Add the Weather Company Data and Watson Text to Speech Services to an Android App

One of the most common services used in Mobile Apps is Geolocation. Knowing the latitude and longitude corresponding to the location of a device, can be the input of other useful services such as Weather Forecasting; and adding “speaking” capabilities to an App is a simple, but powerful way to innovate.

Following this guide, you will learn how to use the Weather Company Data and Watson Text to Speech services in Bluemix, while adding the corresponding capabilities to an Android App that shows the current address where the device is located, according to its geographic coordinates.

As pre-requisites you will need:

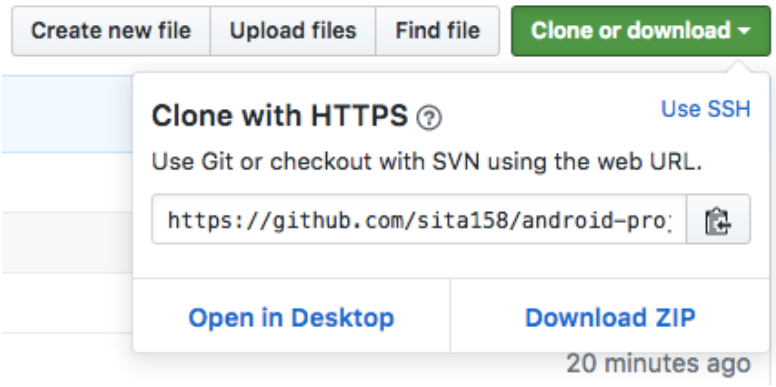
- Android Studio
- An Android physical device to test the App
- A valid Bluemix account

Getting the code and testing the version 0 of the App

To get the base code for this exercise, go the following GitHub repository:

https://github.com/sita158/android-projects-GeoLocation-App

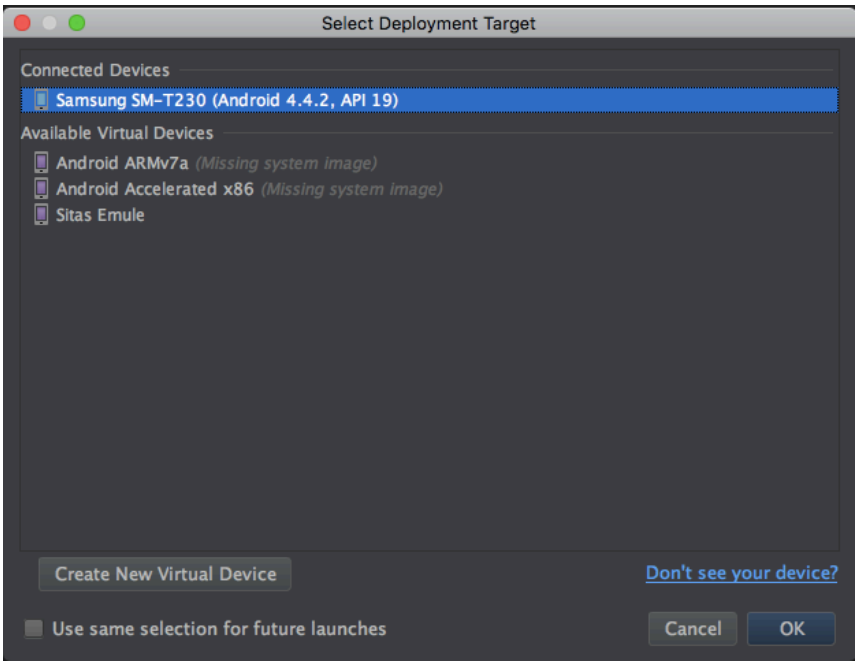
Click on the “Clone or Donwload” button and get the project in a Zip file:



Unzip the downloaded file and open Android Studio. Select the “Open an existing Android Studio Project” option and navigate to the path where the downloaded file was unzipped.

To test the original version of the App, connect an Android device to your computer and click on the “Run app” button:

Ensure your physical Android device is selected and click “Ok”:

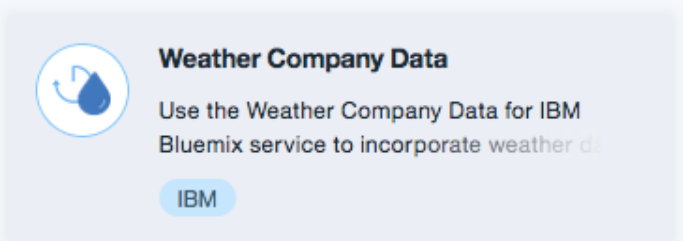


When executing the App, it will show the address corresponding to the last known location of the device. If necessary, click the “Update” button:

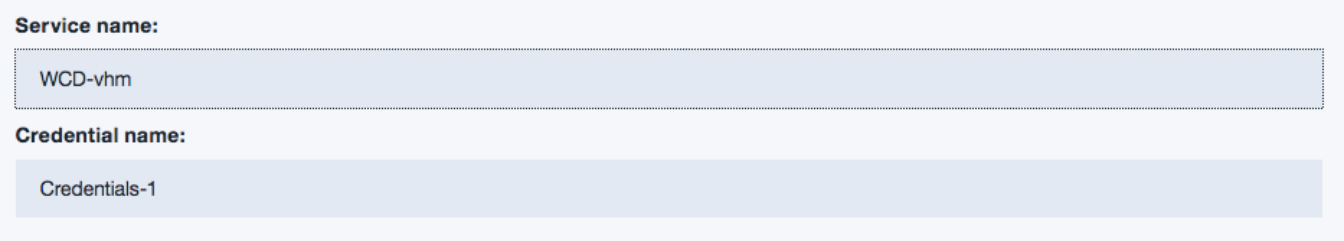
Now you are ready to add the Weather Company Data and the Watson Text to Speech services to the App.

Adding current weather for the location with the Weather Company Data Service

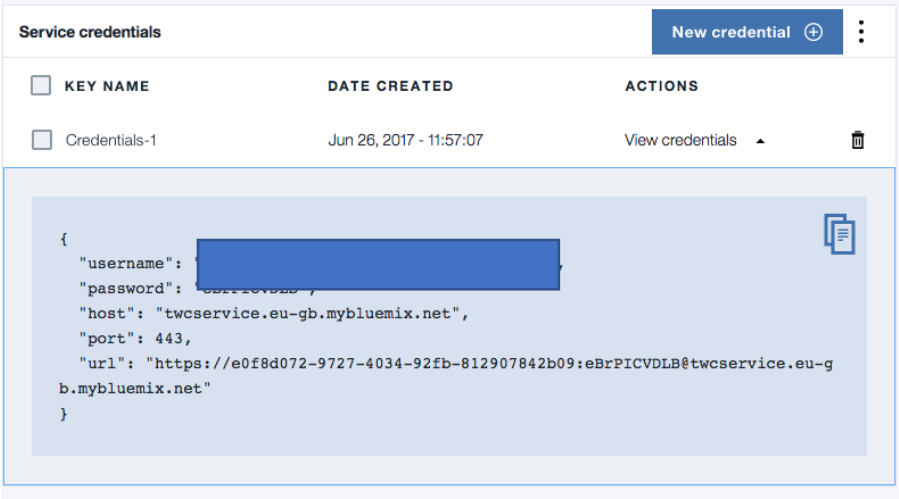
Navigate to bluemix.net in a web browser and Log-in with your valid Bluemix account. In the Services Catalog go to the “Data and Analytics” section and click on the Weather Company Data Service:



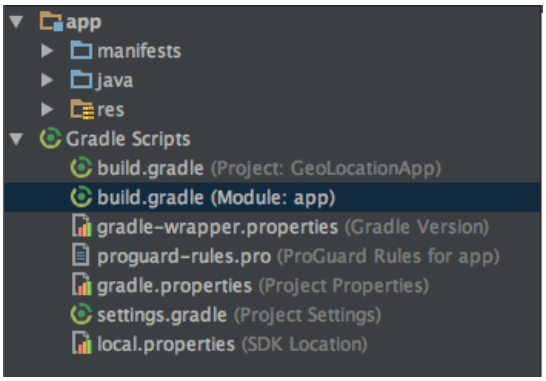
Type a name for the service. Since the name must be unique, use an identificator such as your initials or a number you will remember. Click on “Create”:



Once created navigate to the “Service Credentials” menu, and click on the arrow next to “View Credentials”. Keep these values at hand, since they are going to be used in the next steps to code the REST invocation to the Weather Company Data Service:



Now go to Android Studio and open the build.gradle file corresponding to the Module:app section of the Project:



Add the following line to the “dependencies” section of the file (you will find the code to copy and paste in the snippets/snippets file in the Project you downloaded from GitHub):

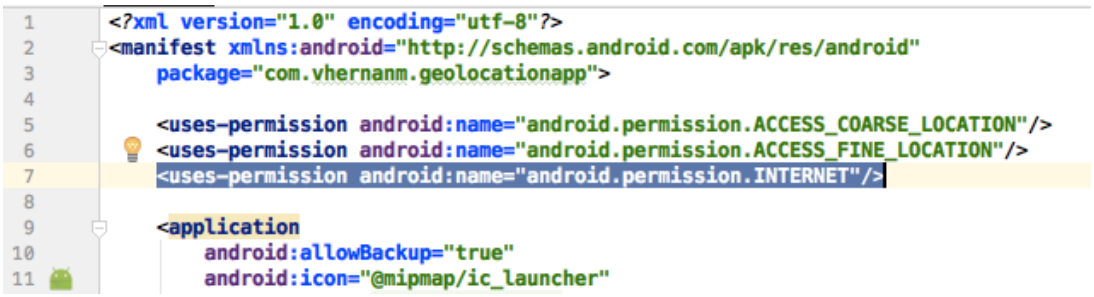
```
compile 'com.android.volley:volley:1.0.0'
```

Click on the “Sync now” link that appears in the top right corner of Android Studio, and ensure the synchronization ends without errors.

Navigate to the app->manifests section of the Project and open the AndroidManifest.xml file. In the “uses-permissions” section add the following line:

```
<uses-permission android:name="android.permission.INTERNET"/>
```

It should look like:



Go to the app->res->layout section of the project and open the activity_main.xml file. Add the following code to create two new labels that appear just at the bottom of all the already existing ones:

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="The weather at that location:"
    android:textSize="15dp"
    android:textAlignment="center"
    android:layout_marginTop="20dp" />
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="-- Weather --"
    android:textSize="15dp"
```

```
android:textAlignment="center"
android:layout_marginTop="20dp"
android:id="@+id/weather"/>
```

The new code should look like this (the newlly added piece is selected):

Now go to the app->java section of the project and open the MainActivity.java file. Locate the updateUI() function and insert in it the following code. Be careful to add all the necessary classes to the "import" section of the class:

```
String url = "<WeatherCompanyDataServiceURL>/api/weather/v1/geocode/" +
mLastLocation.getLatitude() + "/" + mLastLocation.getLongitude() +
"/observations.json";


StringRequest stringRequest = new StringRequest(Request.Method.GET, url, new
Response.Listener<String>() {

    @Override

    public void onResponse(String response) {

        Log.d("App", "Response: " + response);


        try{

            JSONObject weatherObject = new JSONObject(response);

            JSONObject observation = weatherObject.getJSONObject("observation");

            String currentWeather = observation.getString("wx_phrase") + ", " +
observation.getString("temp") + "°F";


            TextView weather = (TextView) findViewById(R.id.weather);

            weather.setText(currentWeather);

        } catch (JSONException e){

            Log.d("App", e.toString());

        }


    }

}, new Response.ErrorListener() {

    @Override

    public void onErrorResponse(VolleyError error) {

        Log.d("App", "Error: " + error.getMessage());

    }
```

```
})
{
    @Override
    public Map<String, String> getHeaders() throws AuthFailureError {
        HashMap<String, String> params = new HashMap<String, String>();
        params.put("Content-Type", "application/json");
        String creds = String.format("%s:%s","<username>","<password>");
        String auth = "Basic " + Base64.encodeToString(creds.getBytes(),
Base64.DEFAULT);
        params.put("Authorization", auth);
        return params;

    }
};

RequestQueue requestQueue = Volley.newRequestQueue(MainActivity.this);
requestQueue.add(stringRequest);
```

Locate the following strings (marked in red in the above code), and substitute with the appropriate values, according to your Weather Company Data Service Credentials:

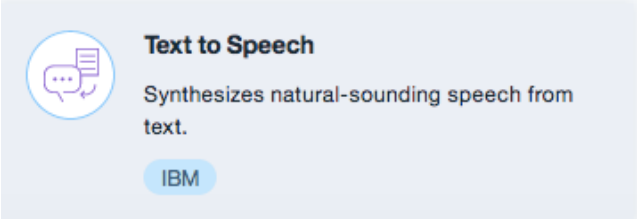
- <WeatherCompanyDataServiceURL> ……. Place the “URL” value from the service credentials
- <username> …… Place the “username” value from the service credentials
- <password> …… Place the “password” value from the service credentials

Once done, you can connect again your Android device, and use the “Run app” button to test the App. You should see something similar to the next screenshot:

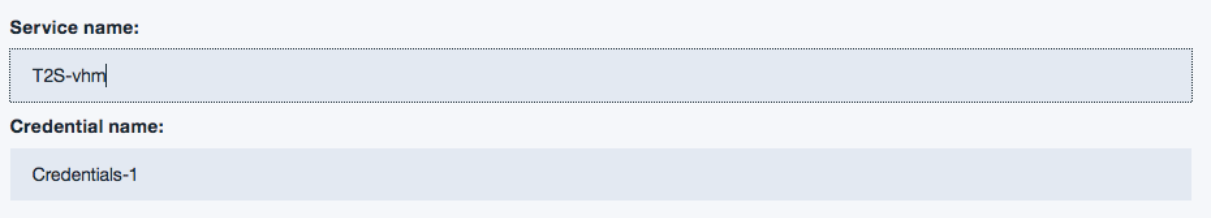

100% 20:25
GeoLocationApp
Welcome to the GeoLocation App
Your current location is:
Morelos 10
Pueblo de Santa Lucía
15000 Ciudad de México, CDMX
México
The weather at that location:
Cloudy, 63°F
Update

Adding “speaking” capabilities with the Watson Text to Speech Service

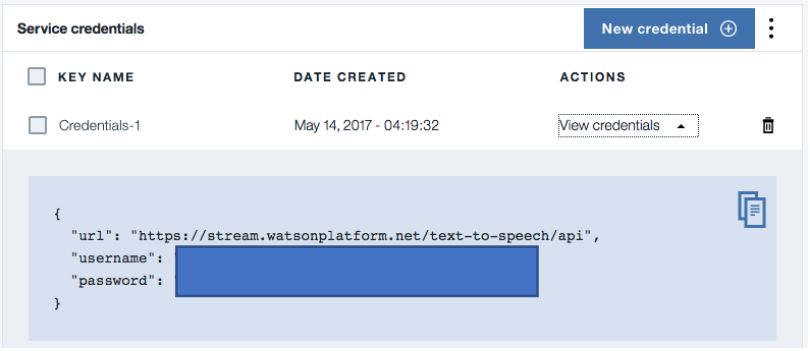
Navigate to bluemix.net in a web browser and Log-in with your valid Bluemix account. In the Services Catalog go to the “Watson” section and click on the Text to Speech service:



Give the instance of the service a unique name using, for example, your initials, and click the “Create” button:

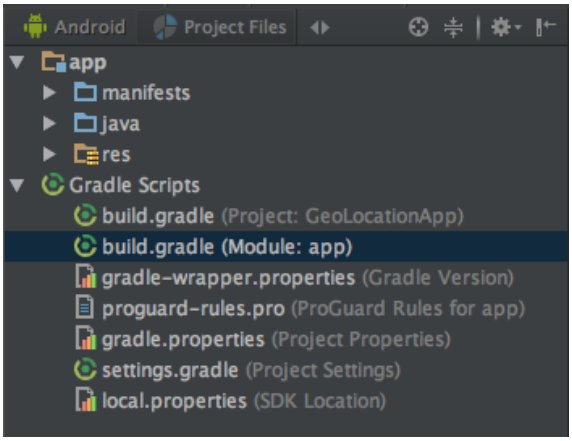


Once created, navigate on the “Service Credentials” menu and display them by clicking on the arrow next to “View Credentials”:



You will use this information later on the exercise when configuring access to the Text to Speech Service.

Go to Android Studio and navigate to the build.gradle file in the Module:app section of the Project:



Add the following lines in the “dependencies” section and click on the “Sync now” link by the right top corner of the Android Studio Window. Make sure the synchronization ends without errors:

```
compile 'com.ibm.watson.developer_cloud:text-to-speech:3.7.2'
compile 'com.ibm.watson.developer_cloud:android-sdk:0.2.3'
```

Navigate to the app->res->layout section of the Project and open the activity_main.xml file. Add the following lines right after the last button to create a new button to invoke the read function of the App:

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAllCaps="false"
    android:text="Read"
```

```
    android:onClick="read"/>
```

The code should look like this:

Navigate to the app->java section of the Project and open the MainActivity.java file. Place the read() fuction right after the onConnectionFailed() function and the AddressResultReceiver class definition:

```
public void read(View view){
    final TextToSpeech service = new TextToSpeech();
    service.setUsernameAndPassword("<username>", "<password>");

    final String frase = mAddressOutput;
    Thread thread = new Thread(new Runnable(){
        public void run() {
            try {
                StreamPlayer streamPlayer = new StreamPlayer();
                streamPlayer.playStream(service.synthesize(frase, Voice.ES_SOFIA,
AudioFormat.WAV).execute());
            }
            catch (Exception e) {
                e.printStackTrace();
            }
        }
```

```
    });


    thread.start();
}
```

Substitute the “username” and “password” values marked in red, with those corresponding to your service credentials.

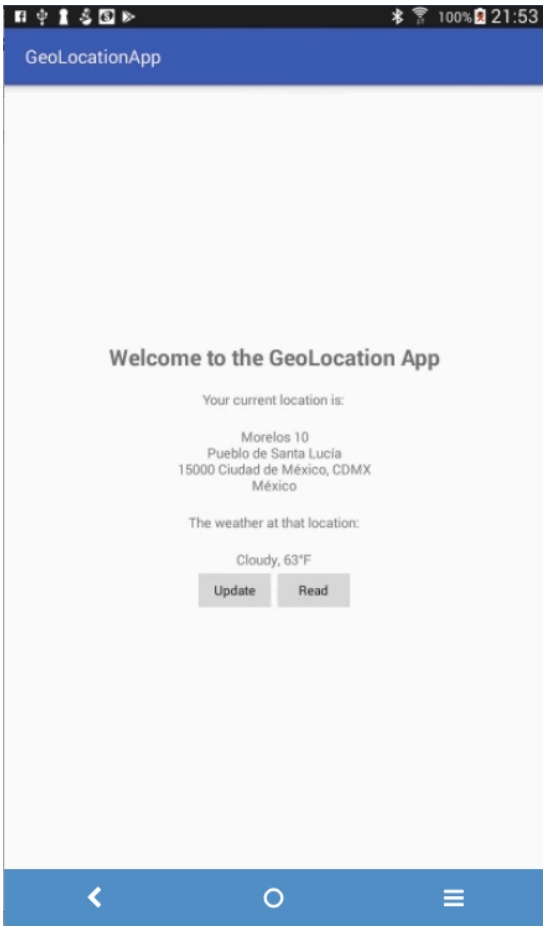
Check the following line, and modify the Voice parameter according to your language and preferred voice:

```
streamPlayer.playStream(service.synthesize(frase, Voice.ES_SOFIA,
AudioFormat.WAV).execute());
```

A list of the available languages and voices can be reviewed in the following link:

https://www.ibm.com/watson/developercloud/text-to-speech/api/v1/?java#get_voice

Use the “Run App” button to test the App on your connected device. Once you have the last location of the device displayed, click the “Read” button and listen to the Text to Speech service reading the address.



You have successfully completed this recipe!