1',

port: '443',

scheme: 'https',

live: false, // This will connect your app to the sandbox server, replace with true to go live.

appname : "myfirstappeceleraterapp"

};

mojio\_client = new MojioClient(config);

* 1. AUTHENTICATING A MOJIO USER

Many of our API calls require an authorized user to be associated with the SDK requests. In order to authenticate a user, you must redirect to the Mojio authentication server. The authorization of the demo App is done by calling the SDK API mojio\_client.authorize(.), which is defined the SDK you just downloaded. The SDK will open a webview with the url we set in the initialization step, which will lead to the authorization page. On that page, just simply put your credential, i.e., user name and password. Then click login, a new page will pop up asking whether grant Mojio to access your information in your Mojio account. Click the allow button will grant the demo App to access your account (otherwise you can logout and finish the demo).

function LoginButton() {

Ti.API.info("mojio\_client.isauthorized() " + mojio\_client.isauthorizedz());

if(mojio\_client.isauthorized()===false) {

//use webview to obtain the token

var webview = mojio\_client.authorize(config.redirect\_uri);

window = Titanium.UI.createWindow();

window.add(webview);

window.open({modal:true});

} else {

mojio\_client.token(function(eren ror, result) {

if (error) {

alert("Authorize Redirect, token could not be retreived:" + error);

} else {

alert("Authorization Successful.");

Titanium.UI.createAlertDialog({title:'Your App has been authroized!', message:result}).show();

Ti.API.info("result " + result);

accessToken = result;

}

});

}

}

Before calling the .authorize() API，the demo App will first check if it has been authorized by calling the API .isauthorized(). If it has been authorized, the APP will try to get the authorization token by calling mojio\_client.token(function(eren ror, result).

* 1. Obtain the Access Token

If you click the GetToken button, this call back function will be called,

function GetTokenButton(){

mojio\_client.token(function(error,result) {

if (error) {

alert("Authorize Redirect, token could not be retreived:" + e);

} else {

alert("Authorization Successful.");

Titanium.UI.createAlertDialog({title:'Your App has been authroized!', message:result}).show();

Ti.API.info("result " + result);

}

});

}

It will set the token to the Mojio client, which will be used to obtain the data, post data etc.

* 1. Fetch Data

If you click the Obtain Data button, the corresponding call back function will be called,

function ObtainDataButton() {

var Vehicle = mojio\_client.model("Vehicle"); // Gets a trip model schema.

Ti.API.info("Vehicle:" + Vehicle);

mojio\_client.get(Vehicle, {}, function(error, result) {

var e = error.error;

//var test = this.responseText;

if (e) {

Ti.API.info("get data returns error:" + e);

console.log(error); // Some error occured.

} else {

var vehicle\_data = mojio\_client.getResults(Vehicle, result); // Helper function to get the results.

lat = parseFloat(vehicle\_data[0].LastLocation.Lat);

lng = parseFloat(vehicle\_data[0].LastLocation.Lng);

Ti.API.info("lat is: " + lat);

Ti.API.info("lng is: " + lng);

loadmap();

}

});

}

If the data is fetched successfully, we will take the latitude and longitude of the corresponding vehicle and show it on the map as explained in the next step.

* 1. Show Data on Map

If the vehicle data is fetched successfully, its location will be shown on the map by calling the function loadmap():

function loadmap() {

map1.region.latitude = lat;

map1.region.longitude = lng;

Ti.API.info("map LAT: " + map1.region.latitude);

var anImageView = Ti.UI.createImageView({

image : '/images/pin.jpg', //setting label as a blob

width : '20',

height : 'auto',

});

var random = MapModule.createAnnotation({

latitude: lat,

longitude: lng,

image: anImageView,

pincolor: MapModule.ANNOTATION\_AZURE,

draggable: false

});

var mapview = MapModule.createView({

mapType: MapModule.NORMAL\_TYPE,

region: {latitude: lat, longitude: lng, latitudeDelta: 0.1, longitudeDelta: 0.1 }

});

mapview.addAnnotation(random);

window.add(mapview);

window.open();

Ti.API.info("map LAT3: " + map1.region.latitude);

alert('last location obtained successfully!');

}